A. Implementation Details

3D Gaussian Splatting Details. Instead of directly using the official 3D Gaussian Splatting code provided by Kerbl et al. [6], we reimplement this algorithm by ourselves due to the need to support learnable MLP background. The official 3D Gaussian Splatting implementation propagates the gradients of the Gaussians in an inverse order, i.e., the Gaussians rendered last get gradient first. Our implementation follows a plenoxel [17] style back propagation that calculates the gradient in the rendering order, which we found much easier to incorporate a per-pixel background.

The depth maps are rendered using the view-space depth of the centers of the Gaussians, which we claim is accurate enough due to the tiny scale of the Gaussians [19]. Besides, we implement a z-variance renderer to support z-var loss proposed by [18]. However, we found that z-var loss seems to have a limited impact on the generated 3D asset, mainly due to the sparsity of Gaussians naturally enforcing a relatively thin surface. During rendering and optimizing, we follow the original 3D Gaussian Splatting to clamp the opacity of the Gaussians into $[0.004, 0.99]$ to ensure a stable gradient and prevent potential overflows or underflows.

Guidance Details. All the guidance of 2D image diffusion models we used in this paper is provided by huggingface diffusers [14]. For StableDiffusion guidance, we opt for the *runwayml/stable-diffusion-v1-5* checkpoint for all the experiments conducted in this paper. We also test the performance of GSGEN under other checkpoints, including *stabilityai/stable-diffusion-2-base* and *stabilityai/stable-diffusion-2-1-base*, but no improvements are observed.

For Point-E diffusion model and its checkpoints, we directly adopted their official implementation.

Training Details. All the assets we demonstrate in this paper and the supplemental video are trained on a single NVIDIA 3090 GPU with a batch size of 4 and take about 40 minutes to optimize for one prompt. The 3D assets we showcase in this paper and supplemental video are obtained under the same hyper-parameter setting since we found our parameters robust toward the input prompt. The number of Gaussians after densification is around $[1e^5, 1e^6]$.

Open-Sourced Resources and Corresponding Licenses. We summarize open-sourced code and resources with corresponding licenses used in our experiments in the following table.

We use Stable DreamFusion and threestudio to obtain the results of DreamFusion, Magic3D, and ProlificDreamer under StableDiffusion and on the prompts that are not included in their papers and project pages since the original implementation has not been open-sourced due to the usage of private diffusion models. The results of Fatansia3D are obtained by running their official implementation with their parameter setting for dog-like shapes.

Table 1. Open-sourced resources used in the experiment.

Resource	License
Stable DreamFusion [13]	Apache License 2.0
Fantasia3D [2]	Apache License 2.0
threestudio [5]	Apache License 2.0
StableDiffusion [11]	MIT License
DeepFloyd IF [1]	DeepFloyd IF License
HuggingFace Diffusers	Apache License 2.0
OpenAI Point-E [9]	MIT License
ULIP [15, 16]	BSD 3-Clause License

B. Additional Results

B.1. User-Guided Generation

Initialization is straightforward for 3D Gaussian Splatting due to its explicit nature, thereby automatically supporting user-guided generation. We evaluate the proposed GSGEN on user-guided generation with shapes provided in Latent-NeRF [8]. In this experiment, the initial points are generated by uniformly sampling points on the mesh surface. To better preserve the user's desired shape, we opt for a relatively small learning rate for positions. We compare the 3D content generated by GSGEN with that generated by the state-of-the-art user-guided generation methods, Latent-NeRF [8] and Fantasia3D [2], as shown in Fig. 1. Our proposed GSGEN achieves the best results among all alternatives in both geometry and textures and mostly keeps the geometrical prior given by the users.

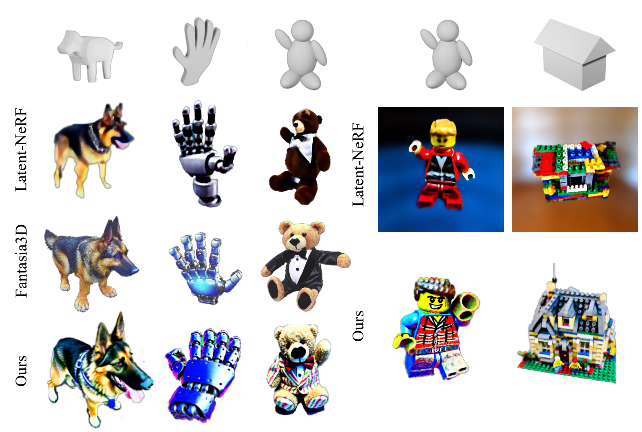


Figure 1. Qualitative comparison results on user-guided generation. The prompts from left to right are *(1) A German Shepherd; (2) A robot hand, realistic; (3) A teddy bear in a tuxedo; (4) a lego man; (5) a house made of lego*.

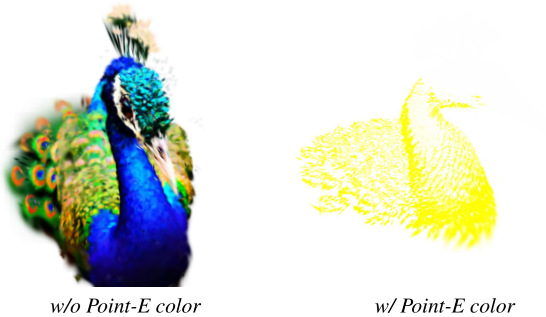


Figure 2. The impact of adopting Point-E generated color.

Figure 3. Comparison between Point-E generated point clouds and GSGEN generated 3D assets.

B.2. More Text-To-3D Results

We present more general text-to-3D generation results of GSGEN in Fig. 8 and Fig. 9. Our approach can generate 3D assets with accurate geometry and improved fidelity.

For more delicate assets generated with GSGEN and the corresponding videos, please watch our supplemental video.

B.3. More Qualitative Comparisons

In addition to the qualitative comparison in the main text, we provide more comparisons with DreamFusion [10] in Fig. 10 and Fig. 11, Magic3D [7] in Fig. 12, Fantasia3D [2] and LatentNeRF [8] in Fig. 13. In order to make a fair comparison, the images of these methods are directly copied from their papers or project pages. Video comparisons are presented in the supplemental video.

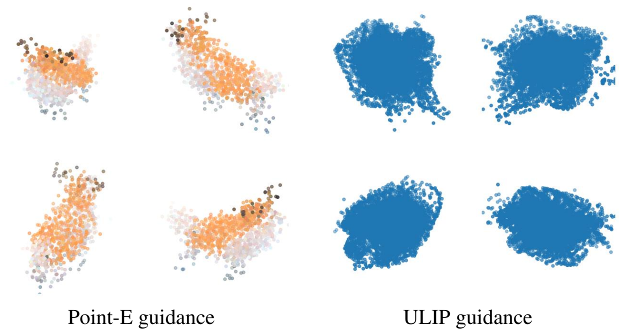


Figure 4. Point clouds optimized under *Point-E* and *ULIP*. Prompt: *A corgi*

B.4. More Experiments

B.4.1 Color Initialization

As illustrated in the main text, GSGEN adopts random color initialization instead of directly applying Point-E generated colors. Fig. 2 demonstrates the detrimental effect of direct utilization of Point-E generated texture.

B.4.2 3D Point Cloud Guidance

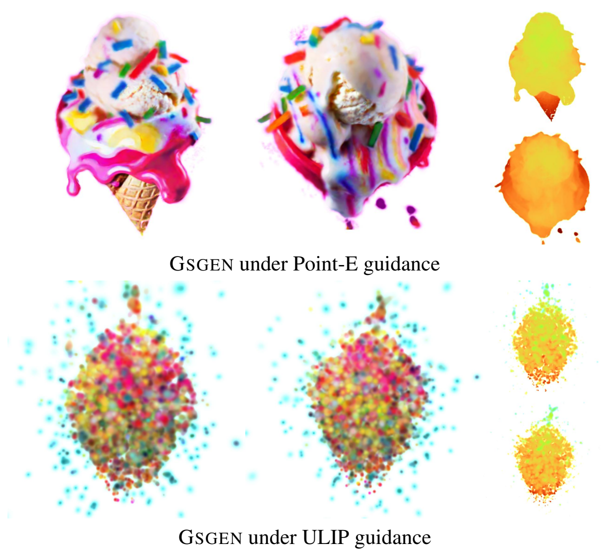
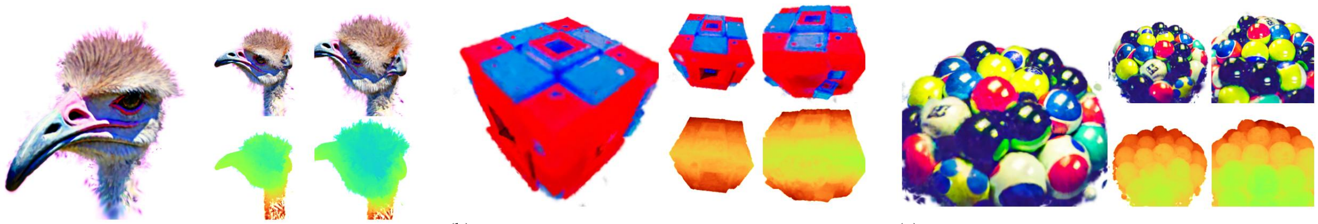


Figure 5. Text-to-3D generation qualitative comparison with 3D prior as Point-E or ULIP. Prompt: *A DSLR photo of an ice cream sundae.*

Our empirical results demonstrate that GSGEN consistently delivers high performance, even in scenarios where Point-E operates sub-optimally. As illustrated in Fig. 3, we showcase point clouds generated by Point-E alongside the corresponding 3D assets created by GSGEN. Our approach demonstrates great performance when Point-E provides only



(a) *A high quality photo of an ostrich*

(b) *A small red cube is sitting on top of a large blue cube. red on top, blue on bottom*

(c) *a zoomed out DSLR photo of a few pool balls sitting on a pool table*

Figure 6. Several typical failure cases of GSGEN.

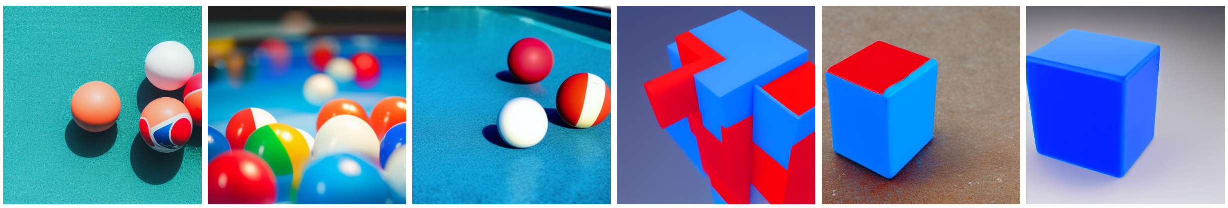


Figure 7. Prompts that StableDiffusion cannot correctly process, which leads to the failure of corresponding text-to-3D generation.

rough guidance. We attribute this to direct 3D prior provided by Point-E assisting in geometrical consistency by correcting major shape deviations in the early stage, without the need to guide fine-grained geometric details.

Except for the Point-E [9] used in our proposed GSGEN, we also test a CLIP-like point cloud understanding model ULIP [15, 16]. While achieving superior performance in zero-shot point cloud classification, ULIP seems ineffective in the context of generation. Fig. 4 demonstrates point clouds generated under the guidance of ULIP and Point-E. Point-E can guide the point cloud to a consistent rough shape with SDS loss while the inner-product similarity provided by ULIP leads to a mess. We substitute the 3D prior in GSGEN from Point-E to ULIP in Fig. 5, yielding similar results to point cloud optimization.

B.4.3 2D Image Guidance

Except for StableDiffusion, we also test the performance of GSGEN under the guidance of *DeepFloyd IF*, another open-sourced cutting-edge text-to-image diffusion model. Compared to StableDiffusion, DeepFloyd IF has an Imagen-like architecture and a much more powerful text encoder. We demonstrate the qualitative comparison between GSGEN under different guidance in Fig. 14. Obviously, assets generated with *DeepFloyd IF* have a much better text-3D alignment, which is primarily attributed to the stronger text understanding provided by T-5 encoder than that of CLIP text encoder. However, due to the modular cascaded design, the input to *DeepFloyd IF* has to be downsampled to 64×64, which may result in a blurry appearance compared to those generated under StableDiffusion.

Our concurrent work MVDream [12] proposes to fine-tune StableDiffusion with 3D aware components on Objaverse [3, 4] to enhance multi-view consistency and alleviate the Janus problem. We also test the performance of GSGEN under MVDream. As shown in Fig. 15, MVDream significantly contributes to multi-view consistency, resulting in more accurate geometry and complete 3D assets (such as more complete panda and Janus-free ostrich). Although alleviating the Janus problem, we empirically find that MVDream demonstrates sub-optimal performance towards complex prompts, as shown in Fig. 16. 3D assets generated with MVDream tend to ignore some parts of the prompt compared to those under StableDiffusion guidance, e.g. the moss on the suit, the vines on the car, and the chicken and waffles on the plate. This demonstrates that introducing 3D prior while retaining the information from the original diffusion model presents a challenging problem, and we consider this issue as our future research.

C. Failure Cases

Despite the introduction of direct 3D prior, we could not completely eliminate the Janus problem, attributed to the ill-posed nature of text-to-3D generation through a 2D prior and the limited capability of the 3D prior we employed.

Fig. 6 showcases several typical failure cases we encountered in our experiments. In Fig. 6a, the geometrical structure is correctly established, but the Janus problem happens on the appearance (another ostrich head on the back head). Fig. 6b and Fig. 6c demonstrates another failure case caused by the limited language understanding of the guidance model. StableDiffusion also fails to generate reasonable images with these prompts, as illustrated in Fig. 7.

References

[1] Alex, Misha Konstantinov, apolinário, Daria Bakshandaeva, Ksenia Ivanova, Sayak Paul, Will Berman, and Emad. deepfloyd/if, 2023. 1

[2] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 1, 2

[3] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects. *CoRR*, abs/2307.05663, 2023. 3

[4] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 13142–13153. IEEE, 2023. 3

[5] Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. threestudio: A unified framework for 3d content generation. https://github.com/threestudio-project/threestudio, 2023. 1

[6] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023. 1

[7] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[8] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. *arXiv preprint arXiv:2211.07600*, 2022. 1, 2

[9] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *CoRR*, abs/2212.08751, 2022. 1, 3

[10] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. 2

[11] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE, 2022. 1

[12] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv:2308.16512*, 2023. 3, 12

[13] Jiaxiang Tang. Stable-dreamfusion: Text-to-3d with stable-diffusion, 2022. https://github.com/ashawkey/stable-dreamfusion. 1

[14] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022. 1

[15] Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning unified representation of language, image and point cloud for 3d understanding. *arXiv preprint arXiv:2212.05171*, 2022. 1, 3

[16] Le Xue, Ning Yu, Shu Zhang, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip-2: Towards scalable multi-modal pre-training for 3d understanding, 2023. 1, 3

[17] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2021. 1

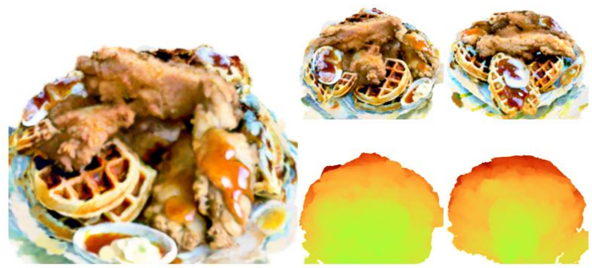
[18] Junzhe Zhu and Peiye Zhuang. Hifa: High-fidelity text-to-3d with advanced diffusion guidance. *CoRR*, abs/2305.18766, 2023. 1

[19] Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus H. Gross. EWA volume splatting. In *12th IEEE Visualization Conference, IEEE Vis 2001, San Diego, CA, USA, October 24-26, 2001, Proceedings*, pages 29–36. IEEE Computer Society, 2001. 1

A bunch of blue rose, highly detailed

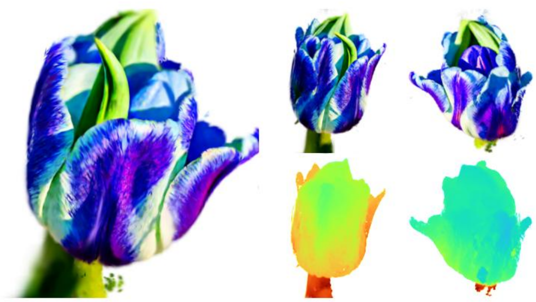
A high quality photo of a dragon

A plush dragon toy



A zoomed out DSLR photo of a plate of fried chicken and waffles with maple syrup on them

A beautiful dress made of feathers, on a mannequin



A high quality photo of a blue tulip

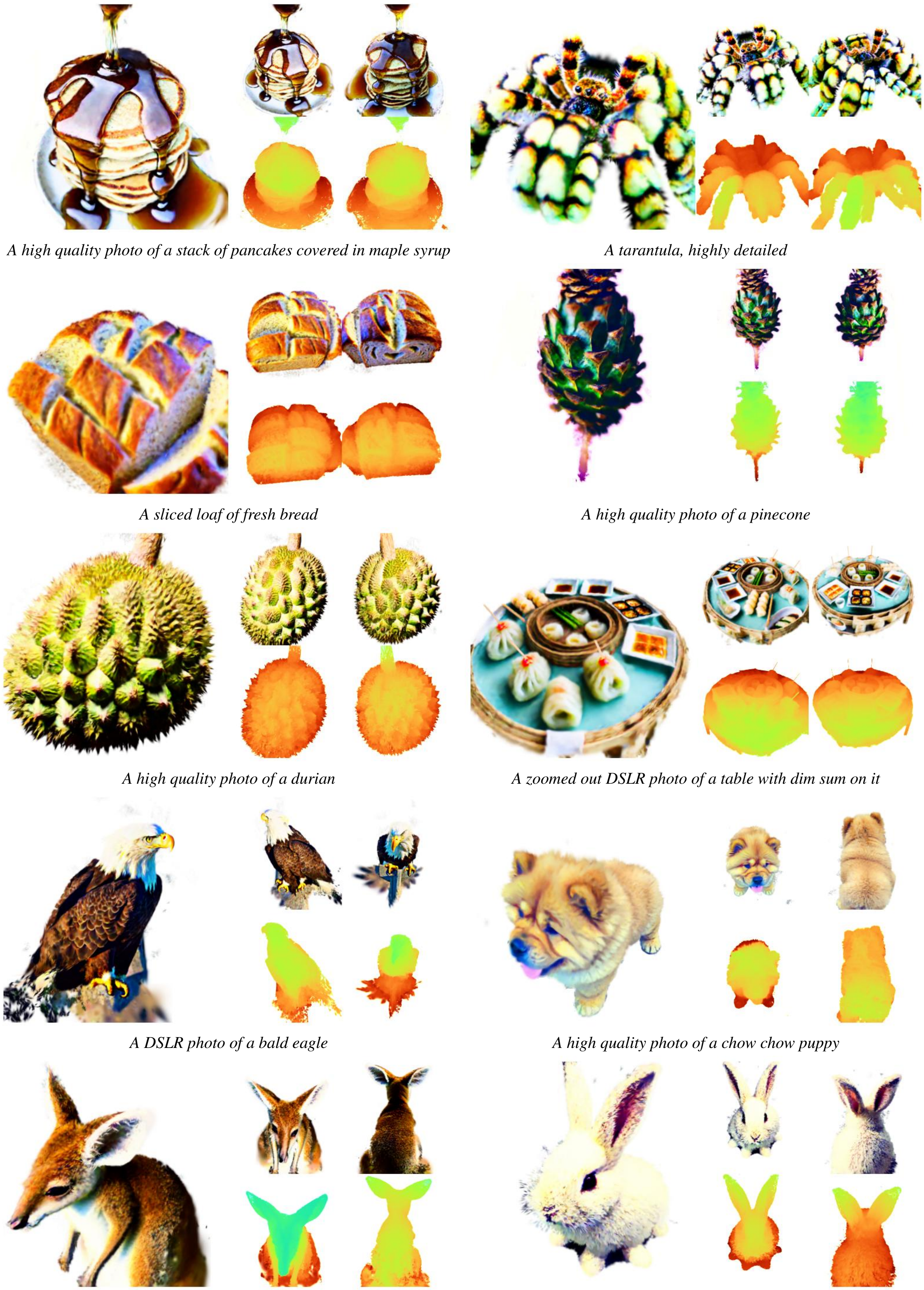
A DSLR photo of a plush triceratops toy, studio lighting, hight resolution

A DSLR photo of a tray of Sushi containing pugs

A DSLR photo of an origami motorcycle

A DSLR photo of a pineapple

Figure 8. More 3D assets generated with GSGEN.



A high quality photo of a stack of pancakes covered in maple syrup

A tarantula, highly detailed

A sliced loaf of fresh bread

A high quality photo of a pinecone

A high quality photo of a durian

A zoomed out DSLR photo of a table with dim sum on it

A DSLR photo of a bald eagle

A high quality photo of a chow chow puppy

A high quality photo of a kangaroo

A high quality photo of a furry rabbit

Figure 9. More 3D assets generated with GSGEN.

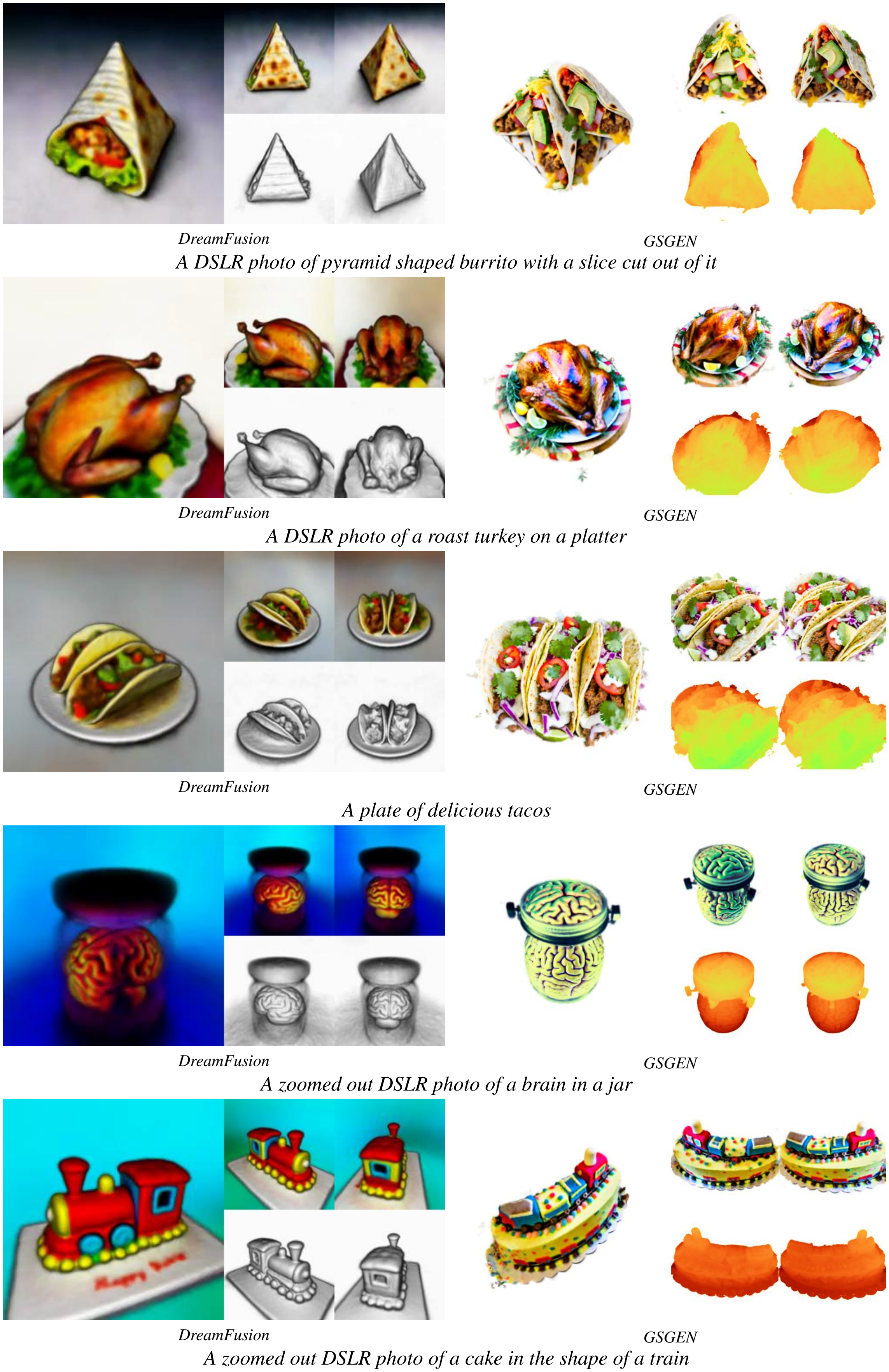


Figure 10. More comparison results with DreamFusion.

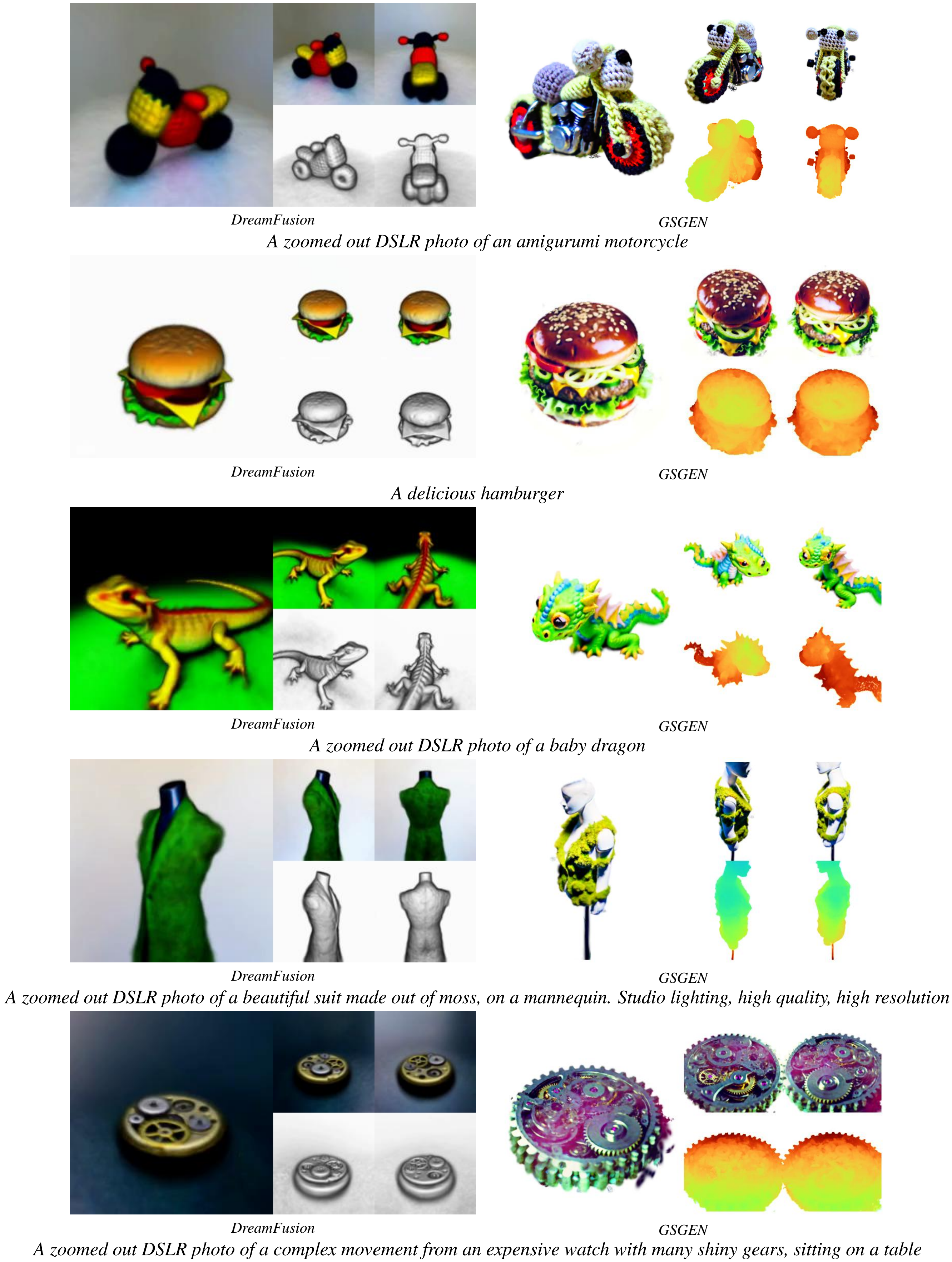
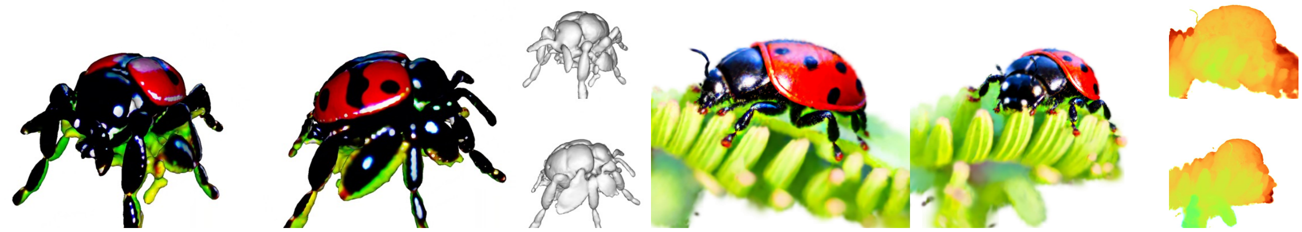
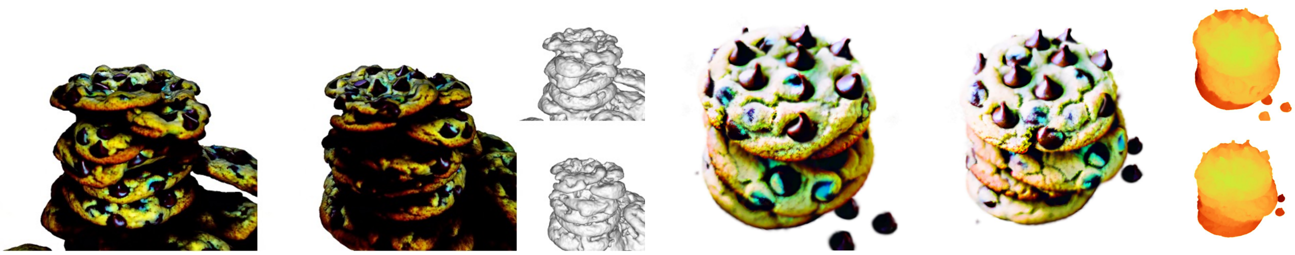


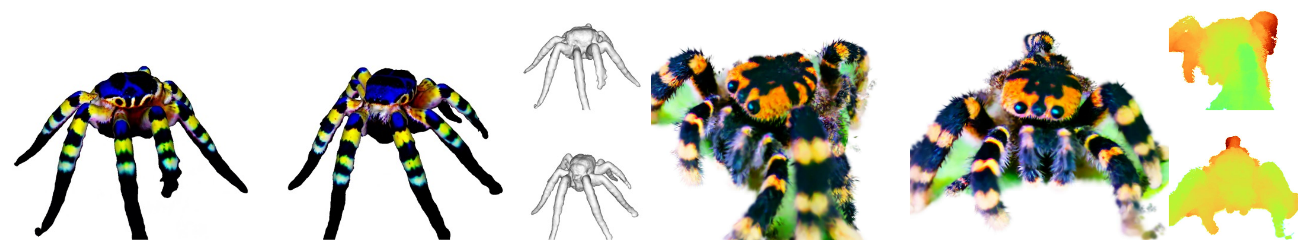
Figure 11. More comparison results with DreamFusion.



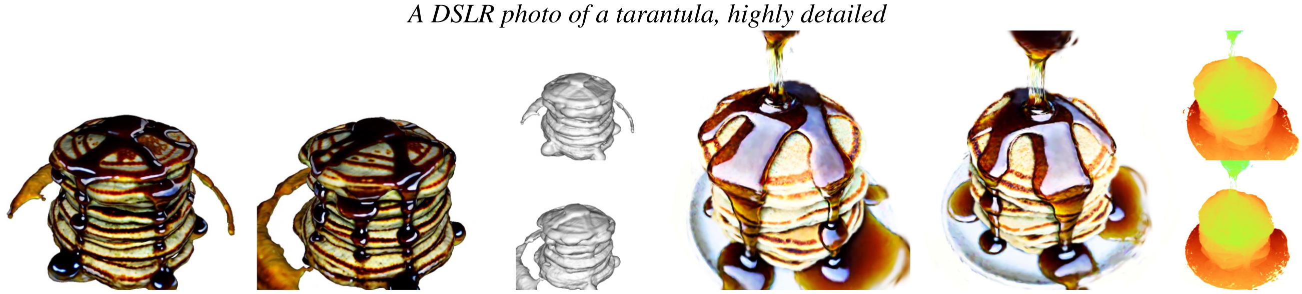
A zoomed out DSLR photo of a ladybug



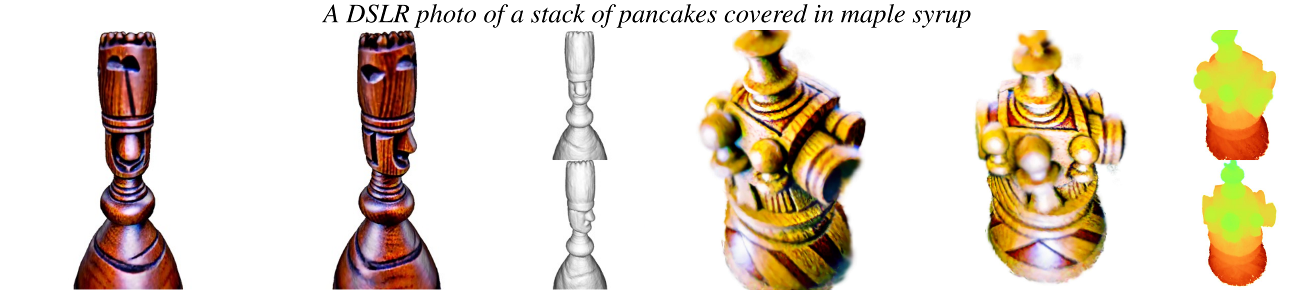
A zoomed out DSLR photo of a plate piled high with chocolate chip cookies



A DSLR photo of a tarantula, highly detailed



A DSLR photo of a stack of pancakes covered in maple syrup



A zoomed out DSLR photo of a beautifully carved wooden knight chess piece

Figure 12. More comparison results with Magic3D.

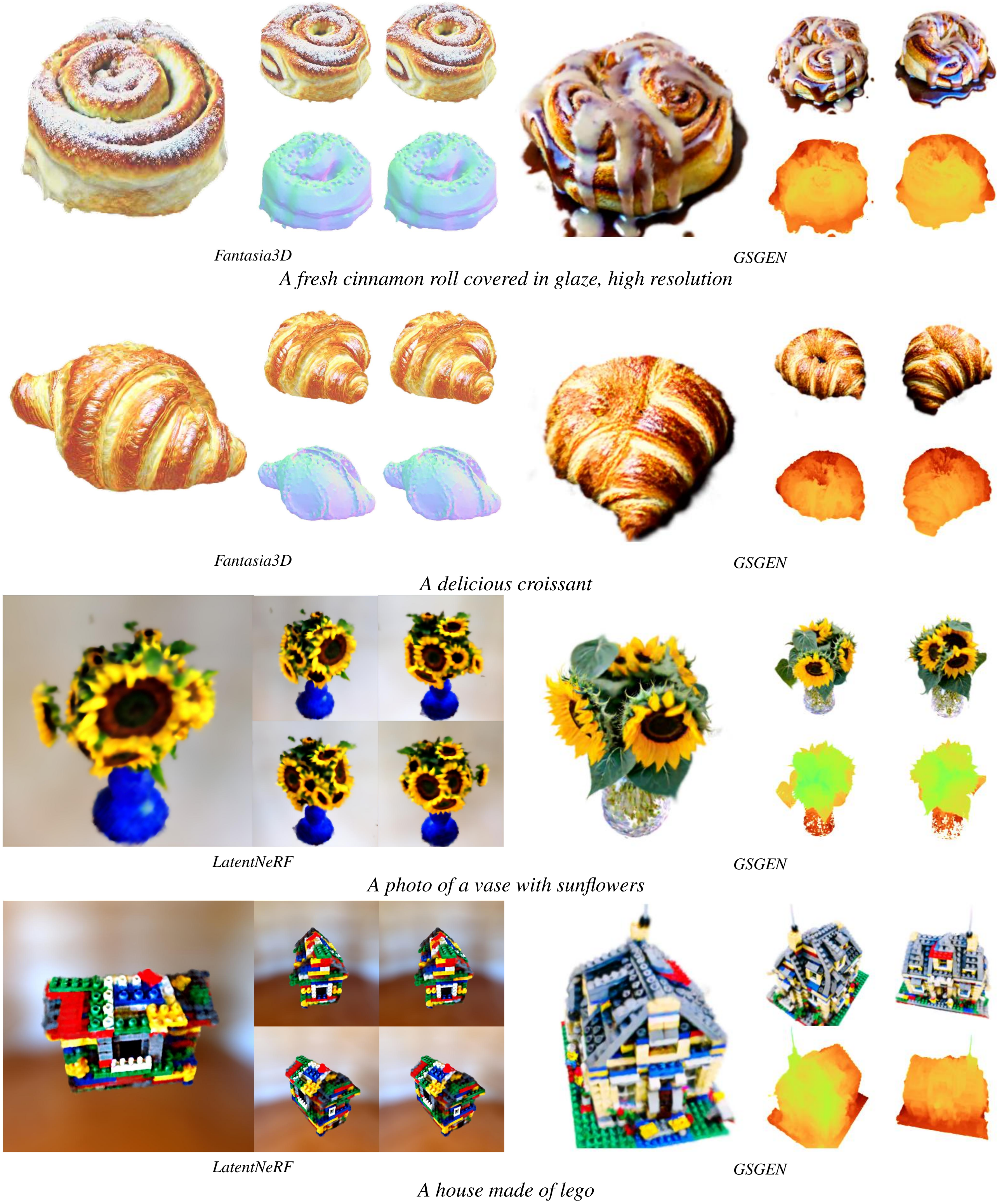


Figure 13. More comparison results with LatentNeRF and Fantasia3D.

Figure 14. Qualitative comparison of GSGEN under StableDiffusion guidance and DeepFloyd IF guidance.

Figure 15. 3D assets generated under the guidance of our concurrent work MVDream [12]. MVDream helps generate more accurate geometry and alleviate the Janus problem, e.g. more complete panda and suit, and the Janus-free ostrich.

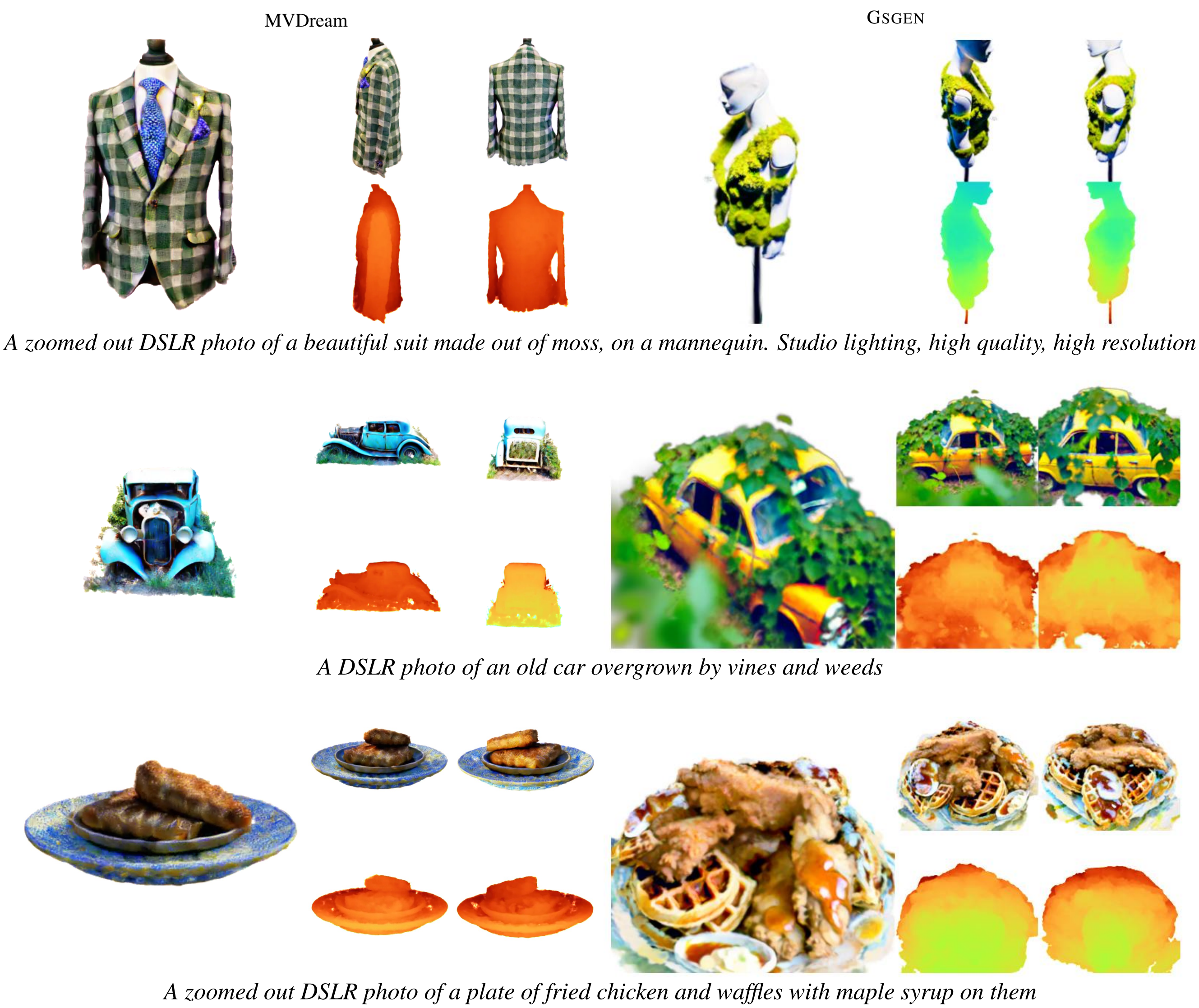


Figure 16. Qualitative comparison of MVDream and GSGEN with StableDiffusion on complex prompts.