HyperDiffusion: Generating Implicit Neural Fields with Weight-Space Diffusion - Supplementary Document -

Ziya Erkoç[1] Fangchang Ma[2] Qi Shan[2] Matthias Nießner[1] Angela Dai[1]
[1]Technical University of Munich [2]Apple

Appendix

In the supplementary document, we present additional 3D and 4D shape results (Section 1) and explain the implementation methods and parameters in detail (Section 2).

1. Additional Qualitative Results

We provide additional unconditional generation results on 3D and 4D generation in Figure 2 and Figure 1. We can generate diverse sets of shapes in both 3D and 4D settings. Resulting meshes are clean, smooth, and can be readily used in any 3D design software and game engines. Although we can output 16 frames for animation sequences, we only show 3 frames in Figure 1. Full animation sequences are available in our website.

2. Implementation Details

We use the diffusion and transformer architecture implementations of [2], which are modified versions of OpenAI and minGPT implementations, respectively. We have a predetermined MLP structure which consists of 3 hidden layers, each with 128 neurons. To process the MLPs with a transformer architecture, we first flatten the MLP weights into a 1D vector. Additionally, as a way to establish correspondence between components within the 1D vector and the MLP layers (*e.g.*, first n values are weights of the first layer), each layer is considered as two tokens, one for its weights and the other for its biases. Hence, in total we have 8 tokens coming from weights and biases. Thanks to this decomposition, transformer may figure out interaction between weights and biases across different layers during training. We also have one additional token representing the sinusoidal embedding of the timestep value. During synthesis of new samples, we again decompose the generated 1D vector into each layer's weights and biases, and load them into the same MLP structure.

Our transformer has 2880 hidden size (i.e., the size of each token after linear projection), 12 layers, and 16 self-attention heads. We use 500 diffusion timesteps in our implementation and a linear noise scheduler ranging between $1e^{-4}$ and $2e^{-2}$. For the sampling strategy, we used DDIM [3] and do not skip any timesteps during sampling. In addition, our denoising network directly predicts the denoised version, following [2]. We observe that $\approx 15\%$ of airplane, $\approx 16\%$ of chair and $\approx 51\%$ of car shapes in our train split of ShapeNet [1] contain major self-intersections in their original shape mesh faces, and so we exclude them from the training set for both our approach as well as for all baselines.

We also apply de-duplication to generated 3D shape results. Note that this has not been applied to baseline approaches, since their quantitative performance degraded with de-duplication. We achieve de-duplication by sampling twice the necessary amount and removing the ones that are very close to each other.

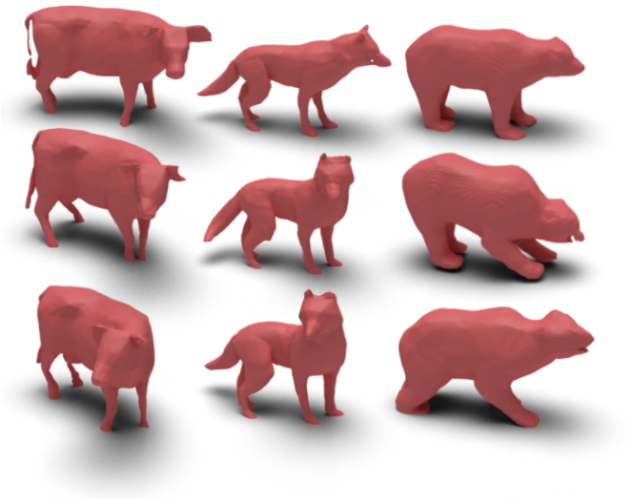

Figure 1: Additional unconditional 4D animation sequence generation results. We refer to our website for animated shape results.

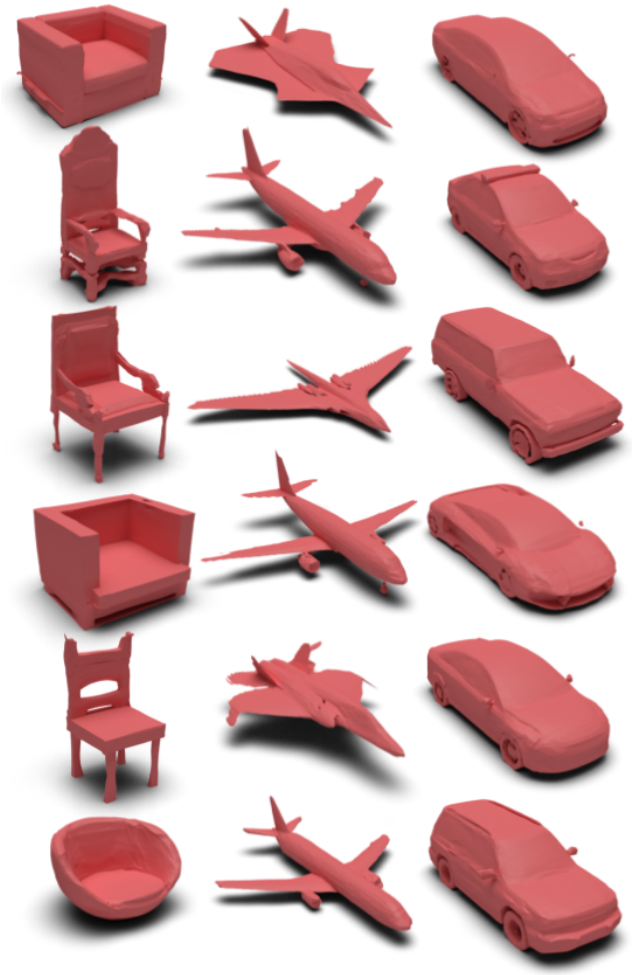

Figure 2: Additional unconditional 3D shape generation results.

References

[1] 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. 1

[2] William Peebles, Ilija Radosavovic, Tim Brooks, Alexei A Efros, and Jitendra Malik. Learning to learn with generative models of neural network checkpoints. *arXiv preprint arXiv:2209.12892*, 2022. 1

[3] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 1