Mosaic-SDF for 3D Generative Models

Supplementary Material

A. Generative model evaluation

In this section we provide additional information on the experiments described in Section 4.3.

A.1. Metrics

We measure distances between shape distributions following previous works [27, 51, 52, 54]. We quantify differences between a set of reference shapes S_r and a set of generated shapes S_g. We describe a shape $\mathcal{Y} \in S_r$ as a point cloud of size N sampled from a reference mesh using the farthest point sampling [9]. Similarly, $\mathcal{X} \in S_g$ is a point cloud sampled from a generated surface mesh, extracted as the 0-level set of the SDF (or 0.5 level set for occupancy function) using the Marching Cubes algorithm [26].

Geometric shape similarity. The Chamfer Distance (CD) and the Earth Mover Distance (EMD) measure similarity between two point clouds:

$$\mathrm{CD}(\mathcal{X},\mathcal{Y}) = \sum_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}\in\mathcal{Y}} \|x-y\|_2^2 + \sum_{\boldsymbol{y}\in\mathcal{Y}} \min_{\boldsymbol{x}\in\mathcal{X}} \|x-y\|_2^2 \quad (18)$$

$$\mathrm{EMD}(\mathcal{X},\mathcal{Y}) = \min_{\gamma:\mathcal{X}\to\mathcal{Y}} \sum_{\boldsymbol{x}\in\mathcal{X}} \|x-\gamma(y)\|_2 \quad (19)$$

where γ is the bijection between the point clouds, and $N = 5K$. In the following, we denote by $D(\mathcal{X},\mathcal{Y})$ distance measure between two point clouds, referring to either CD or EMD.

Geometrical distances between sets of shapes. The CD and EMD distances between point clouds are used to define the following distances between *sets* of shapes S_r and S_g: Coverage (COV) quantifying the diversity of S_g by counting the number of reference shapes that are matched to at least one generated shape; Minimum Matching Distance (MMD) measuring the fidelity of the generated shapes to the reference set; and 1-Nearest Neighbor Accuracy (1-NNA) describing the distributional similarity between the generated shapes and the reference set, quantifying both quality and diversity. Next we provide the mathematical definitions of these distance measures:

$$\mathrm{COV}(S_g,S_r) = \frac{|\{\arg\min_{\mathcal{Y}\in S_r} D(\mathcal{X},\mathcal{Y}) | \mathcal{X}\in S_g\}|}{|S_r|} \quad (20)$$

$$\mathrm{MMD}(S_g,S_r) = \frac{1}{|S_r|} \sum_{\mathcal{Y}\in S_r} \min_{\mathcal{X}\in S_g} D(\mathcal{X},\mathcal{Y}) \quad (21)$$

$$\text{1-NNA}(S_g,S_r) = \frac{\sum_{\mathcal{X}\in S_g} \mathbb{I}\left[N_{\mathcal{X}} \in S_g\right] + \sum_{\mathcal{Y}\in S_r} \mathbb{I}\left[N_{\mathcal{Y}} \in S_r\right]}{|S_r| + |S_g|} \quad (22)$$

where $\mathbb{I}(\cdot)$ is the indicator function and $N_{\mathcal{X}}$ is the nearest neighbor of $\mathcal{X}$ in the set $S_r \cup S_g - \{\mathcal{X}\}$.

Perceptual distances between sets of shapes. Alongside the geometric distance-based metrics, we adopt the 3D analogs of the Fréchet Inception Distance (FID) and Kernel Inception Distance (KID) suggested in previous works [35, 54]. In the 3D case, FID/KID are computed on the feature sets computed by pushing $N = 2046$ point samples into a pre-trained PointNet++ network [39]. We denote by $\mathcal{R}$ and $\mathcal{G}$ the sets of the extracted features from the reference shapes S_r and the generated shaped S_g, respectively. We can further define (μ_r, Σ_r) as the the mean and covariance statistics computed from the feature set $\mathcal{R}$, and similarly (μ_g, Σ_g) for $\mathcal{G}$. As in [54], the Fréchet PointNet++ Distance (FPD) and Kernel PointNet++ Distance (KPD) are defined by

$$\mathrm{FPD}(S_g,S_r) = \|\mu_g - \mu_r\| + \mathrm{Tr}\left(\Sigma_g + \Sigma_r - 2(\Sigma_g\Sigma_r)^{\frac{1}{2}}\right) \quad (23)$$

$$\mathrm{FPD}(S_g,S_r) = \left(\frac{1}{|\mathcal{R}|} \sum_{\mathbf{x}\in\mathcal{R}} \max_{\mathbf{y}\in\mathcal{G}} K(\mathbf{x},\mathbf{y})\right)^2 \quad (24)$$

where $K(\cdot,\cdot)$ is a polynomial kernel function distance.

A.2. Computation of distance metrics

Following previous works we compute the geometric distances, *i.e.*, COV, MMD and 1-NNA, with the reference set of shapes, S_r, chosen to be the test split; and we generate an equal number of shape from our generated set S_g. The per-class test split given by [53] consists of 5% of the shapes from each class, with varying numbers of shapes in each class. We used the following released codes to compute CD[1] and EMD[2], and computed the distance metrics from the official code of [52][3].

For computing the perceptual distances, FPD and KPD, we follow [54] and use 1K generated shapes S_g, and as the reference set S_r we take 1K randomly sampled shapes from the train split. We utilize a pre-trained PointNet++ model from [50] to extract the features $\mathcal{R}$ and $\mathcal{G}$.

To run the baselines, we use the official implementation of each method together with the pre-trained model they supply: The per-class unconditional models from Neural Wavelet[4], and the class conditioned model from 3DILG[5] and 3DShape2VecSet[6].

[1] https://github.com/ThibaultGROUEIX/ChamferDistancePytorch

[2] https://github.com/daerduoCarey/PyTorchEMD

[3] https://github.com/nv-tlabs/LION/tree/main

[4] https://github.com/edward1997104/Wavelet-Generation

[5] https://github.com/1zb/3DILG

[6] https://github.com/1zb/3DShape2VecSet

B. Additional implementation details

In this section we provide additional implementation details missing from the main paper.

Computing M-SDF representation. As described in Section 3.2 and algorithm 1, the computation of the M-SDF representation for a given shape $\mathcal{S}$ consists of two stages: initialization and fine-tuning. For both stages we require a good estimation of the ground truth SDF $F_{\mathcal{S}}$, and for that we use the open-source library of Kaolin-Wisp[7]. To obtain s (equation 7), that serves as the initial scale, we sample the surface densely and search for the minimal distance between the dense sampling set and the initialized volume centers. For the fine-tuning stage we sample for supervision a set of 300K points on the surface, and 200K points near the surface perturbed with Gaussian noise with variance 0.01. In each fine-tuning step we sample a random batch of 16K points, used to compute the loss in equation 8. We run the fine-tuning for 1K steps with ADAM optimizer [17] and learning rate of $1\mathrm{e}{-4}$.

M-SDF representation configuration. The number of local grids is chosen to be $n = 1024$ as this is the common size of the generated point cloud used in previous point-based diffusion models [35]. The grid resolution was set to $k = 7$, as it is the highest dimension that our transformer architecture can be consistent with.

Other representations configurations. In section 4.2 we compare M-SDF representation to existing popular SDF representations used in 3D generative models. For the experiment results presented in Figure 5 we follow the configurations commonly used with these representations in previous works. Specifically, for INR we had 8 hidden layers and changed the width of the hidden layers appropriately to the given parameter budget. As for the 3D grid and triplane we only adjusted the grid's resolution, where the triplane planes have fixed features dimension of 32.

Conditioning tokens. To complete the architecture description in Section 4.1, we add details regarding the conditioning mechanism $\boldsymbol{c}$. For the class conditioning generation 4.3 we use a learned per-class embedding where each class is described using a latent vector of size 128. For a selected class latent we first apply a linear layer projecting it to the transformer dimension, *i.e.*, 1024 before feeding it to the transformer. For the text conditioning generation 4.4, we utilize a pre-trained text model [40] as our textual embedding, result in a token embedding with feature size of 768 and maximum sequence length of 32. We feed these additional 32 tokens to the transformer, after applying a linear layer projecting to 1024.

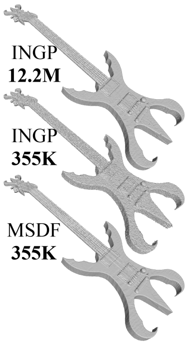
Generation timing. In Table 3 we report the time (in seconds) and Number of Function Evaluations (NFE) for generating one sample according to algorithm 3, using different ODE solvers: Midpoint rule, and Dormand–Prince method (DOPRI) [6]. We further report the effect of the different solvers on the quality of the generated shapes using the 1-NNA metric. For this experiment we evaluated our class-conditioning model, on 300 generated samples from the 'airplane' class. Please note that in all of the paper's experiments and evaluations we used the DOPRI as our ODE solver, however as Table 3 indicates using the Midpoint method, with either 25 or 50 steps, results in faster generation and only a mild degradation in quality. Using recent advances in fast sampling of flow models is expected to reduce these times further.

	NFE	Time	1-NNA (↓,%)	
		(sec.)	CD	EMD
Midpoint-25	50.00	6.13	59.16	67.57
Midpoint-50	100.00	12.19	61.88	69.06
DOPRI	138.46	16.97	57.67	64.85

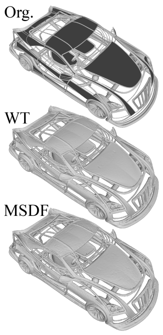
Table 3. Generation complexity and quality for different ODE solvers.

C. Additional representation evaluations

Comparison to Instant-NGP representation. To complement with the M-SDF representation evaluation shown in Figure 3, we further examine the Instant-NGP (INGP) [33] representation power for a fixed parameter budget. Please note that the INGP representation has a more complicated structure, as it consists of a triplet: coarse level grid; ordered set of hash tables; and weights of a small MLP. These lead to a representation incorporating different tensors with various symmetries, which might be possible to work with but was not done in previous 3D generative models and is non-trivial. As the inset shows, INGP best performance is achieved when using the INGP's original configuration (12.2M params), which is considerably larger than MSDF (355K params) that still leads to better approximation of the guitar.



The manifold assumption Man-made 3D shapes are typically not manifolds, however practically all shapes can be described as manifolds (*e.g.*, with small width). The manifold assumption has its advantages in defining implicit representations and post-processing (extracting a mesh) and therefore widely common assumption in other 3D generative models. In the inset we also add a visualization of a model with thin structures, after its processing to be watertight (WT), and the MSDF representation result.



[7] https://github.com/NVIDIAGameWorks/kaolin-wisp

D. Additional results

Class conditioning generation. In Figure 8 we show additional qualitative comparison of the class-conditioned generation compared to the relevant baselines. On top we show the common classes across all baselines. Below the dashed line we further present other classes in a comparison to 3DILG[53] and S2VS[54], which trained a class conditioning model similarly to us. Note that M-SDF generation are overall sharper with more details, while baselines tend to over smooth.

Guidance scale ablation. We perform an ablation study regarding the guidance scale ω we use in the sampling algorithm 3. Figure 9 depicts the generation samples using different guidance scales, with both our class-conditioning model (top) and the text-conditioning model (bottom). We further provide quantitative comparison in Table 4, when sampling our class conditioning model using various guiding scales. Here, we perform similar evaluation to the class conditioning evaluation in Table 1, and report metrics for the 5 largest classes in the ShapeNetCore-V2 (3D Warehouse) [4] dataset. As somewhat expected, the $\omega = 0$ performs best when comparing shape *distributions*, however qualitatively, taking a higher ω tends to result in a more ”common” or ”average” shape. In the main paper we therefore opted $\omega = 0$ for the class-conditional shape generation, and $\omega = 5$ for the text-conditioned shape generation.

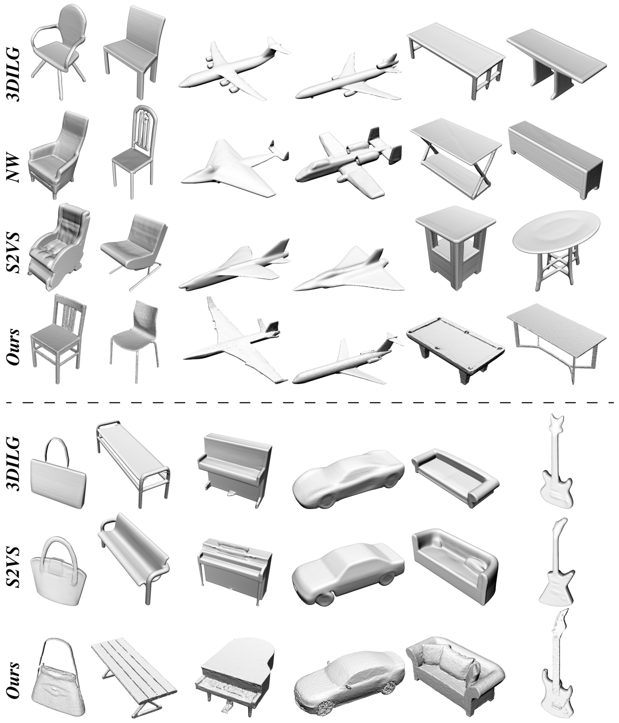


Figure 8. Class conditioned generation of 3D shapes compared to relevant baselines.

	FPD (↓)	KPD (↓)	COV (↑,%)		MMD (↓)		1-NNA (↓,%)	
			CD	EMD	CD	EMD	CD	EMD
airplane								
Ours $\omega = 0$	0.37	0.37	50.99	48.02	3.46	3.71	57.67	64.85
Ours $\omega = 1$	0.71	0.75	44.06	40.59	4.03	3.92	65.35	74.75
Ours $\omega = 2$	0.80	0.79	37.13	41.09	4.65	3.84	70.54	73.27
Ours $\omega = 5$	1.10	1.19	36.63	35.64	4.90	4.14	74.50	78.22
Ours $\omega = 10$	1.97	2.90	27.72	27.72	6.25	4.54	86.14	81.93
car								
Ours $\omega = 0$	0.45	0.47	42.86	45.14	2.75	2.78	65.71	70.00
Ours $\omega = 1$	0.85	1.00	32.00	37.71	3.14	2.87	77.14	72.29
Ours $\omega = 2$	0.96	1.15	29.71	35.43	3.24	2.85	75.14	72.29
Ours $\omega = 5$	1.08	1.28	28.57	34.86	3.41	3.02	80.86	74.86
Ours $\omega = 10$	1.39	2.11	24.00	28.57	3.53	3.18	85.71	83.71
chair								
Ours $\omega = 0$	0.51	0.20	45.86	51.48	16.08	9.17	55.92	55.47
Ours $\omega = 1$	0.78	0.64	44.08	42.60	18.70	10.39	56.07	64.79
Ours $\omega = 2$	0.94	0.85	38.46	42.60	20.16	10.59	66.27	70.56
Ours $\omega = 5$	1.34	1.42	30.77	33.43	22.43	11.13	74.26	74.26
Ours $\omega = 10$	1.67	1.92	31.07	30.47	22.51	11.62	76.18	77.96
sofa								
Ours $\omega = 0$	0.64	0.65	44.94	50.63	11.21	7.21	59.49	58.23
Ours $\omega = 1$	1.31	1.64	36.08	36.71	15.31	8.08	69.30	63.92
Ours $\omega = 2$	1.68	2.25	27.85	34.81	17.50	8.44	80.70	72.78
Ours $\omega = 5$	2.51	4.02	22.15	31.65	20.03	8.89	87.66	79.43
Ours $\omega = 10$	3.48	6.57	17.72	25.32	21.07	9.86	89.56	80.70
table								
Ours $\omega = 0$	0.49	0.18	52.26	55.58	13.10	7.60	52.14	51.54
Ours $\omega = 1$	1.26	1.43	39.90	43.47	15.16	8.47	65.20	63.06
Ours $\omega = 2$	1.97	2.55	32.30	31.12	18.63	9.81	73.40	73.99
Ours $\omega = 5$	3.08	4.55	17.81	18.05	36.52	14.99	89.31	88.95
Ours $\omega = 10$	4.34	8.34	13.54	14.25	54.02	19.28	95.72	94.42

Table 4. Ablation study on the Classifier Free Guidance (CFG) scale used for sampling ω. KPD and MMD-CD multiplied by 10^3, MMD-EMD by 10^2.

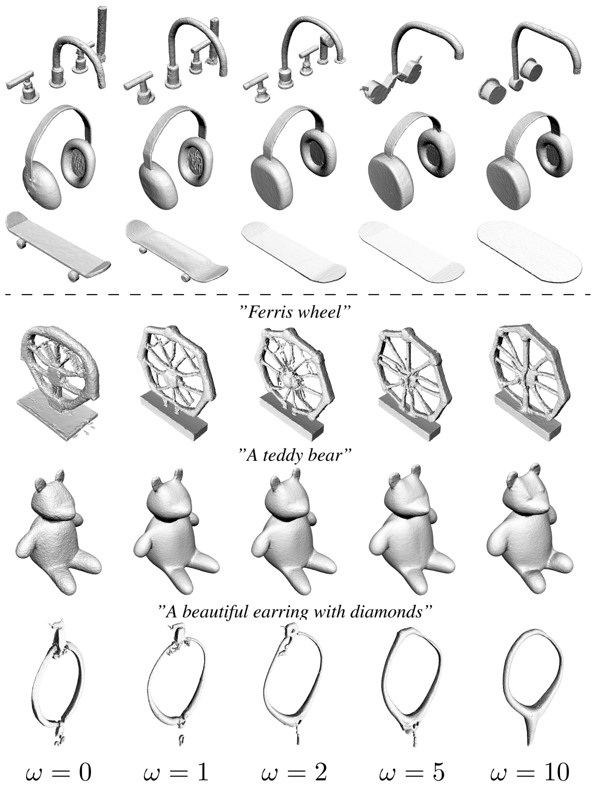


Figure 9. Ablation of guidance scale ω use in sampling our class-conditioned model (top) and our text-conditioned model (bottom).

Text-to-3D generation. In Figure 10 we show additional generated shapes from our text-conditioned model.

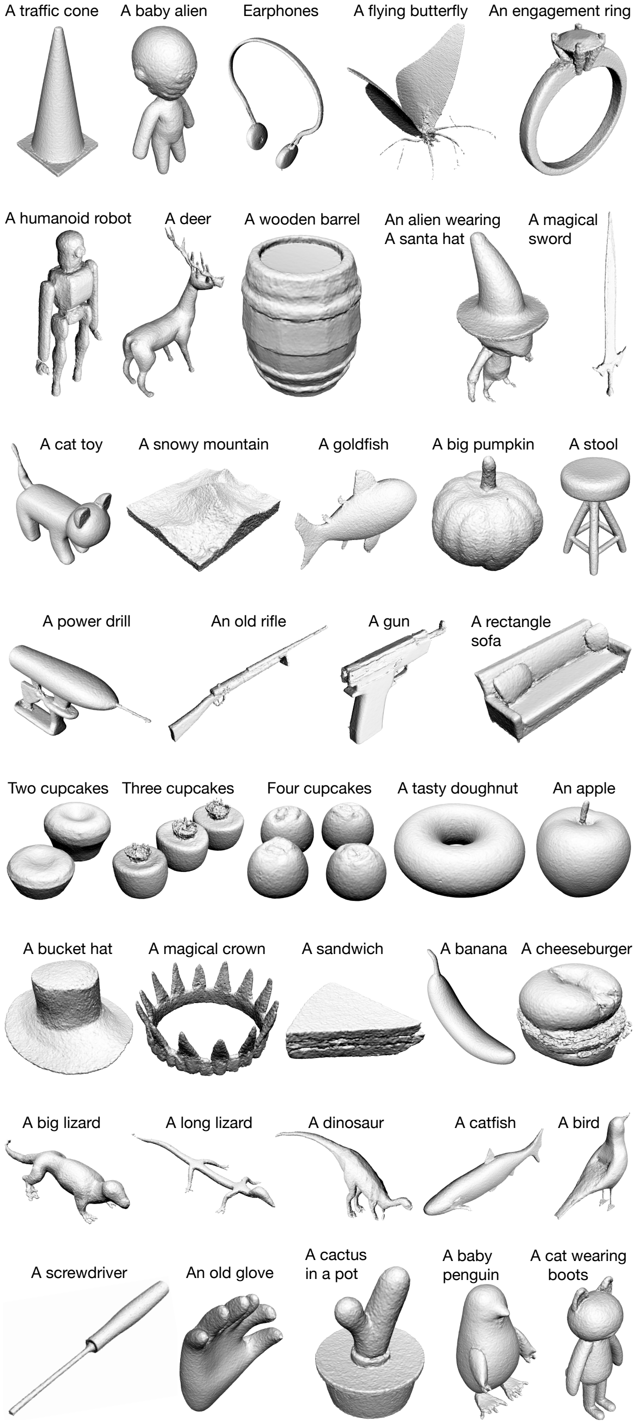


Figure 10. Additional text-to-3D samples from a Flow Matching model trained on M-SDF representations of 600K pairs of shapes and text.