Hierarchical Gaussian Mixture Model Splatting for Efficient and Part Controllable 3D Generation

Supplementary Material

1. Datasets and Metrics

Details of Data. We provide the total number of objects used for training, the number of views rendered per object for obtaining Gaussian primitives and the distribution of camera poses used for rendering in Tab. 1. For image-to-3D dataset 3DBiCar [6], we employ the script provided by the authors to sample additional models to fit Gaussian primitives. We use the same filter from LGM [8] to get the subset of Objaverse [3] dataset. We use the render script to render ShapeNet dataset and 3DBiCar dataset.

Dataset	#Objects	#Views/object	Rotation	Elevation
ShapeNet Car	7,462	150	$[0, 2\pi]$	$[\frac{1}{6}\pi, \frac{1}{2}\pi]$
ShapeNet Chair	6,775	150	$[0, 2\pi]$	$[\frac{1}{6}\pi, \frac{1}{2}\pi]$
OmniObject3D	5,795	100	$[0, 2\pi]$	$[0, \frac{1}{2}\pi]$
3DBiCar	6,500	150	$[0, 2\pi]$	$[\frac{1}{6}\pi, \frac{2}{3}\pi]$
Objaverse	82,575	150	$[0, 2\pi]$	$[0, \frac{2}{3}\pi]$

Table 1. Details of each dataset.

Mesh Evaluation Metrics We sample $2,048$ points from the mesh extracted from generated Gaussian primitives and compute the evaluation metrics. We follow prior works [2, 4] evaluating Minimum Matching Distance (MMD), Coverage (COV), and FPD. For MMD and FPD, lower is better; for COV, higher is better.

$$\text{MMD}(S_g, S_r) = \frac{1}{|S_r|} \sum_{Y \in S_r} \min_{X \in S_g} D(X, Y) \tag{1}$$

$$\text{COV}(S_g, S_r) = \frac{|\{\arg\min_{Y \in S_r} D(X, Y) | X \in S_g\}|}{|S_r|} \tag{2}$$

where $D(X, Y)$ is the Charmer distance of point cloud X and Y. FPD, analogous to FID, is computed by first extracting features from the sampled point clouds and then applying the Fréchet distance formulation to calculate the FPD:

$$\text{FPD} = \|\mu_r - \mu_g\|^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}) \tag{3}$$

where μ_r and μ_g represent the mean of the reference and generated features extracted from PointNet++, respectively; Σ_r and Σ_g are the covariance matrices of the reference and generated features extracted from PointNet++, respectively.

2. Implement Details

HGMM Splatting Construction. We employ 0-degree spherical harmonics to obtain Gaussian primitives. This approach significantly reduces the dimensionality of the Gaussian primitives while preserving fitting quality, thereby simplifying the complexity of diffusion modeling. For other optimization settings, we keep them consistent with the original 3D Gaussian Splatting.

To construct the *explicit component*, we set the HGMM tree level to 6. For the building of *implicit component*, we design a 6-layer decoder with attention and MLP splits, utilizing a global latent $\mathbf{z} \in \mathbb{R}^{512}$ to hierarchically divide Gaussian primitives into 256 parts. At the final layer, each part-level latent is represented as $\mathbf{z}_i^{l=6} \in \mathbb{R}^{32}$. We use multiple MLPs to extract Gaussian parameters from the part-level latents at each layer. The output of MLPs corresponds to the parameters:

$$\left\{\hat{\pi}_i^{l=d} \in \mathbb{R}, \mu_i^{l=d} \in \mathbb{R}^{14}, \hat{U}_i^{l=d} \in \mathbb{R}^{14\times14}, \sqrt{\lambda}_i^{l=d} \in \mathbb{R}^{14}\right\}, \tag{4}$$

which are used afterward to create the parameters:

$$\Omega_i^{l=d} = \left\{\hat{\pi}_i^{l=d}, \mu_i^{l=d}, \Sigma_i^{l=d}\right\} \tag{5}$$

of each Gaussian in the HGMM. We employ multiple MLPs to predict feature vectors in $\hat{U}_i^{l=d}$. We ensure that the mixture weights sum to probability 1 by applying the $softmax$ to all the node weights $\pi_i^{l=d}$ in the group of siblings. The covariance Σ_i is derived using the eigen-decomposition:

$$\Sigma_i = U_i^{-1} D_i U_i, \tag{6}$$

where D_i is a diagonal matrix with the vector λ_i as its diagonal entries, and U_i is a unitary matrix obtained by applying the Gram-Schmidt orthogonalization process to $\hat{U}_i$. This decomposition ensures that Σ_i remains a positive definite matrix (PSD), as it relies on positive real eigenvalues and their corresponding eigenvectors, represented by the columns of U.

Condition Encoder. For image-to-3D generation, we adapt the pretrained DINO ViT-B/16 [1] to encode the 512×512 conditional images into conditional feature tokens $\mathbf{c} \in \mathbb{R}^{768}$. For text-to-3D creation, we utilize CLIP-L/14 [7] to encode the text prompts into $\mathbf{c} \in \mathbb{R}^{512}$ conditional feature tokens.

Controllable Implicit Latents Generation. For controllable implicit latents generation, we add additional attention module to the decoder in HGMM Splatting construction and remove the MLPs for extracting GMM parameters

Module	Parameters	**ShapeNet Car**	**ShapeNet Chair**	**OmniObject3D**	**3DBiCar**	**Objaverse**
Diffusion	Diffusion steps	1,000	1,000	1,000	1,000	1,000
	Noise schedule	Cosine	Cosine	Cosine	Cosine	Cosine
	Inference sampler	DPM-solver [5]	DPM-solver [5]	DPM-solver [5]	DPM-solver [5]	DPM-solver [5]
Network	Channels	384	384	384	384	512
	Num blocks	48	48	48	48	96
	k	8	8	8	8	8
	m	4	4	4	4	4

Table 2. Detailed configuration of rigorous explicit splatting generation.

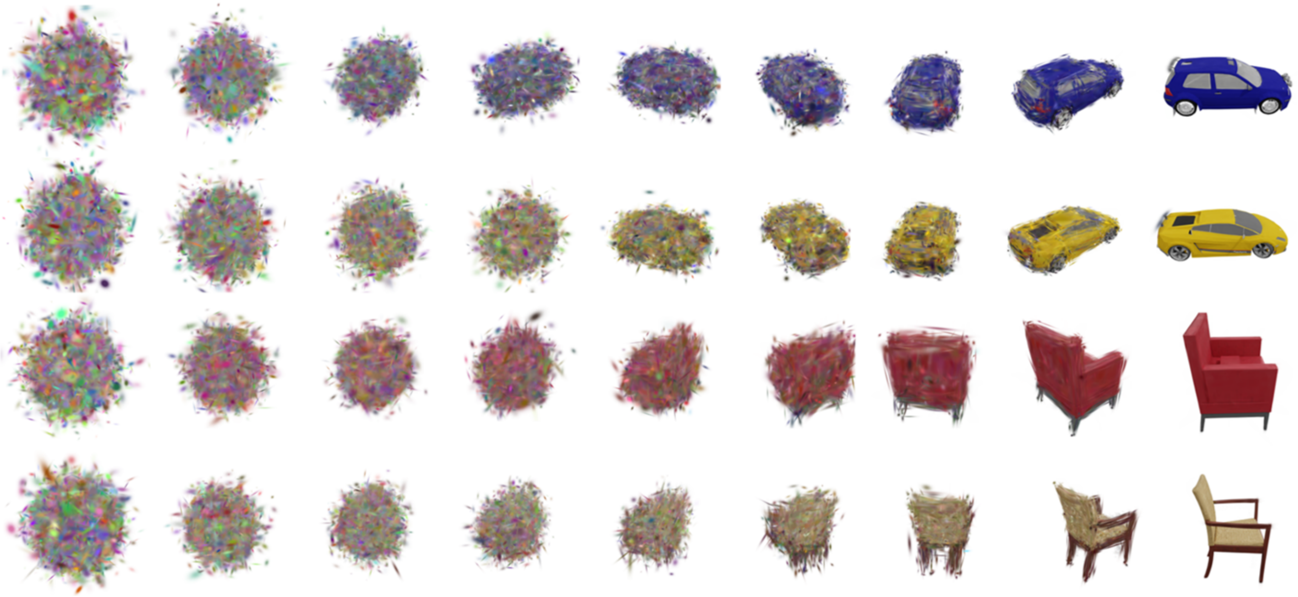


Figure 1. Denoising process in generation.

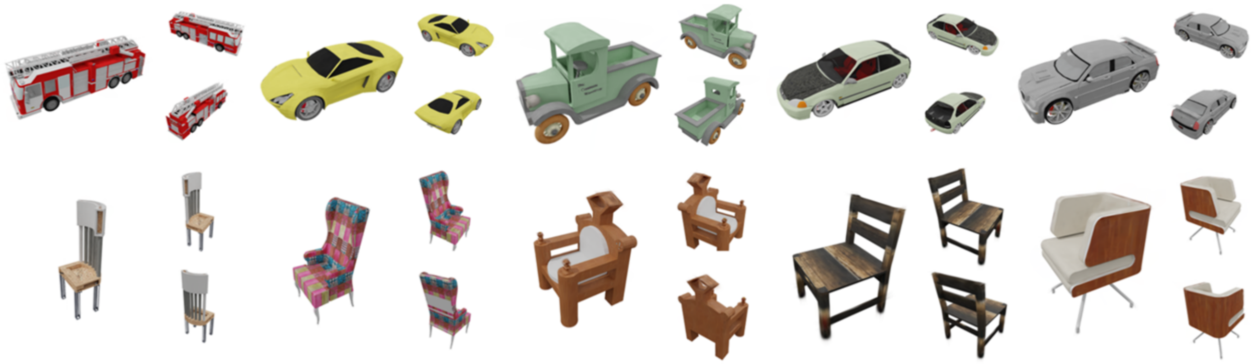


Figure 2. More qualitative results of unconditional generation.

to formulate the diffuser. We set the diffusion step to $1,000$ and use cosine noise schedule. For inference, we use DPM-Solver [5] to sample implicit latents.

Rigorous Explicit Splatting Generation. For rigorous explicit splatting generation, we report our config in Tab. 2, including noise schedule, diffusion steps, inference sampler, model channels, and the number of blocks. For the large-scale Objaverse dataset, we opt for a larger model to accommodate its complexity. We also report the connectivity of the input graph, denoted by k, and the connectivity of the graph used for tree scanning, denoted by m. We use the AdamW optimizer and apply an exponential moving average (EMA) with a rate of 0.9999 during training. To train Diffusion Mamba, we normalize the input data and then perform de-normalization on the predicted Gaussian primitives before rendering, ensuring numerical stability without the need for clamping operations such as those used in GaussianCube and LGM during the initial phase.

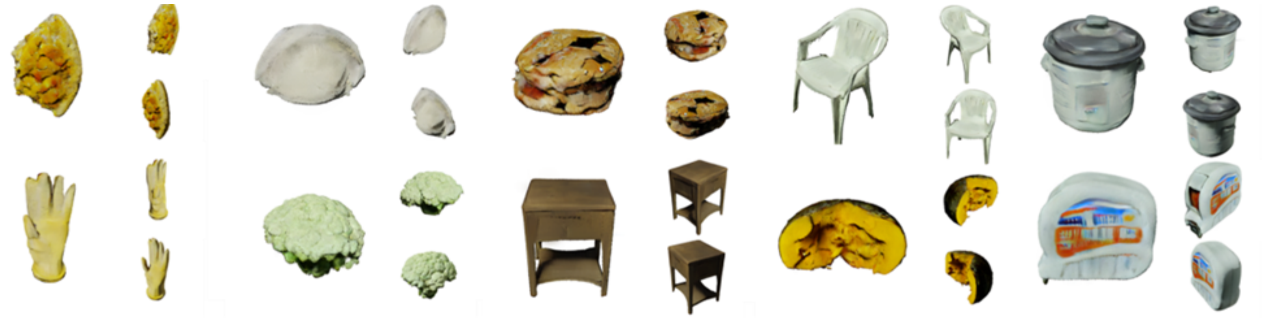
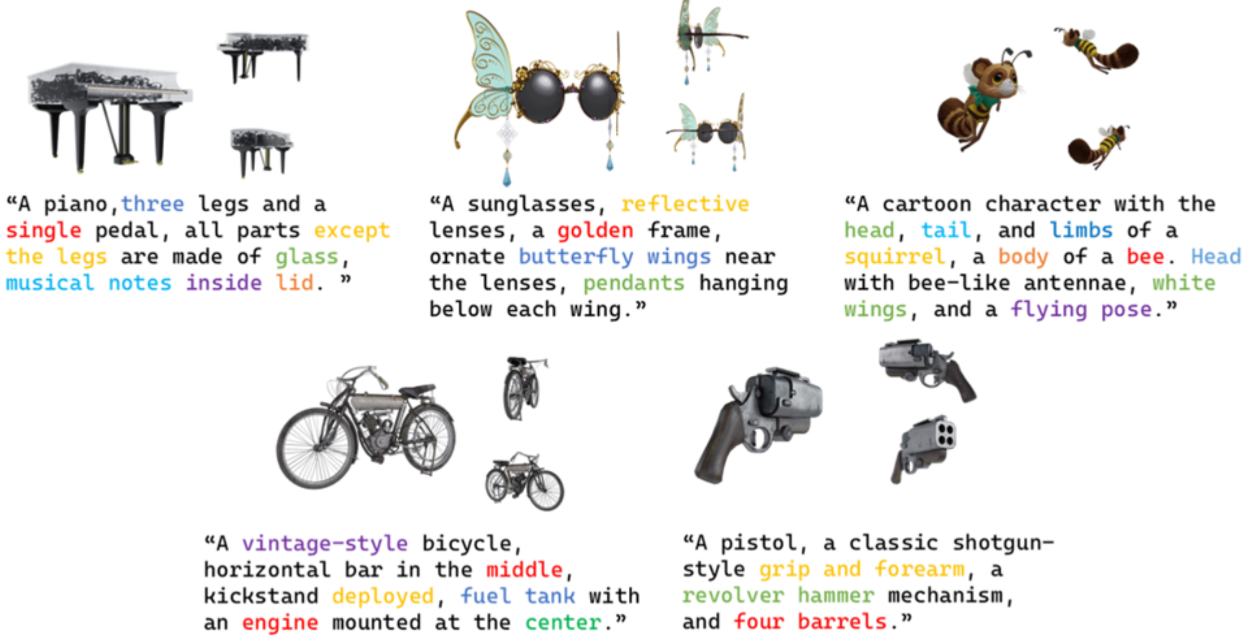


Figure 3. More qualitative results of class-conditioned generation.



“A piano,three legs and a single pedal, all parts except the legs are made of glass, musical notes inside lid. ”

“A sunglasses, reflective lenses, a golden frame, ornate butterfly wings near the lenses, pendants hanging below each wing.”

“A cartoon character with the head, tail, and limbs of a squirrel, a body of a bee. Head with bee-like antennae, white wings, and a flying pose.”

“A vintage-style bicycle, horizontal bar in the middle, kickstand deployed, fuel tank with an engine mounted at the center.”

“A pistol, a classic shotgun-style grip and forearm, a revolver hammer mechanism, and four barrels.”

Figure 4. More qualitative results of text-to-3D generation.

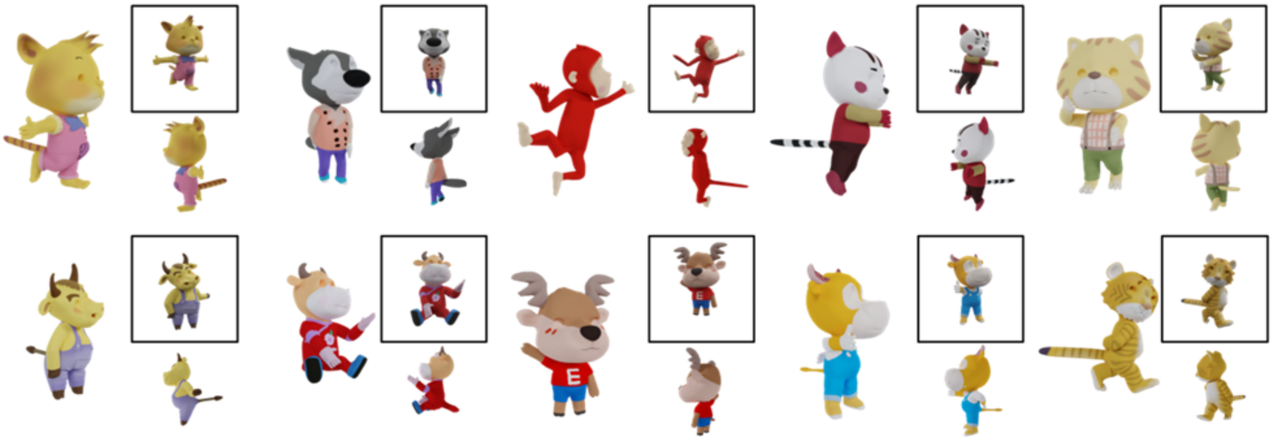


Figure 5. More qualitative results of image-to-3D generation.

3. Additional Visual Results

We illustrate the denoising process of unconditional generation in Fig. 1. To provide a comprehensive view of the process, we render images from different perspectives at various stages of generation. We also provide videos of the generation process in supplementary materials. Fig. 1 demonstrates that our model initially captures a global structure from noise, followed by progressive refinement of Gaussian details across multiple levels to yield the final result, consistent with our progressive generation framework.

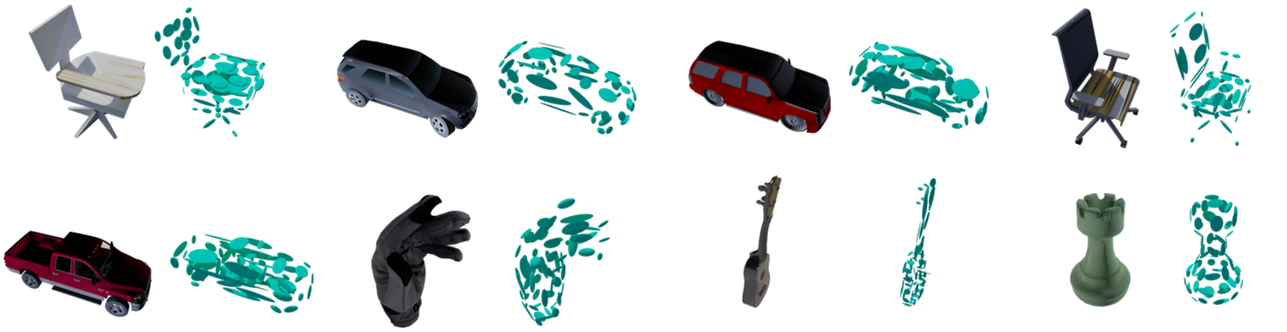


Figure 6. Visualization of single layer part-level latents with generated objects.

We provide additional results for unconditional generation, class-conditioned generation, text-to-3D generation, and image-to-3D generation in Fig. 2, Fig. 3, Fig. 4, and Fig. 5, demonstrating that our model can produce high-quality results. For text-to-3D generation in Fig. 5, our model can generate Gaussian primitives with various styles and details, such as glass textures, as well as combinations of two objects (e.g., a squirrel bee).

For Image-to-3D generation in Fig. 5, Our method demonstrates robust performance in unseen regions, attributed to the incorporation of geometric priors from the mesh during Gaussian primitive initialization. By using part-level latents to describe different levels of detail, our cascaded diffusion model leverages this mixed representation to enhance performance in unseen areas. To further demonstrate the superiority of our HGMM Splatting and cascaded diffusion model, we visualize the generated latents $\{\mathbf{z}^{\mathbf{l=3}}\}$ at the level=3, alongside the corresponding Gaussian primitives in Fig. 6, highlighting our method's ability to capture fine-grained details and showcasing its advantages compared to other 3D generation methods.

4. Broader Impacts

Our method enables the generation of high-quality 3D assets featuring complex geometries and intricate textures, while also offering a high degree of controllability in the generation process. Like all generative models, particular caution is required when dealing with sensitive tasks. Our image-to-3D dataset [6] is meticulously created by human experts, whose contributions have been invaluable in enabling the development of our model. The poses of all generated results are solely intended to demonstrate the generative capability of the model.

It is important to recognize the potential misuse of models trained on this dataset. Risks, such as the generation of misleading or harmful content, apply equally to unconditional, class-conditioned, text-to-3D, and image-to-3D generation. To address these concerns, we emphasize the necessity of robust safeguards and the promotion of ethical practices in the use of our technology and related advancements.

References

[1] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 1

[2] 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*, pages 4456–4465, 2023. 1

[3] 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*, pages 13142–13153, 2023. 1

[4] Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14300–14310, 2023. 1

[5] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022. 2

[6] Zhongjin Luo, Shengcai Cai, Jinguo Dong, Ruibo Ming, Liangdong Qiu, Xiaohang Zhan, and Xiaoguang Han. Rabit: Parametric modeling of 3d biped cartoon characters with a topological-consistent dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12825–12835, 2023. 1, 4

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[8] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2025. 1