VQ3D: Learning a 3D-Aware Generative Model on ImageNet

SUPPLEMENTARY MATERIALS

Kyle Sargent
Stanford University

Jing Yu Koh
Carnegie Mellon University

Han Zhang
Google Research

Huiwen Chang
OpenAI

Charles Herrmann
Google Research

Pratul Srinivasan
Google Research

Jiajun Wu
Stanford University

Deqing Sun
Google Research

Thanks for checking the supplementary materials. in which we provide additional details for the ease of replicating the results of our method. For video results, **we encourage the reader to consult the project webpage**.

1. Implementation details

1.1. NeRF Model

We train and evaluate all models at 256x256 resolution, except pi-GAN [3] which we train and evaluate at 128x128 following [4].

We use a constant 49.13 degree field of view and pinhole camera model. We use a camera radius of 2.732 following [10] and a canonical pose at $(-2.732, 0, 0)$. All views canonical and novel are looking at $(0, 0, 0)$ and have a constant camera up vector of $(0, 0, 1)$. We sample novel view camera locations uniformly in a disc in the YZ-plane centered at the canonical pose with radius $.4$. We use a near plane of .7 and far plane of $1e6$. We find that using the slightly large near plane of .7 was necessary in order to avoid a failure mode where all the content was clustered very close to the camera leading to poor novel views; we hope to eliminate this failure mode in future work.

We perform volume rendering at the full 256x256 resolution using the importance sampling scheme of [1]. We have a separate proposal and NeRF MLP and render in two stages, the first stage using the proposal MLP to evaluate a wide range of sample locations, and the second stage using the NeRF MLP queried at locations determined by importance sampling of the weights and locations from the first stage. During training, we add a stop-grad between the proposal and NeRF MLP like [1] and supervise the Proposal MLP with the interlevel loss. Our NeRF MLP is not view dependent and the only input it receives is triplane features which are determined by looking up the contracted 3D points of the sample locations. We apply a fixed orthonormal transformation to all points before triplane lookup because our canonical pose is axis-aligned, so we desire that our triplanes are not axis-aligned to avoid artifacts.

We evaluate 32 samples along each ray for each sampling stage. Thus, rendering a full 256x256 RGB image takes 256x256x64 triplane lookups and MLP evaluations. We use the same number of ray samples, 32, for training, FID evaluation, and rendering videos.

1.2. Setup and hyperparameters

We train with the Adam optimizer [9] with $\beta_1 = .9, \beta_2 = .99$, and cosine learning rate schedule with 50K warmup steps, similar to [15], with an initial autoencoder LR of 0 and max LR of 1e-4. We use codebook size 8192 and l_2-normalized, factorized codebook with embedding dimension 8.

Different from [15], we do not use weight decay, and our discriminator LR is scaled down from the autoencoder LR by .5 so that the discriminator does not overpower the autoencoder, which was an issue especially in early training.

Due to the many losses in our Stage 1 training, we outline their weights in Table 1 and reference the original implementation if they are not losses designed by us.

1.3. Discriminators

We use StyleGAN [8] discriminators for both the main and novel view discriminator. They are identical except that the novel view discriminator accepts 4-channel RGBD images, and the main view discriminator accepts 3-channel RGB images.

1.4. Timing and throughput

We train our main model on ImageNet for 1000K steps in Stage 1, and 340K steps in Stage 2. Stage 1 training takes 16 days and Stage 2 training takes 3 days on 64 CloudTPUv4. We note that good performance (<25 FID) can be achieved

Loss	Weight
l_2 [15]	1
Perceptual [15]	1e-1
Logit-laplace [15]	1e-1
Discriminator [15]	1e-1
Novel discriminator	1e-1
Quantizer [15]	1
Weighted pointwise depth (λ_{depth})	1e1
Negative depth scale penalty (λ_{s1})	1
Large depth scale penalty (λ_{s2})	1e-3
Interlevel [1]	1
Distortion [1]	2.5e-1

Table 1: Weights of various losses used in Stage 1 training of our autoencoder.

with a fraction of this training time (around 200K steps for both stages), but we train as long as possible to achieve the best results. For inference on a single V100, our Stage 1 model renders 8.7 img/s. We train with a Stage 1 batch size of 128 and Stage 2 batch size of 512. For each batch in Stage 1, we render 256 images; 128 to reconstruct the full batch at the canonical view, and an additional 128 novel views to be critiqued by the novel view discriminator. Though this is expensive, our volume rendering stage is made cheaper even than [4] by using 32 instead of 64 hidden units for the feature MLPs and using 32 instead of 48 samples per ray. We leverage gradient accumulation in Stage 2 training in order to train with 512 batch size.

1.5. Details about MVS-1

Due to the simplicity of CompCars and the complexity of ImageNet, it is desirable to study a dataset of intermediate complexity to better understand the shortcomings of GAN-based methods. We thus introduce a new dataset, Multiview ShapeNet-1, to serve as this intermediate.

Multiview ShapeNet was introduced in [13] to study the problem of novel view synthesis from single images. We synthesize a version of this dataset called Multiview Shapenet-1. Different from the original dataset, we have only one view per synthesized scene (which prevents novel view supervision as used in [13]), and we have only one salient object per scene. We synthesize the dataset with 360-degree views of a random salient object sampled from the 55 ShapeNet [5] object categories rendered against a random HDR background. During dataset synthesis, in addition to sampling camera poses with random elevation and azimuth, we also randomly sample the camera field of view and adjust the camera radius from the center of the scene accordingly so that the salient object is generally within the picture frame. Examples synthesized from this dataset are shown in Figure 1.

top-k	1000	2000	3000	4000	8192	8192	8192
top-p	1.00	1.00	1.00	1.00	1.00	0.98	0.95
FID ↓	31.5	33.2	34.1	34.7	35.4	32.2	35.7

Table 2: FID scores on ImageNet from sampling over top-k and top-p values. The size of our full codebook is 8192.

We will make this dataset available upon acceptance.

1.6. Evaluation

As is standard [15], we compute Stage 1 metrics (reconstruction) over the ImageNet validation set and Stage 2 metrics (generation) over real samples from the train set and generated samples. We use 50K samples to evaluate FID for all methods. We sample views for Stage 2 FID computation uniformly in a disc of radius .2 tangent to the sphere at the canonical pose.

We use the Depth Accuracy metric used in [14, 4], but differently we don't mask out any invalid regions because our monocular depth estimator DPT [12] predicts a dense depth map over the input and every pixel is assumed to be valid. We also use inverse depth instead of depth because we model much larger scenes than either [14] or [4].

We experiment with classifier guidance but find it gives only a small performance boost, and so investigating model improvements was more worthwhile to improve the FID than tweaking classifier guidance settings.

2. Additional experiments

Sampling We analyze the performance of VQ3D with top-p and top-k sampling in Table 2, as VQ-GAN [6] noted these sampling changes can give significant performance improvements analogous to truncation sampling for GANs [2]. For VQ3D, a top-k of 1000 and top-p of 1.0 give the best results.

Additional tuning of baselines. Although the strongest baselines, EG3D [4] and StyleNeRF [7], perform poorly on ImageNet, they may need to be tuned to perform well on this new dataset. To verify that the limitation of the baseline methods is fundamental, we extensively tune both on Imagenet for a range of hyperparameters in Tables 3 and 4. We see the baselines do not achieve good performance for a range of hyperparameter settings. Additionally, we observe that EG3D has significant inter-run variance in terms of FID on ImageNet, even when rerunning the same configuration, which may indicate instability for large datasets such as ImageNet. When running the same config multiple times, we report the best value achieved among all runs.

Depth loss ablation. We compare our proposed pointwise depth loss against a loss on accumulated depth in the table below. Without $\mathcal{L}_{pointwise}$, FID and especially Depth

EG3D Tuning	Sweep	ImageNet FID
R1 gamma	{.3, .6}	{**82**, 99}
Density reg.	{.125, .25, .5}	{91, **82**, 96}
Disc. LR (1e-3)	{.5, 1, 2, 4}	{122, **82**, 116, 113}
Gen. LR (1e-3)	{.625, 1.25, 2.5, 5}	{111, **82**, 106, 136}

Table 3: Hyperparameter tuning of EG3D on ImageNet.

StyleNeRF Tuning	Sweep	ImageNet FID
R1 gamma	{.15, .3, .6}	{75, **73**, 74}
Disc. LR (1e-3)	{.625, 1.25, 2.5, 5}	{96, 87, 73, **69**}
Gen. LR (1e-3)	{.625, 1.25, 2.5, 5}	{78, 74, **73**, 107}

Table 4: Hyperparameter tuning of StyleNeRF on ImageNet.

Accuracy (DA) quickly degrade as the novel view radius increases. Such models do not infer realistic sharp surfaces, but instead infer densities along camera rays which only sum to plausible depths near the canonical viewpoint.

	$r = 0.0$		$r = 0.2$		$r = 0.4$	
Generation	FID	DA↓	FID	DA↓	FID	DA↓
$\mathcal{L}_{accum}$	33.0	0.24	37.3	0.42	49.7	0.64
$\mathcal{L}_{pointwise}$ (ours)	**32.5**	**0.16**	**35.4**	**0.16**	**41.1**	**0.18**

Table 5: Depth loss ablation

These metrics are comparable to the ablations in Tables 2 & 3; we obtain our SOTA FID of 16.8 (Table 1) via the longest possible training and most optimal sampling config.

Normals evaluation. For fairer geometry evaluation, we evaluate the predicted normals via a version of Normal Consistency (NC) from MonoSDF adapted to our setting, shown in the table below. VQ3D has more accurate normals than the baselines, although the gap is less pronounced than for Depth Accuracy (c.f. Table 1).

Depth accuracy and StyleNeRF. We provide additional details about depth accuracy for StyleNeRF. We designed a depth loss described in the main paper which improved the depth accuracy of EG3D, pi-GAN, and GIRAFFE without compromising FID. However, we were unable to improve the depth accuracy of StyleNeRF with depth losses. To shed some light on this issue, we analyze the depth accuracy of StyleNeRF models over a wide range of hyperparameter settings in Table 7. Depth accuracies > 1.90 indicate depth maps have collapsed to a flat plane. We see that the learning of geometry for StyleNeRF is unstable. On ImageNet, geometry is not learned for most settings. Adding a depth loss does not improve geometry, although 3 hyperparameter settings (doubled batch size, slightly reduced discriminator LR, increased generator LR) improve depth accuracy somewhat (to 1.64, 1.70, 1.82, respectively.) We note these depth accuracies are not close to either EG3D with depth loss (.88) or our model (.13). In general, whether or not StyleNeRF learns geometry is highly sensitive to individual hyperparameter settings.

Generation	piGAN	GIRAFFE	EG3D	StyleNeRF	VQ3D
NC↑	0.25	0.40	0.28	0.49	**0.63**

Table 6: Evaluation of normals.

StyleNeRF Tuning	Sweep	Depth Accuracy ↓
Depth loss weight	{0, .5}	{**1.96**, 2.00 }
Batch size	{64, 128}	{1.96, **1.64** }
R1 gamma	{.15, .3, .6}	{1.97, 1.96, **1.95**}
Disc. LR (1e-3)	{.625, 1.25, 2.5, 5}	{1.95, **1.70**, 1.96, 1.96}
Gen. LR (1e-3)	{.625, 1.25, 2.5, 5}	{1.97, 1.96, 1.96, **1.82**}

Table 7: Depth accuracy for StyleNeRF.

Note on EG3D performance. Our EG3D FID was 82.2 but it is not directly comparable to superior FIDs reported by concurrent work. The 3DGP authors train EG3D with their proposed dataset filtering (removing around 2/3 of the ImageNet training set). IVID's EG3D FID is 40.4, but for an easier task (128 resolution generation instead of 256). We retrained EG3D to more closely match the IVID settings, in particular the lower resolution, and obtain an FID of 51.4, closing the majority (74%) of the gap. IVID further used classes in lieu of poses for generator and discriminator pose-conditioning, which might further close the gap.

3. Additional samples

We show additional uncurated generated samples with geometry in Figure 3 and Figure 4.

References

[1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 1, 2

[2] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 2

[3] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *arXiv*, 2020. 1

[4] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021. 1, 2

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2

[6] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis, 2020. 2

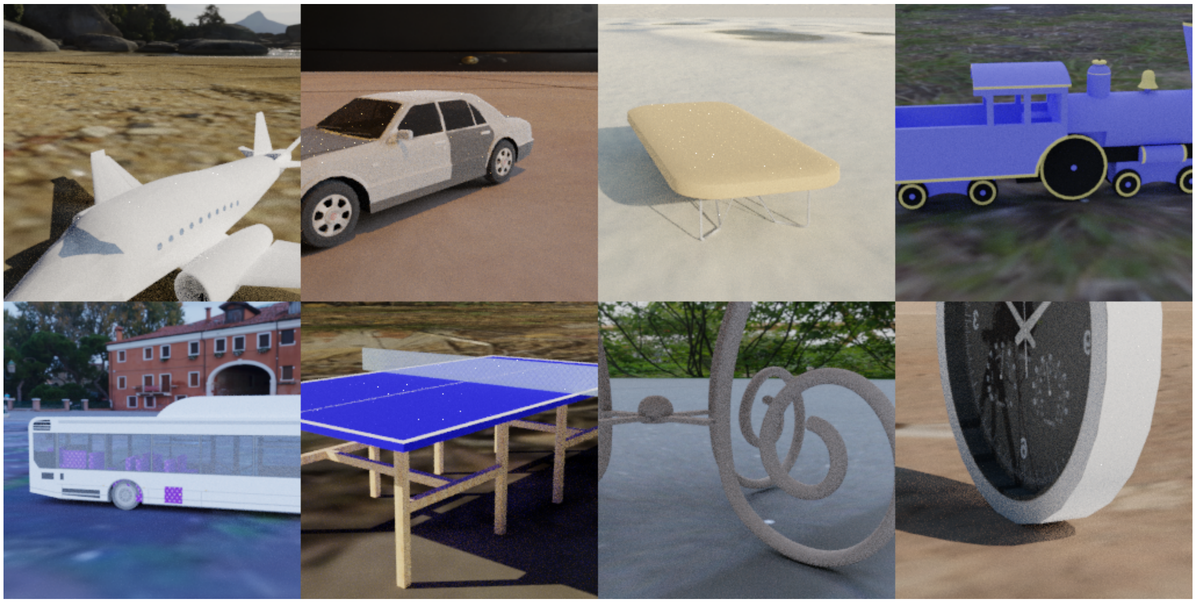

Figure 1: Samples from our synthesized MVS-1 dataset.

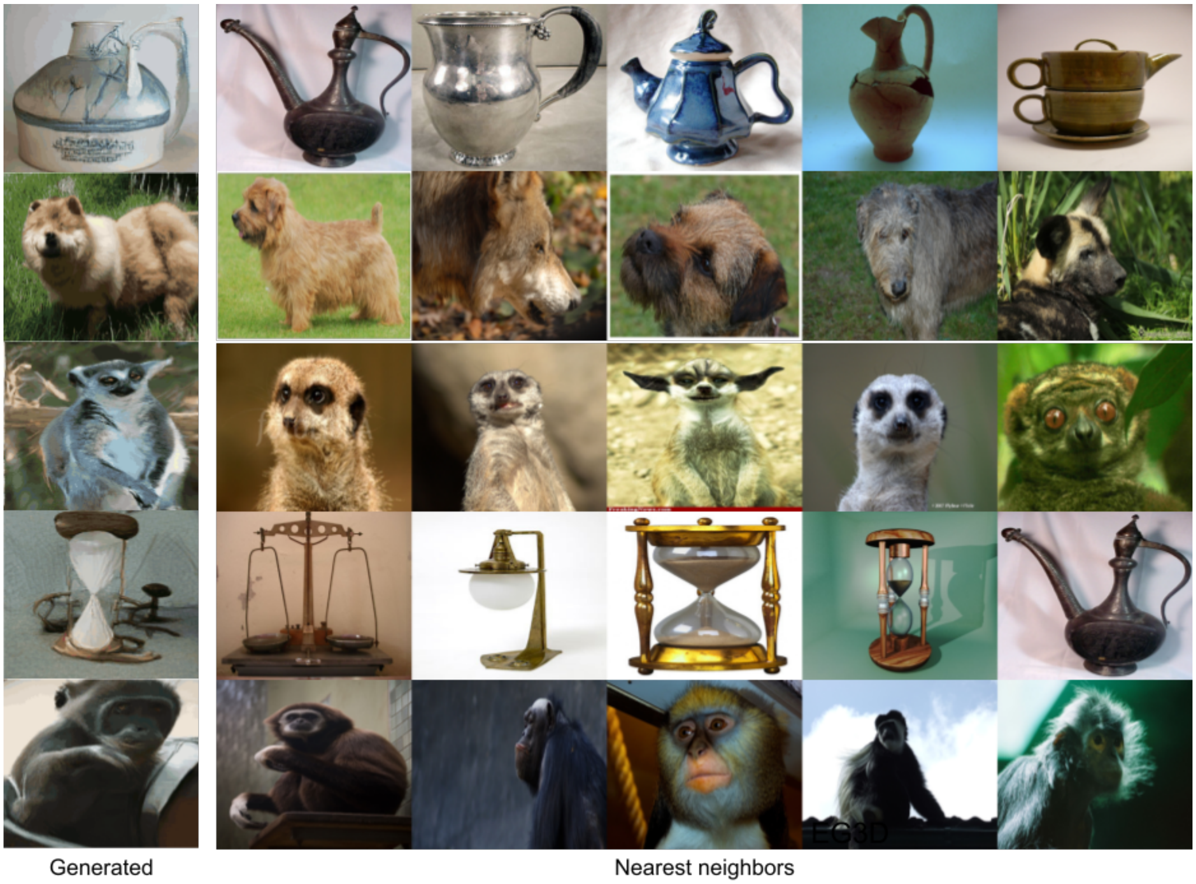

Figure 2: Nearest neighbors in the ImageNet training set for generated examples from our model, computed via CLIP [11] similarity.

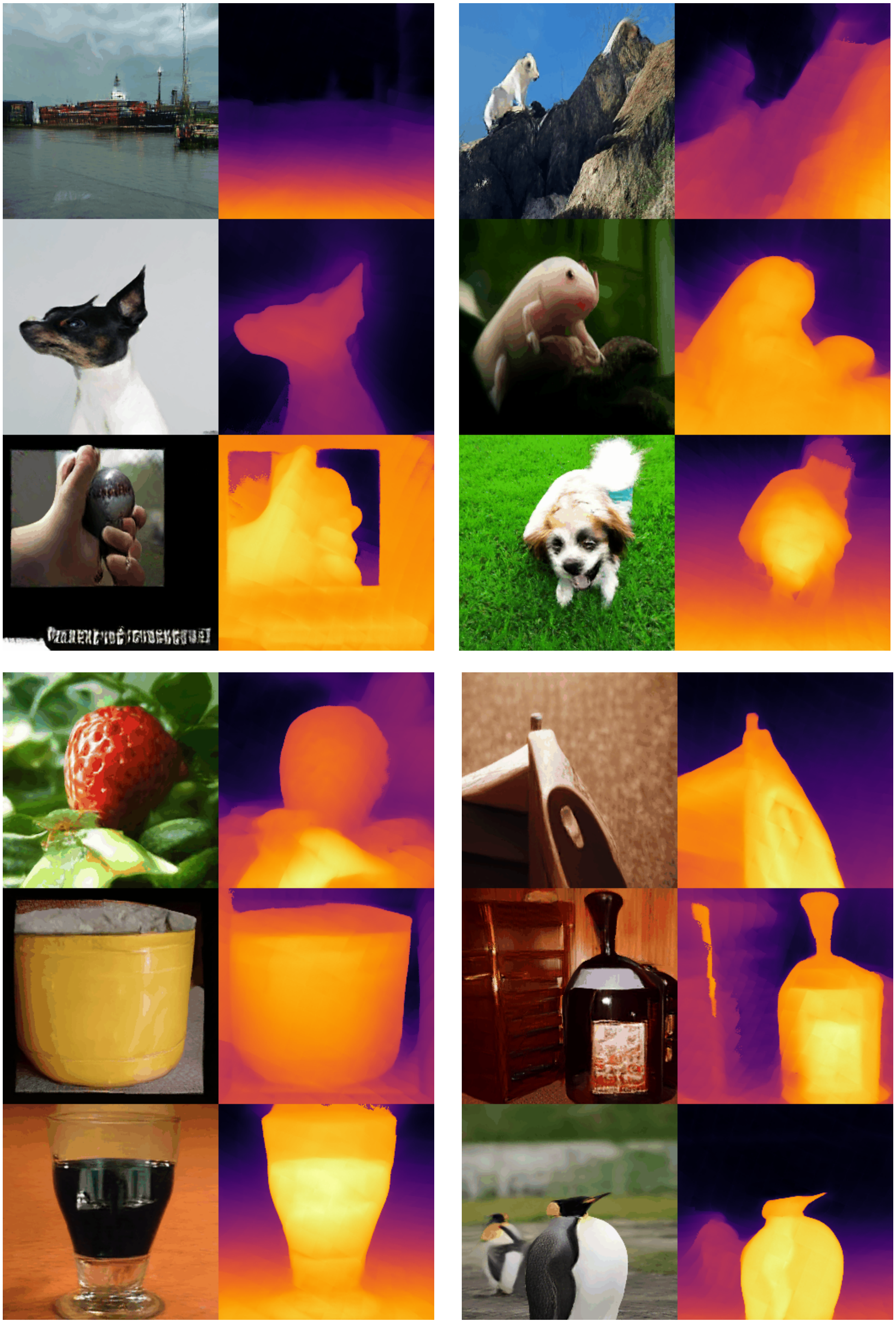

Figure 3: Uncurated fully generated samples from our Stage 2 model.

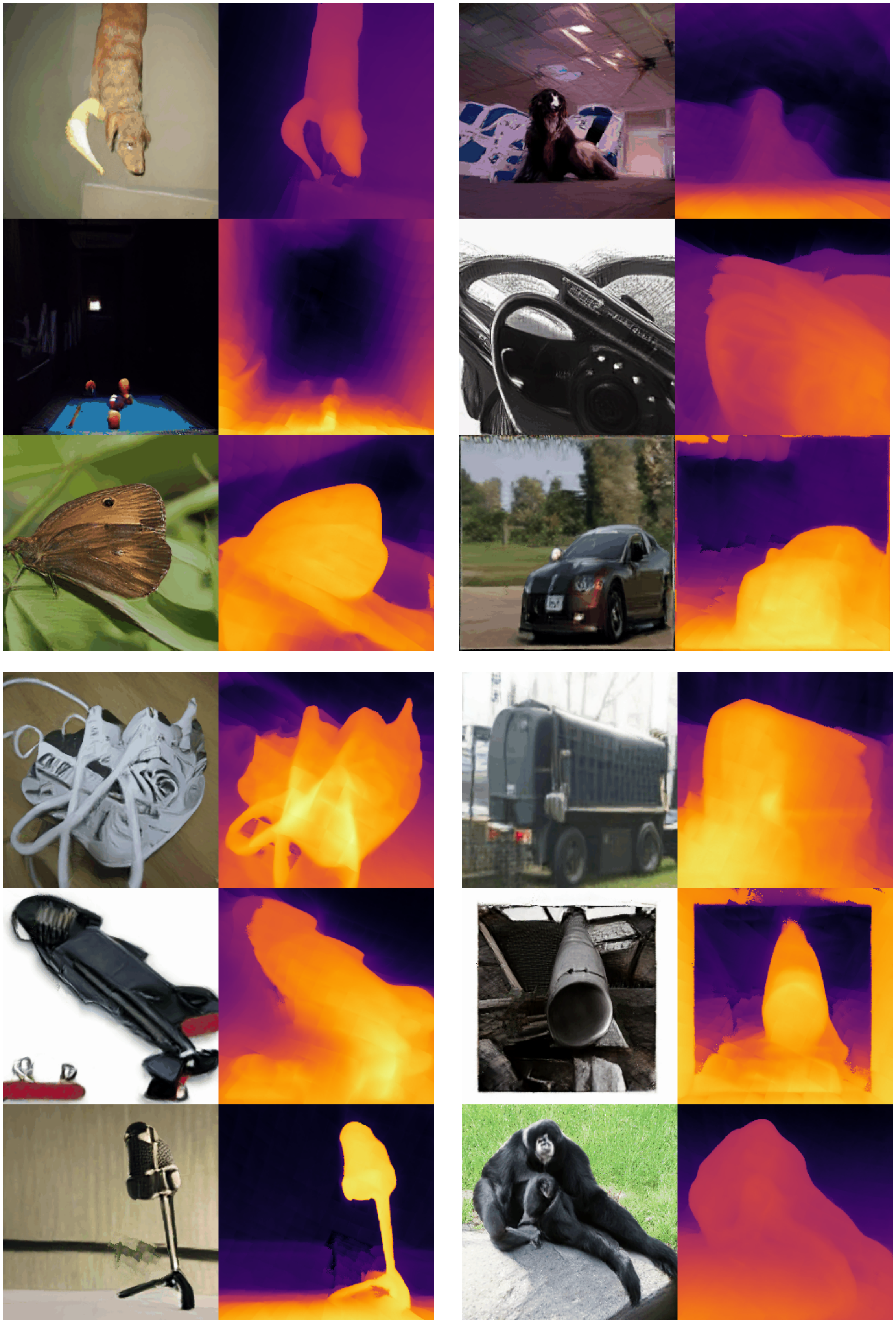

Figure 4: More uncurated fully generated samples from our Stage 2 model.

[7] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021. 2

[8] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[10] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. 4

[12] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ArXiv preprint*, 2021. 2

[13] Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas Funkhouser, and Andrea Tagliasacchi. Scene Representation Transformer: Geometry-Free Novel View Synthesis Through Set-Latent Scene Representations. *CVPR*, 2022. 2

[14] Yichun Shi, Divyansh Aggarwal, and Anil K. Jain. Lifting 2d stylegan for 3d-aware face generation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. arXiv, 2020. 2

[15] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021. 1, 2