Memorization in 3D Shape Generation: An Empirical Study

Supplementary Material

A. Metrics Definition and Implementation

In this section, we detail the definitions and implementation of the metrics and the evaluation framework from Section 3.

A.1. Distance Metrics

Chamfer distance (CD) measures the average squared nearest-neighbor distance between two point clouds. We randomly sample 4096 points to compute CD. Given two point clouds $S_1, S_2 \subset \mathbb{R}^3$, their CD is defined as:

$$d_{\text{CD}}(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \frac{1}{|S_2|} \sum_{y \in S_2} \min_{x \in S_1} \|y - x\|_2^2 \tag{1}$$

Point cloud encoders encode a point cloud S into an embedding $f(S) \in \mathbb{R}^D$. We consider the following encoders and embedding dimensions:

$$\begin{aligned} D_{\text{PointNet++}} &= 512, \\ D_{\text{Uni3D}} &= 1024, \\ D_{\text{ULIP-2}} &= 1280 \end{aligned}$$

Given two point clouds S_1 and S_2, we define their L2-normalized embedding distance as:

$$d_{\text{emb}}(S_1, S_2) = 1 - \big\langle f(S_1), f(S_2) \big\rangle. \tag{2}$$

Light Field Distance (LFD) is computed by first rendering 256×256-resolution silhouettes of an object from 10 canonical viewpoints. Then, a feature vector is extracted from each silhouette by concatenating its Zernike moments and Fourier descriptors. The LFD between two shapes is computed by summing the L_1 distances between the corresponding view descriptors across the 10 viewpoints. We use the original implementation [6] to compute LFD.

Image encoders. For each mesh M, we render 12 RGB images $\{I_i(M)\}_{i=1}^{12}$ at resolution 256×256 using the same pipeline in Appendix D. We encode each view with an image encoder and use the [CLS] token as the per-view representation. The final embedding $f(M) \in \mathbb{R}^D$ for the mesh M is obtained by averaging the [CLS] tokens across the 12 views and L2-normalizing the result. Given two meshes M_1 and M_2, the embedding distance induced by an image encoder is defined analogously to the point cloud encoders as:

$$d_{\text{emb}}(M_1, M_2) = 1 - \big\langle f(M_1), f(M_2) \big\rangle. \tag{3}$$

A.2. Memorization Metrics

Mann-Whitney U Test is a well-established non-parametric test. Following [34], we use it to detect data-copying behavior, and we only consider its global version.

Let P denote the underlying data distribution. The training set T and the held-out test set P_{test} are both drawn i.i.d. from P. Let Q denote the set generated by a generative model. Let $d(x, T)$ be the distance from a point x to the training set T (*e.g.*, the distance to its nearest neighbor in T under a chosen metric). For each $x \in P_{\text{test}}$, define $A = d(x, T)$, and for each $y \in Q$, define $B = d(y, T)$. When the model generalizes, the distance distributions induced by P_{test} and Q should match. The null hypothesis, therefore, states that a generated distance B is larger than a test distance A with probability $1/2$:

$$H_0 : \Delta_T(P_{\text{test}}, Q) = \frac{1}{2}, \tag{4}$$

where

$$\Delta_T(P_{\text{test}}, Q) = \Pr(B > A). \tag{5}$$

In practice, let $n = |P_{\text{test}}|$ and $m = |Q|$, and collect the sets of distances:

$$\{A_i\}_{i=1}^n = \{d(x_i, T)\}_{i=1}^n, \qquad \{B_j\}_{j=1}^m = \{d(y_j, T)\}_{j=1}^m.$$

The Mann-Whitney U statistic for Q is:

$$U_Q = \sum_{j=1}^{m} \sum_{i=1}^{n} \mathbb{I}\big[B_j > A_i\big], \tag{6}$$

which is an unbiased estimator of $mn\,\Delta_T(P_{\text{test}}, Q)$. Equivalently, if we rank all $n + m$ values $\{A_i\}_{i=1}^n \cup \{B_j\}_{j=1}^m$ from smallest to largest, and let $R(B_j)$ denote the rank of B_j, then:

$$R_Q = \sum_{j=1}^{m} R(B_j), \qquad U_Q = R_Q - \frac{m(m+1)}{2}. \tag{7}$$

Mann-Whitney z-score. Under the null hypothesis H_0 and for $m, n \gtrsim 20$, the statistic U_Q is approximately normally distributed with mean μ_U and standard deviation σ_U:

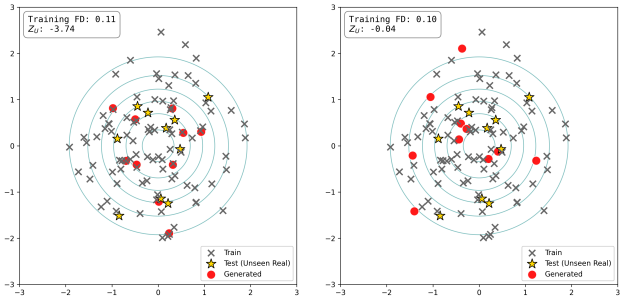
$$\mu_U = \frac{mn}{2}, \qquad \sigma_U = \sqrt{\frac{mn(m+n+1)}{12}}. \tag{8}$$

We therefore define the normalized Mann-Whitney z-score as:

$$Z_U(P_{\text{test}}, Q; T) = \frac{U_Q - \mu_U}{\sigma_U}. \tag{9}$$

Intuitively, $Z_U \ll 0$ indicates data-copying (generated samples are systematically *closer* to the training set than test samples). $Z_U \approx 0$ indicates generalization.

A.3. Evaluation Framework



(a) Memorization (b) Generalization

Figure 10. **Toy example illustrating the relationship between Z_U and training FD.** Grey crosses, gold stars, and red circles denote training, test, and generated data, respectively. Training and test data are sampled from a 2D Gaussian distribution. In the left plot, the generated samples memorize the training data, whereas in the right plot, the generated samples generalize. Z_U (-3.74 *vs*. -0.04) captures the difference in memorization between the two scenarios, while training FD does not (0.11 *vs*. 0.10).

Encoder. We employ training FD and test FD as quality metrics. Based on Section 3.1, we select Uni3D as the best semantic encoder, and we use it in our FD calculation.

Relation between Z_U and training FD. While training FD often correlates with memorization (*e.g.*, a decrease in Z_U generally accompanies a decrease in training FD), here, we use a toy example in Figure 10 to show that a low training FD may not always indicate memorization. In Figure 10a, we draw training, test, and generated samples from a 2D Gaussian distribution, where the generated samples are created by sampling near the training points (*i.e.*, memorization). In Figure 10b, the generated samples follow the same distribution, but are not particularly close to the training data (*i.e.*, generalization). Z_U (-3.74 *vs*. -0.04) captures the difference in memorization between the two scenarios, whereas training FD is almost identical.

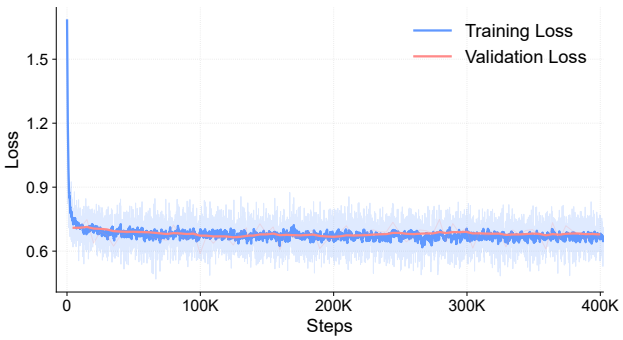


Figure 11. **Baseline model loss curves.** The training and validation losses of the baseline model throughout the training process.

Alternative metrics. We observe that although the baseline model progresses from low-quality to high-quality generation throughout training (see Figure 3), the training and validation losses fail to reflect this evolution. In Figure 11, both losses converge at an early stage (approximately 10K steps) and then oscillate around 0.6. Consequently, we do not consider loss to be a reliable metric for quality evaluation.

B. Existing Models' Retrieval Visualization

In this section, we visualize retrieval results for several existing models. Table 7 summarizes the conditioning type, training data, and data split for all models considered.

model	cond.	training dataset	split
NFD	Uncond	ShapeNet *chair*	-
Wavelet Generation	Uncond	ShapeNet *chair*	IM-NET
LAS-Diffusion	Uncond	ShapeNet *chair*	IM-NET
LAS-Diffusion	Class-cond	Five ShapeNet classes	IM-NET
3DShape2VecSet	Class-cond	ShapeNet	3DILG
Michelangelo	Text-cond	ShapeNet	3DILG
3DTopia-XL	Text-cond	256K Objaverse subset	-
Trellis	Text-cond	Trellis500K	-

Table 7. **Conditioning type, training dataset, and dataset split for all evaluated models.** A dash (-) indicates that the official split is not publicly released.

Figure 25 shows retrieval results for six models on the *chair* category. Figure 26 shows retrieval results for five models on their respective full training sets.

The trends are consistent with the quantitative results in Section 4: models trained on smaller datasets (NFD, LAS-Diffusion, and Wavelet Generation) exhibit strong memorization; even among generated chairs in the 60^{th}–80^{th} percentiles (ranked by distance to the nearest training shape), many closely match training shapes. In contrast, models trained on larger and more diverse datasets show strong generalization, with even generated shapes at lower percentiles being noticeably more diverse and exhibiting novel features.

C. Detailed Experimental Setups

This section details the model architectures, datasets, and training configurations employed in the controlled experiments of Section 5 and 6. Furthermore, we include supplementary experiments and qualitative visualizations to support the findings presented in the main text.

C.1. Dataset and Model Setup

Dataset pipeline. Our dataset is sourced from Objaverse-XL [11]. We rank classes by frequency within the Objaverse-LVIS subset and select the 100 most common single-object classes. Rather than re-captioning the entire Objaverse-XL, we use the existing Cap3D captions [31].

To ensure data quality, we filter caption-label pairs for semantic alignment using Qwen3-8B Embedding [60]. We

Caption Template:

```
Given an object description,
classify it into one of the
fine-grained object classes.
The description is:
{description}
```

Label Template:

```
"class label: {class}"
```

Figure 12. **Instruction template** used to encode Cap3D captions and object labels with Qwen3-8B Embedding.

compute caption and class label embeddings using the instruction templates shown in Figure 12 and measure their cosine similarity. We retain approximately 229K caption-label pairs by applying a similarity threshold of $\tau = 0.7$.

You are a strict data screener for single-object 3D captions.
TASK: Given (label, caption), decide if the caption predominantly and concretely describes ONE instance of the object class == label.
STRICT RUBRIC (all must be true for KEEP=YES):
1) SINGLE OBJECT: Caption is about a single object instance, not a set, row, collection, scene, room, or environment.
2) LABEL DOMINATES: The main described entity is the label; the label isn't just a small part of something else.
3) NOT PART-ONLY: The description is not mainly a part-of-object (*e.g.*, "table leg", "door handle"), unless the label itself is such a part class.
4) WHEN UNSURE, SAY NO. Favor precision over recall.
SIMPLIFY CAPTION (if KEEP=YES):
- Remove color and material words.
- Keep geometry/parts/structure words that help identify the object shape.
OUTPUT FORMAT (MUST be a valid one-line JSON):

```
{ "keep": "yes"|"no",
"simple_caption": "<string or empty
if no>" }
```

Figure 13. Prompt - Filtering out misaligned caption-label pairs.

Then, we apply Qwen3-30B [51] to aggressively remove mislabeled pairs. As shown in Figure 13, we keep only pairs for which the LLM is confident and discard low-quality captions. To reduce class imbalance, we cap the most frequent object class at 10K examples. As a result, we obtain around 140K caption-label pairs, which we randomly split into 120K for training and 20K for testing. The class distribution of our customized dataset is shown in Figure 14.

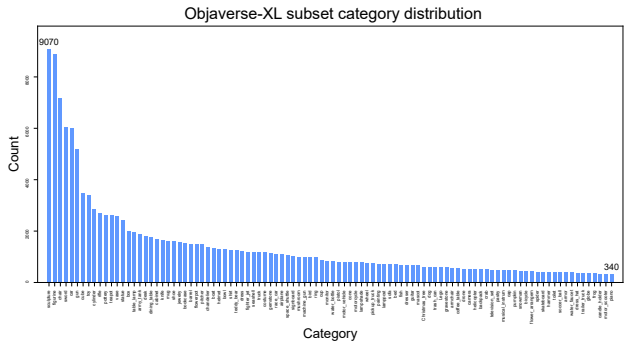


Figure 14. **Category distribution** of our customized 100-category subset of Objaverse-XL.

Vecset autoencoder. We use the latest pre-trained Vecset autoencoder, VecSetX, released by 3DShape2VecSet [56]. The model takes sub-sampled input point embeddings as queries and is trained with an SDF regression objective [28] and an Eikonal regularizer [14]. The bottleneck follows the normalized bottleneck autoencoder design [54]. The network consists of 24 layers of self-attention blocks with a hidden dimension of 1024, and the resulting latent code has shape 1024×32 by default.

Formally, let $D(x, f)$ denote the predicted SDF value at any query spatial location x given latent code f, and let $s(x)$ denote the ground-truth signed distance. The autoencoder is trained using a combination of an SDF regression loss and an Eikonal loss:

$$\mathcal{L} = \mathcal{L}_{\text{SDF}} + \lambda_{\text{eik}}\, \mathcal{L}_{\text{eik}}. \tag{10}$$

The SDF regression term is:

$$\mathcal{L}_{\text{SDF}} = \mathbb{E}_x \big[|D(x, f) - s(x)| \big], \tag{11}$$

and the Eikonal regularization term is:

$$\mathcal{L}_{\text{eik}} = \mathbb{E}_x \Big[\big(\|\nabla_x D(x, f)\|_2 - 1 \big)^2 \Big]. \tag{12}$$

Diffusion backbone. We build upon the Hunyuan3D 2.1 codebase [23]. The architecture is a flow-based diffusion model composed of multiple Hunyuan-DiT blocks [29]. It uses cross-attention layers to inject conditioning signals and replaces the feed-forward layers in the last few Transformer blocks with mixture-of-experts (MoE) layers.

C.2. Baseline and Sanity Check

In this section, we introduce three model sizes and describe controlled experiments on dataset size and model size.

Baseline model configuration. Our baseline model comprises approximately 323M parameters, consists of 12 Transformer blocks with a hidden dimension of 1024, and 16 attention heads. Following the official Hunyuan3D 2.1

implementation, we disable positional embeddings and attention pooling. The model is text-conditional: we use CLIP-B/16 [39] as the text encoder, giving a conditioning dimension of 512. We use three MoE layers in the backbone, each with four experts and top-2 gating.

model	#params	blocks	hidden dim	attn heads
small	80M	12	512	8
baseline	323M	12	1024	16
large	1.5B	16	2048	16

Table 8. **Hyperparameter specifications for model variants.** All models share the same text encoder and MoE settings.

Model scaling variants. To investigate the impact of model capacity, we introduce a lightweight small model and a scaled-up large model. These variants retain the same architectural components as the baseline, including the CLIP-B/16 text encoder, MoE configuration, and the lack of positional embeddings, but vary in network depth and width. We summarize the specific hyperparameter settings for all three model configurations in Table 8.

Dataset size sanity check. We train our baseline model with several class numbers and dataset size settings. Figure 15 shows that as the dataset size scales up, the memorization behavior is substantially reduced. Since the 16-class subset contains only about 58K training shapes, for the 16-class experiments, we only use up to 50K samples.

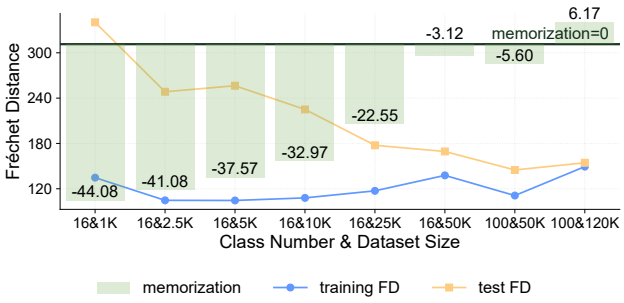


Figure 15. **Larger datasets reduce memorization.** With the same diffusion model, increasing the dataset size decreases Z_U.

Model size sanity check. We also evaluate how model size affects memorization. We train the small, baseline, and large models on a 16-class subset of 50K shapes for 200K steps. Figure 16 shows that larger models exhibit stronger memorization. It is worth noting that the small model fails to generate high-quality shapes and has high FD on both the training and test sets.

Our experimental results are consistent with the existing literature [5, 42, 53] on the impact of dataset size and model size on memorization.

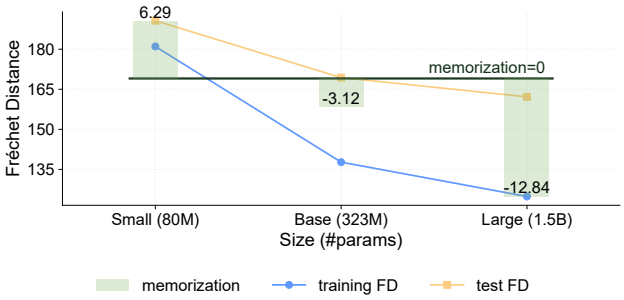


Figure 16. **Larger models strengthen memorization.** With the same training dataset, increasing the model size decreases Z_U.

C.3. Data Modality

In this section, we provide additional details on the implementation and training of the image generation model, along with supplementary experimental results.

Dataset. Instead of using the baseline model's 16-class training data, we use the 100-class dataset. We ensure this dataset is identical to the 50K dataset used for the 100-class model presented in Figure 15.

Rendering pipeline. We render the 256×256 image dataset following the procedures detailed in Appendix D. To constrain the dataset size to 50K, we exclusively use the sixth view from the 12 renderings available for each object.

Image autoencoder. We use the Flux VAE to encode input images of resolution $3 \times 256 \times 256$ into latent grids of shape $16 \times 32 \times 32$. After injecting 2D positional embeddings, we flatten the spatial dimensions to yield a latent sequence of size 1024×16, which serves as the input to the diffusion model. We select the Flux VAE not only for its superior reconstruction capabilities but also to ensure the latent sequence length aligns with our Vecset configuration.

modality	DinoV2 (%)	SSCD (%)	LFD (%)
image	**52.71**	**68.05**	-
3D	22.04	32.56	46.44

Table 9. **Source data retrieval rate using the same 2500 training prompts for both image and 3D generative models. Higher** retrieval rate means higher probability of reproducing the source data. Images show a stronger tendency to replicate than 3D shapes.

Source data retrieval rate measures how often the nearest training sample to a generated sample is exactly the ground truth sample for the input prompt. While previous work [42] on large-scale text-to-image models reports that training prompts rarely regenerate their exact source images, Table 9 shows that our image model has a higher probability of retrieving training data than the 3D generative model when conditioned on training captions. This further demonstrates that images are more susceptible to memorization.

C.4. Data Diversity

Our experiments use the 16, 32, 64, and 100 most frequent classes in the original dataset. For each class subset, we sample 50K examples by taking samples from each class in proportion to its frequency within the selected class subset.

C.5. Conditioning Granularity

In this section, we provide further information on how we construct class and text conditions of different granularities.

Coarse-grained class labels. We use the same dataset as the baseline but introduce an additional set of coarse-grained class labels. We adopt the GObjaverse taxonomy, which originally comprises 10 categories: animals, daily-use, electronics, furniture, human-shape, transportation, buildings&outdoor, plants, food, and poor-quality. However, since the distribution of our top-16 classes (Figure 14) shows minimal representation of buildings&outdoor, plants, and food, we exclude these categories from our experiments. Additionally, we remove the poor-quality category and re-classify any abstract shape descriptions as daily-use. Consequently, each shape in our dataset is assigned to one of the remaining 6 coarse-grained categories (animals, daily-use, electronics, furniture, human-shape, and transportation) using the prompt shown in Figure 17. The coarse-grained category distribution is shown in Figure 18.

You are a precise data labeler.
TASK: Given a label and a caption, choose exactly ONE super-category from the provided CLOSED SET.
ALLOWED CATEGORIES:
- animals, daily-use, electronics, furniture, human-shape, transportation

GUIDANCE:
- Use the caption semantics to determine the super-category.
- **Semantic Rule**: If the description suggests a toy, LEGO, or miniature but depicts a real-world concept (*e.g.*, a toy car), choose the semantic category (*e.g.*, Transportation).
- **Closed Set**: Never invent categories; pick ONLY from the allowed list.
- **Fallback**: If classification is truly impossible, select "daily-use".

OUTPUT FORMAT (Strict one-line JSON):
`{ "category": "<Category Name>" }`

Figure 17. Prompt - GObjaverse category classification.

Multi-granularity captioning. We generate captions using Qwen3-VL-8B-Instruct [2] across three distinct levels of granularity: phrase, sentence, and paragraph. To prompt Qwen3-VL, we randomly select four views from the 12 rendered images detailed in Appendix D. We show the lengths of different text granularities in Figure 19. The prompt used for the multi-granularity captioning is shown in Figure 20.

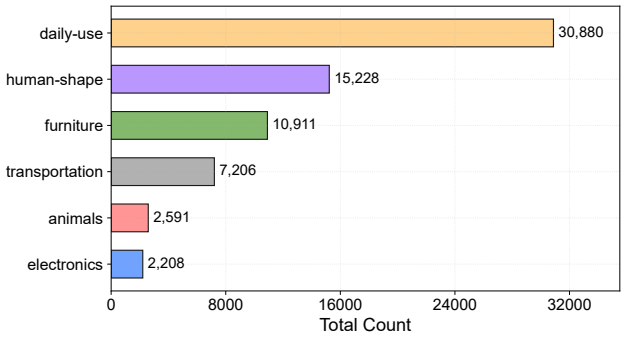


Figure 18. **Coarse-grained category distribution.** We use six categories (animals, daily-use, electronics, furniture, human-shape, and transportation) from the GObjaverse taxonomy.

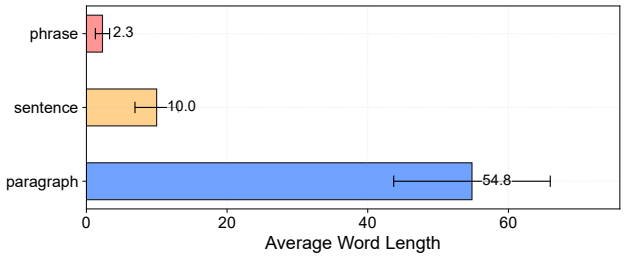


Figure 19. **Caption length distribution across granularities.** The distribution of caption length for phrase, sentence, and paragraph captions demonstrates distinct levels of detail.

You are given multiple views of the SAME object captured from different angles.
TASK: Combine evidence across views and describe the object at three distinct granularities.
GRANULARITY DEFINITIONS:
- **Phrase**: A concise 3-10 word noun phrase (no punctuation at the end).
- **Sentence**: 1-2 sentences summarizing main parts, colors, and overall shape.
- **Paragraph**: 3-6 sentences covering fine details like materials, textures, distinctive features, geometry, and any context seen in the background.

CONSTRAINTS:
- Return **ONLY** a compact JSON object.
- No extra text. No markdown. No newlines.
- Avoid speculation; if something is unknown, say 'unknown'.

OUTPUT FORMAT (Must use these exact keys):
`{ "phrase": "...", "sentence": "...", "paragraph": "..." }`

Figure 20. Prompt - Multi-granularity caption generation.

Class-conditional model configuration. Our class-

conditional model shares nearly the same architecture as our baseline model; the only difference lies in the conditioning embedder. Instead of using a text encoder, we construct a trainable 512-dimensional embedding matrix. For each class label, we learn a corresponding class embedding vector, which is then injected into the model via cross-attention.

C.6. Guidance Scale

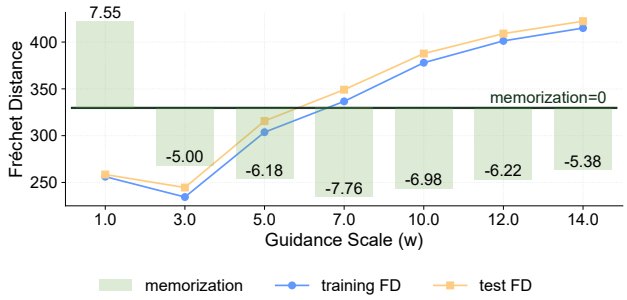


Figure 21. **Moderate guidance scales lead to the strongest memorization (class-conditional checkpoint)**, while further increasing the guidance scale reduces memorization, consistent with Section 6.1. Weak guidance also yields low memorization.

In Section 6.1, we evaluate six distinct guidance scales, $w \in \{0, 1, 3, 5, 7, 10\}$, across both our baseline and large models (details in Appendix C.2). Here, we provide an additional guidance scale experiment in Figure 21. Specifically, we run the same evaluation on a class-conditional checkpoint (details in Appendix C.5). This additional result is consistent with the conclusion in Section 6.1: memorization is strongest at moderate guidance scales, while stronger guidance reduces memorization.

w	gen	top match	caption (top match)	LFD
1			"a geometric sculpture with a trapezoidal base and a rectangular prism suspended above it by rods"	2427
3			Same as $w = 1$	1282
5			Same as $w = 1$	2467
7			"round base, vertical stem, domed shade"	7555
10			Same as $w = 7$	7999

Table 10. **Guidance scale case study for the prompt "a geometric sculpture with a trapezoidal base and a rectangular prism suspended above it by rods".** The model reproduces the training shape at lower scales ($w \in \{1, 3, 5\}$), and emphasizes sub-phrases such as "trapezoidal base" and "rods", but fails to generate the rectangular prism at larger w.

w	gen	top match	caption (top match)	LFD
1			"a cube"	1396
3			"blocky humanoid figure standing on a base"	390
5			"chair with high curved backrest and rectangular base"	1991
7			"abstract sculpture with rectangular body, square cap, smaller square base, recessed trapezoidal shape on vertical face"	2805
10			Same as $w = 7$	2188

Table 11. **Guidance scale case study for the prompt "blocky humanoid figure standing on a base".** The model generates a similar but lower-quality blocky figure with no guidance, reproduces the training shape when $w = 3$, and emphasizes sub-phrases such as "figure standing on a base", but fails to generate the blocky head with larger w.

Interestingly, in Figure 21, as the guidance scale increases from 3 to 7, training FD moves in the opposite direction of Z_U, despite the two metrics being generally correlated. One possible explanation is that, in this case, increasing the guidance scale leads to lower diversity and higher similarity to training shapes in the generated outputs. Because FD measures similarity at the distribution level, reduced diversity among generated samples results in a higher training FD. In contrast, Z_U is computed based on object-level similarity and is therefore less impacted by diversity.

Supplementary case studies. Tables 10 and 11 present two additional case studies involving different guidance scales. These results are consistent with the discussion in Section 6.1. We observe that generated shapes produced with higher guidance scales often fail to fully align with the complete prompt. Although these scales yield novel shapes that are distinct from the training set, we argue that while large guidance scales mitigate memorization, they do so at the cost of reduced prompt alignment.

C.7. Latent Space

Vecset is produced by applying cross-attention from a set of latent queries to point cloud positional embeddings. Following 3DShape2VecSet [56], the queries can be learnable vectors or sub-sampled point queries. Previous studies [7, 59] apply different sub-sampling rates to obtain different Vecset lengths. We adopt a similar point query design and compare three sequence lengths: 768, 1024, and 1280.

Vecset autoencoder performance. In Table 12, we evaluate the autoencoder's reconstruction performance. Varying the Vecset length does not significantly affect reconstruction

latent shape	dataset	CD ($\times 10^3$) $\downarrow$	F-Score ($t = 0.01$) $\uparrow$
(768×32)	Objaverse	12.50	0.86
(1024×32)	Objaverse	12.27	0.86
(1280×32)	Objaverse	**12.09**	**0.87**

Table 12. **Reconstruction performance of VecSetX** trained on the entire Objaverse. Different Vecset lengths result in similar performance in terms of both CD and F-Score.

quality: all three lengths yield comparable performance.

Supplementary case studies. Figures 22 and 23 present two additional case studies with different Vecset lengths. The results are consistent with the discussion in Section 6.2: shapes produced with longer Vecsets maintain high fidelity while showing reduced memorization.

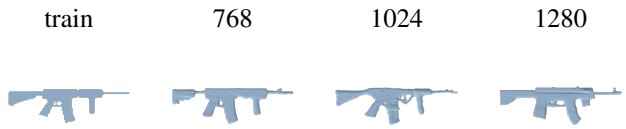


Figure 22. **Shapes generated with different Vecset lengths for the prompt "an assault rifle with a stock, foregrip, and pistol grip; a body; and a long barrel".** Vecsets of longer sequence lengths (1024 and 1280) generate high-quality shapes aligned with the training prompt, while exhibiting novel features (*e.g.*, stocks and foregrips) that are different from the training shape.

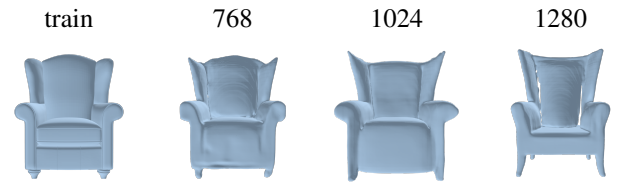


Figure 23. **Shapes generated with different Vecset lengths for the prompt "wing chair".** Vecsets of longer sequence lengths (1024 and 1280) generate high-quality shapes that are well aligned with the training prompt, while exhibiting novel features (*e.g.*, arms and wings) that are different from the training shape.

C.8. Rotation Augmentation

Rotation invariance of distance metrics. We evaluate the robustness of LFD and Uni3D by measuring the distance between an object's original pose and its rotated poses under each metric. Specifically, we apply random rotations sampled from $[0°, 360°]$ independently along the pitch (x), yaw (y), and roll (z) axes. If a metric is rotation invariant, the distance between the original and the rotated shapes should remain near zero.

As illustrated in Figure 24, LFD is sensitive to rotation, with high distances between original and rotated objects for all axes. In contrast, objects rotated along the yaw axis have a near-zero distance to the original objects under Uni3D.

Rotation implementation. To guarantee the accuracy of Z_U and FD, we limit our experiments to yaw rotation. In our dataset, the y-axis corresponds to the yaw axis. Consequently, we restrict our rotations to four discrete angles around the y-axis ($0°$, $90°$, $180°$, and $270°$) to ensure Uni3D remains consistent. For LFD, we compute distances across these four distinct poses for each generated sample and report the minimum retrieval distance.

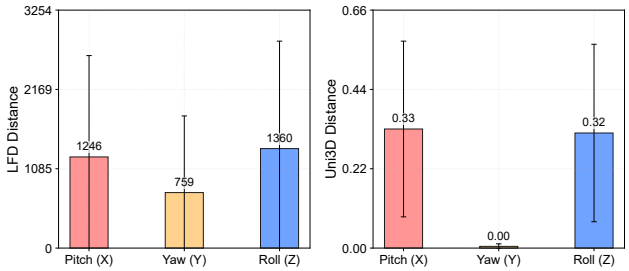


Figure 24. **Robustness of distance metrics to rotation.** Mean and standard deviation of LFD (left) and Uni3D (right) distances between original and randomly rotated objects. LFD exhibits significant variation across all rotation axes, indicating that it is not rotation-invariant. Uni3D demonstrates strong robustness (*i.e.*, low distances) specifically under yaw rotation.

D. Rendering Details

We use Blender to render $N = 12$ views for each object at 256×256 resolution. Prior to rendering, the geometry is normalized to fit within a unit cube centered at the origin. Camera poses are sampled uniformly from a spherical shell with radii $r \in [1.5, 2.2]$, oriented to face the object center. Lighting is similarly randomized for each view using an area light with varying energy and position. To ensure alignment across the dataset, we use a fixed random seed, resulting in identical camera trajectories and lighting conditions for every object.

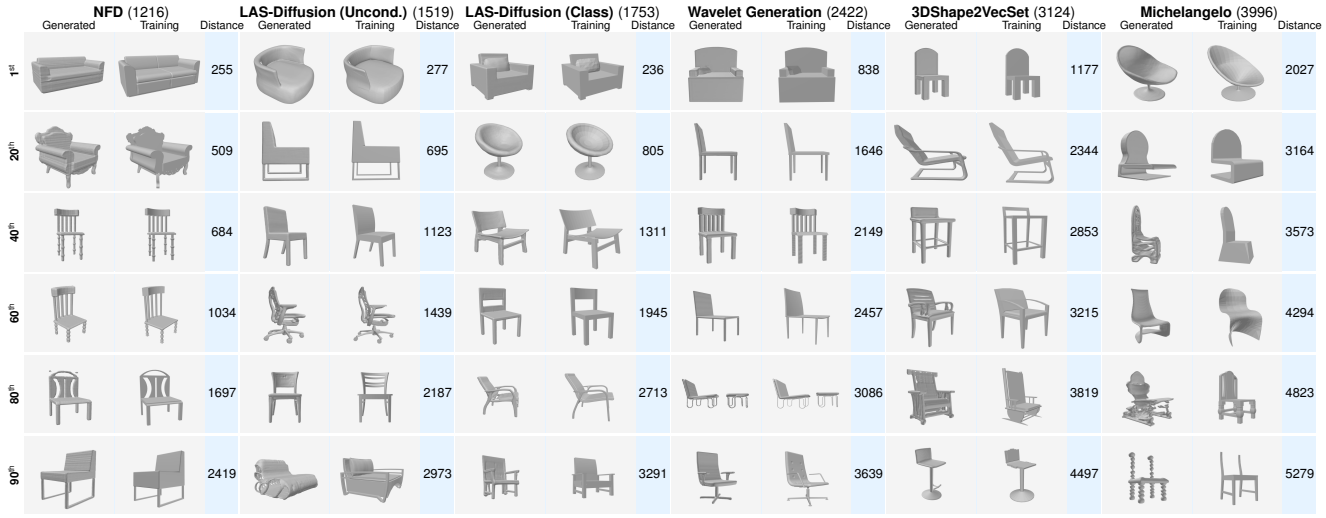


Figure 25. **Qualitative retrieval results on ShapeNet's *chair* category.** We generate 100 chairs for each model and visualize the nearest training shapes for generated samples at the 1^{st}, 20^{th}, 40^{th}, 60^{th}, 80^{th}, and 90^{th} percentiles of the LFD distance distribution for each model. NFD, unconditional LAS-Diffusion, and Wavelet Generation exhibit strong memorization: even generated shapes at the 60^{th}-80^{th} percentiles remain very close to their nearest training shapes. In contrast, conditional LAS-Diffusion, 3DShape2VecSet, and Michelangelo show novel geometric features even for generated samples with lower nearest-neighbor distances, indicating stronger generalization.

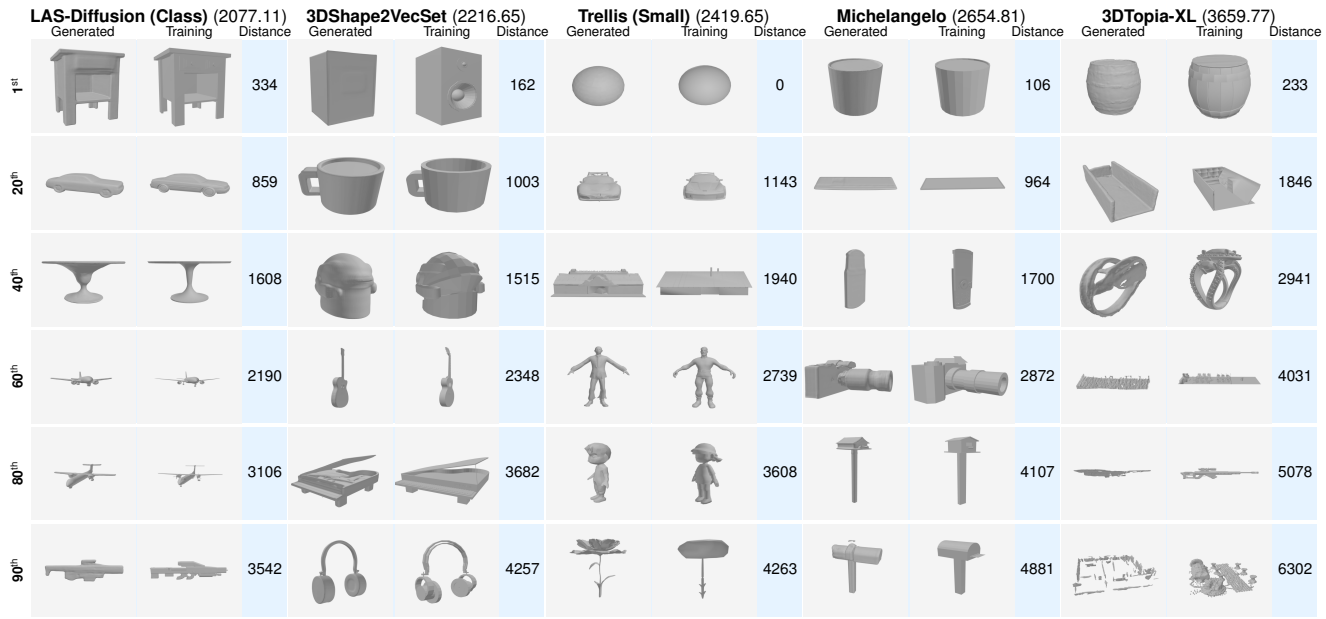


Figure 26. **Qualitative retrieval results on the entire training sets.** We generate 100 samples for each model and visualize the nearest training shapes for generated samples at the 1^{st}, 20^{th}, 40^{th}, 60^{th}, 80^{th}, and 90^{th} percentiles of the nearest-neighbor LFD distance distribution for each model trained on large datasets. Across all models, the retrieved training shapes already become novel at moderate percentiles (*e.g.*, 20^{th}-60^{th}). Although LFD is category-sensitive and may not always retrieve visually near-identical shapes, the overall trends suggest that these models trained on large datasets primarily generalize rather than copy individual training examples. We note that 3DTopia-XL does not generalize well and often degenerates, producing low-quality shapes even when using training prompts.