FOURIER LATENT FRAMEWORK FOR 3D POINT CLOUD GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generating 3D point clouds poses unique challenges due to their unordered nature and variable cardinality. Existing generative approaches often couple point cloud shape representations to the number of points, limiting their flexibility and scalability. We propose FourierDiT, a Fourier latent framework that decouples shape representation from point sampling. Our key idea is to represent a point cloud through its underlying shape distribution using empirical characteristic-function evaluations at a set of learnable spatial frequencies. The resulting Fourier latent representation is permutation-invariant by construction and has a fixed dimensionality independent of point-cloud cardinality. Building on this representation, we train a shape-conditioned vector-field decoder via flow matching, allowing an arbitrary number of points to be sampled from a single latent code. A frequency-conditioned Diffusion Transformer further learns the distribution of Fourier latent codes, enabling the synthesis of novel shapes. Extensive experiments demonstrate competitive generation quality and improved training efficiency, while reconstruction and interpolation results highlight the effectiveness of the learned Fourier latent space. Our framework provides a flexible and efficient approach to 3D point cloud generation without tying the point cloud representation to a fixed number of points.

1 INTRODUCTION

Point clouds are among the most natural and widely used representations for 3D shapes, arising directly from sensors such as LiDAR and depth cameras and serving as inputs to a broad range of downstream tasks in vision and graphics (Recht et al., 2019; Taha and Hanbury, 2015), robotics (Varley et al., 2017; Zhong et al., 2020), and autonomous driving (Chen et al., 2017; Yue et al., 2018). Unlike images or voxel grids, point clouds are unordered sets of points with variable cardinality, which makes generative modeling over them fundamentally different from generative modeling in Euclidean, grid-structured domains: a point cloud generator must respect permutation invariance, support a variable number of output points, and ideally produce shapes whose fidelity does not degrade when more or fewer points are sampled from the underlying surface.

A substantial body of work has explored 3D point cloud generation, beginning with sparse convolutional and point convolutional architectures and, more recently, through the lens of deep generative modeling (Ruthotto and Haber, 2021; Oussidi and Elhassouny, 2018; Achlioptas et al., 2018). Diffusion- and latent-diffusion-based methods (Ren et al., 2024; Zeng et al., 2022; Zhou et al., 2021) adapt generative frameworks to point sets or point-structured latent representations. While effective, many such methods operate on fixed-cardinality point clouds, with the size of the representation processed by the generative model tied to the number of points. As a result, exploiting denser point-cloud observations can increase the representation size and the associated computational cost. Autoregressive next-point prediction methods, often built on transformer architectures (Wu et al., 2022; 2024; Zhao et al., 2021), instead generate points sequentially. This imposes a fixed generation ordering on a set that has no intrinsic order, which can hinder generation quality. More recently, a novel autoregressive framework based on next-scale, level-of-detail prediction (Meng et al., 2026) generates points in a coarse-to-fine manner, but still operates with a prescribed output cardinality.

Across these approaches, a common challenge is to construct a shape representation that is both independent of point-cloud cardinality and suitable for flexible generative modeling. Methods operating directly on point sets often couple the modeled representation to the number of points, while latent approaches may retain point-structured representations or lack an explicit geometric interpretation. In this work, we introduce *FourierDiT*, a Fourier latent framework for 3D point cloud generation that addresses these limitations by encoding a point cloud's underlying probability distribution, rather than the point set itself, via a collection of characteristic-function evaluations. We summarize our contributions as follows:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

- We introduce a Fourier latent representation of 3D shape distributions based on empirical characteristic-function evaluations. The resulting representation is permutation invariant, independent of input point-cloud cardinality, and endowed with an explicit multi-frequency structure, allowing denser point clouds to provide more accurate empirical shape representations without increasing the latent dimensionality.
- We develop a two-stage generative framework that separates shape representation from point sampling. A generative model learns the distribution of Fourier shape latents, while a shape-conditioned flow-matching decoder realizes each latent as a continuous 3D distribution from which an arbitrary number of points can be generated.
- We demonstrate that the proposed framework achieves competitive point-cloud generation quality while substantially reducing training cost. We further study its cardinality scaling, reconstruction and interpolation behavior, and key design choices through extensive ablation experiments.

Concretely, given a set of frequency vectors, our approach represents a 3D shape distribution by the real and imaginary parts of its characteristic function evaluated at those frequencies. This *Fourier shape representation* has three properties that make it well suited for point cloud generation: **1. Permutation invariance:** Because the representation is defined as an integral (empirically, an average) over the point set, it is invariant to point ordering by construction, without requiring specialized symmetric architectures. **2. Cardinality independence:** The Fourier encoder maps input point clouds with an arbitrary number of points to a fixed-dimensional latent representation, while denser inputs provide more accurate empirical estimates of the underlying shape without increasing the latent dimensionality. The decoder can in turn generate an arbitrary number of points from the same latent code at sampling time. **3. Interpretability and multi-frequency structure:** Low-frequency components of the representation capture coarse, global geometry, while higher-frequency components capture fine-grained geometric detail, giving the latent code a simple and interpretable spectral structure rather than an opaque learned embedding.

Building on this representation, we design a two-stage generative framework combining a flow-matching decoder with a latent diffusion model. In Stage I, we jointly learn a Fourier latent representation and a shape-conditioned decoder. We encode each point cloud into a fixed-dimensional Fourier latent code using a fixed number of learnable frequency vectors, and train a latent-conditioned vector-field decoder via flow matching. Given a Fourier latent code, the decoder defines a continuous-time velocity field in $\mathbb{R}^3$ that transports particles from a simple noise distribution to the corresponding point distribution of the shape. Since all particles evolve under the same shared vector field, an arbitrary number of points can be generated from a single latent code. In Stage II, we learn a generative model over the Fourier latent space. Specifically, we train a Diffusion Transformer (DiT) (Peebles and Xie, 2023) to model the distribution of Fourier latent codes across shapes, enabling novel shape representations to be sampled directly. At generation time, the two stages are composed: a Fourier latent code is first sampled from the learned latent diffusion model and then decoded into a point cloud by evolving an arbitrary number of particles under its corresponding 3D vector field. This design cleanly separates shape representation from point sampling: the Fourier latent code specifies a global representation of the underlying 3D point distribution, while the decoder's vector field specifies how individual particles move toward that distribution. A generated point cloud is therefore a finite particle realization of the 3D point distribution represented by the latent, rather than a fixed, canonically ordered set of points.

The rest of the paper is organized as follows. Section 2 reviews related work and preliminaries. Section 3 presents our proposed method with the corresponding training and generation pipeline. Section 4 demonstrates its effectiveness through experiments. Finally, Section 5 concludes the paper and discusses its limitations and potential future directions.

2 Related Work and Preliminaries

Point cloud generation. 3D point cloud generation has advanced substantially through deep generative modeling. Continuous normalizing flows (Rezende and Mohamed, 2015; Chen et al., 2018) were among the first approaches applied to this task, with PointFlow (Yang et al., 2019) learning the latent distribution of point clouds via flow-based transformations. Subsequent work has largely shifted toward denoising diffusion probabilistic models (Ho et al., 2020; Song et al., 2021b), with approaches such as DPM (Luo and Hu, 2021), ShapeGF (Cai et al., 2020), PVD (Zhou et al., 2021), LION (Zeng et al., 2022), and TIGER (Ren et al., 2024) generating point clouds by gradually denoising either raw coordinates or latent representations in continuous space. To accelerate sampling, subsequent efforts have explored straight-flow formulations (Lipman et al., 2023; Wu et al., 2023) and ODE-based solvers (Song et al., 2021a).

Fourier and spectral approaches to point cloud generation. Another line of work related to ours concerns Fourier- and spectral-based approaches to point cloud generation. Unlike images on a regular grid, point clouds are irregularly

sampled and permutation-invariant, so Fourier and spectral methods do not apply directly (Luo and Hu, 2021). Several works nonetheless exploit frequency structure. Random Fourier features are widely used to embed coordinates or timesteps before a generative backbone (Tancik et al., 2020), giving the model direct access to high-frequency detail that networks otherwise struggle to learn due to spectral bias. Other methods generate directly in a spectral domain: Spectral-GAN (Ramasinghe et al., 2020) synthesizes point clouds via a spherical-harmonics representation that decouples model complexity from output resolution, and FrePolad (Zhou et al., 2024) adds spherical-harmonics frequency rectification to a latent diffusion pipeline while supporting variable cardinality. Still others leverage the fact that low- and high-frequency bands capture coarse shape and fine detail respectively, guiding a coarse-to-fine denoising process (Liu et al., 2026).

Probability paths, diffusion, and flow matching. Generative modeling can be viewed as constructing a probability path $\{p_t\}_{t\in[0,1]}$ between a simple reference distribution and a target data distribution. A continuous probability path can be associated with a time-dependent vector field u_t whose trajectories satisfy $\frac{dx_t}{dt} = u_t(x_t)$, and whose induced density evolves according to the corresponding continuity equation.

Diffusion models construct such a path through progressive Gaussian corruption of clean data. A commonly used parameterization is

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I),$$

where the signal-to-noise ratio decreases with t. Generation reverses this corruption process using a learned denoising model. In continuous time, the same marginal probability path also admits a deterministic probability-flow ODE representation (Song et al., 2021b).

Flow matching instead directly learns a vector field that realizes a prescribed probability path. Given a conditional path $p_t(x \mid x_1)$ with tractable conditional velocity $u_t(x \mid x_1)$, a parameterized field v_θ is trained by

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{t,x_1,x_t}\left[\|v_\theta(x_t,t) - u_t(x_t \mid x_1)\|_2^2\right],$$

with $x_t \sim p_t(\cdot \mid x_1)$ (Lipman et al., 2023). After training, samples are obtained by integrating the learned ODE from the reference distribution to the target. Additional related work and preliminaries are provided in Appendices A and B, respectively.

3 Method

Our framework separates point-cloud sampling from shape-level generative modeling by representing each shape in a fixed-dimensional Fourier latent space. In **Stage I**, we encode a point cloud through empirical characteristic-function evaluations at a set of learnable spatial frequencies, and jointly optimize these frequencies with a shape-conditioned decoder so that the resulting Fourier representation retains the geometric information needed for reconstruction. Because the representation has a fixed dimensionality independent of the number of input points, denser point clouds can be used to obtain a more accurate empirical estimate of the underlying shape distribution without increasing the latent sequence length or the training cost of the subsequent latent diffusion model. In **Stage II**, we train a standard diffusion transformer (DiT) in this Fourier latent space. Each Fourier coefficient is paired with its associated spatial frequency, providing the DiT with the spectral identity of each latent token while otherwise retaining the standard diffusion formulation. An overview of the framework is shown in Figure 1.

3.1 Stage I: Learning the Fourier Latent Representation and Decoder

3.1.1 Fourier Latent Encoder

Let μ denote the underlying probability distribution of a 3D shape on $\mathbb{R}^3$. Given a set of frequency vectors

$$K = \{k_m\}_{m=1}^M \in \mathbb{R}^{M\times 3},$$

we define the Fourier shape representation of μ as

$$\Phi_K(\mu) = [\Phi_1(\mu), \ldots, \Phi_M(\mu)]^\top \in \mathbb{R}^{M\times 2},$$

where

$$\Phi_m(\mu) = \begin{bmatrix} \mathrm{Re}\,\varphi_\mu(k_m) \\ \mathrm{Im}\,\varphi_\mu(k_m) \end{bmatrix} = \begin{bmatrix} \int_{\mathbb{R}^3} \cos(k_m^\top x)\, d\mu(x) \\ \int_{\mathbb{R}^3} \sin(k_m^\top x)\, d\mu(x) \end{bmatrix},$$

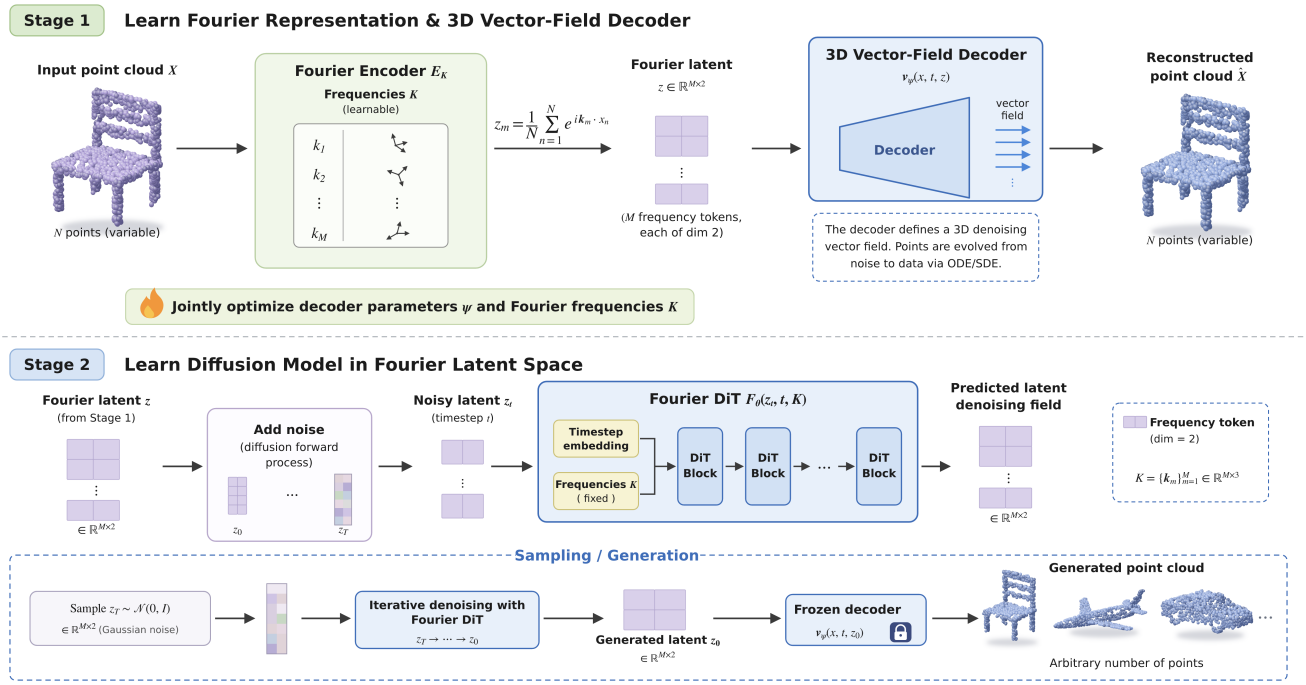


Figure 1: Overview of the proposed Fourier latent framework. In Stage I, a point cloud is encoded into a fixed-dimensional Fourier latent representation, which conditions a flow-matching decoder in continuous 3D space. In Stage II, a DiT model learns the distribution of Fourier latent codes.

and

$$\varphi_\mu(k) = \int_{\mathbb{R}^3} e^{ik^\top x}\, d\mu(x)$$

denotes the characteristic function of μ. We denote the resulting Fourier latent code by $z := \Phi_K(\mu)$.

The Fourier latent code is therefore a finite collection of characteristic-function evaluations that describes the shape distribution through its responses at a set of spatial frequencies. Low-frequency components characterize coarse, global geometric structure, whereas higher-frequency components capture finer geometric variations. The number of frequencies M controls both the dimensionality and the expressiveness of the latent representation.

Since the latent code consists of only M characteristic-function evaluations, its ability to represent shape geometry depends on where the characteristic function is sampled in frequency space. We therefore treat $K = \{k_m\}_{m=1}^{M}$ as learnable parameters and optimize them jointly with the shape-conditioned decoder, allowing the frequency locations to adapt to the geometric information most relevant for reconstructing the underlying shapes.

Empirical Fourier Encoder. In practice, the underlying distribution μ is not directly observed. Instead, we observe a finite point cloud $X_N = \{x_j\}_{j=1}^{N} \in \mathbb{R}^{N\times 3}$, which induces the empirical measure $\hat{\mu}_N := \frac{1}{N}\sum_{j=1}^{N} \delta_{x_j}$.

The Fourier encoder E_K maps X_N to its empirical Fourier representation

$$\mathrm{E}_K(X_N) := \Phi_K(\hat{\mu}_N) \in \mathbb{R}^{M\times 2},$$

where

$$\Phi_m(\hat{\mu}_N) = \frac{1}{N}\sum_{j=1}^{N} \begin{bmatrix} \cos(k_m^\top x_j) \\ \sin(k_m^\top x_j) \end{bmatrix}.$$

We denote the resulting empirical Fourier latent code by

$$z_N := \mathrm{E}_K(X_N) = \Phi_K(\hat{\mu}_N).$$

This representation has two properties that make it particularly suitable for point-cloud data. First, z_N is *permutation invariant* by construction, since it depends on the point cloud only through its empirical measure $\hat{\mu}_N$. Second, by the

law of large numbers, the empirical Fourier representation converges to its population counterpart:

$$\mathrm{E}_K(X_N) = \Phi_K(\hat{\mu}_N) \longrightarrow \Phi_K(\mu) \qquad \text{as } N \to \infty.$$

Crucially, the dimensionality of z_N is $M \times 2$, independent of the input cardinality N. Thus, the same encoder can process point clouds with different numbers of points,

$$X_{2048},\ X_{8192},\ X_{15000} \quad \longmapsto \quad z_N \in \mathbb{R}^{M\times 2},$$

without changing the dimensionality or structure of the latent space. Increasing N provides a more accurate empirical estimate of the underlying shape distribution, while the Fourier latent remains in $\mathbb{R}^{M\times 2}$. Thus, denser point clouds can be used to construct more reliable shape representations without increasing either the latent dimensionality or the sequence length processed by the subsequent latent diffusion model.

3.1.2 Shape-Latent-Conditioned Vector Field Decoder

Given a Fourier latent code $z \in \mathcal{Z}$, we model the decoder as a shape-conditioned, time-dependent vector field

$$v_\psi : \mathbb{R}^3 \times [0,1] \times \mathcal{Z} \to \mathbb{R}^3.$$

For a fixed latent code z, the conditional field $v_\psi^z(x,t) := v_\psi(x,t,z)$ defines a continuous transport from a Gaussian reference distribution toward the point distribution of the shape represented by z. In this way, the Fourier latent specifies the shape, while the decoder determines how individual particles are transported to realize that shape in 3D space.

Given z, points are generated independently by sampling $x_{0,j} \sim \mathcal{N}(0, I_3)$ and evolving $\frac{dx_{t,j}}{dt} = v_\psi(x_{t,j}, t, z)$. All particles share the same shape-conditioned vector field but do not interact with one another. The decoder therefore models a distribution over 3D points conditioned on the Fourier shape representation, allowing the output cardinality to be chosen freely at inference time.

We train the decoder with conditional flow matching. Let $p_t(x \mid x_1)$ denote a conditional probability path from the reference distribution to a target point x_1, with corresponding conditional velocity field $u_t(x \mid x_1)$. For a training point cloud $X_N = \{x_j\}_{j=1}^N$, we first compute the shape-level Fourier representation

$$z_N = \Phi_K(\hat{\mu}_N)$$

using all N input points. While the flow-matching target is defined for individual points, the predicted vector field is conditioned on this representation of the entire shape. We jointly optimize ψ and K using

$$\mathcal{L}_{\text{decoder}}(\psi, K) = \mathbb{E}_{\substack{X_N\sim\mathcal{D},\, x_1\sim\hat{\mu}_N,\\ t\sim\mathrm{Unif}[0,1],\, x_t\sim p_t(\cdot|x_1)}} \left[\|v_\psi(x_t, t, z_N) - u_t(x_t \mid x_1)\|_2^2 \right]. \tag{1}$$

Since z_N is differentiable with respect to K, gradients from the flow-matching objective propagate through the Fourier encoder to the frequency locations. The decoder and frequency set are therefore learned jointly, encouraging the selected frequencies to capture the geometric information most useful for decoding the underlying shapes. Once trained, the resulting shape-conditioned vector field can transport any number of independently initialized particles toward the distribution represented by a given Fourier latent code. The complete Stage-I training procedure is summarized in Algorithm 1.

3.2 Stage II: Frequency-Conditioned Latent Diffusion

In Stage II, we learn the distribution of Fourier shape representations obtained from Stage I using a Diffusion Transformer (DiT). The learned frequency set $K = \{k_m\}_{m=1}^M$ is kept fixed throughout this stage. For each training point cloud, the Fourier encoder produces a latent sequence $z = \{z_m\}_{m=1}^M \in \mathbb{R}^{M\times 2}$, where

$$z_m = \begin{bmatrix} \mathrm{Re}\, \varphi_{\hat{\mu}}(k_m) \\ \mathrm{Im}\, \varphi_{\hat{\mu}}(k_m) \end{bmatrix},$$

so that each Fourier coefficient z_m is naturally associated with a spatial frequency $k_m \in \mathbb{R}^3$. We apply a standard diffusion process to the Fourier latent z and use a DiT to model its distribution across shapes.

Frequency-conditioned Fourier tokens. The value of each Fourier coefficient is meaningful relative to the frequency at which the characteristic function is evaluated. Moreover, the continuity of the characteristic function induces local structure in frequency space: evaluations at nearby frequency locations are locally related. We therefore incorporate the corresponding spatial frequency into each Fourier token, allowing the DiT to model dependencies among coefficients with respect to their locations in frequency space.

At diffusion timestep t, let $z_{t,m} \in \mathbb{R}^2$ denote the noisy version of the m-th Fourier coefficient. We concatenate it with its associated spatial frequency and construct the input token as

$$h_{t,m} = \mathrm{Embed}\left([z_{t,m}; k_m]\right), \qquad m = 1, \ldots, M. \tag{2}$$

The resulting sequence $\{h_{t,m}\}_{m=1}^{M}$ is processed by standard DiT blocks, with the diffusion timestep supplied through the usual timestep conditioning.

Importantly, incorporating k_m into the token representation also makes the self-attention interactions frequency aware. For tokens m and n, the standard attention score is computed as

$$s_{mn} = \frac{q_m^\top \kappa_n}{\sqrt{d}}, \qquad q_m = W_Q h_{t,m}, \qquad \kappa_n = W_K h_{t,n},$$

where κ_n denotes the attention key and is distinguished from the spatial frequency k_n. Since both the query and key are derived from embeddings containing their corresponding frequency coordinates, the attention weights can depend on relationships between Fourier coefficients in frequency space. This allows the DiT to exploit local dependencies among nearby frequencies when useful, while retaining the flexibility to model long-range interactions across the spectrum.

Apart from this frequency-conditioned token construction, we follow the standard diffusion training and sampling procedure. Given a clean Fourier latent z, the forward diffusion process produces a noisy latent z_t, and the DiT is trained using the standard v-prediction objective to recover the corresponding diffusion target from z_t and the diffusion timestep. The complete Stage-II training procedure is summarized in Algorithm 2. At inference time, reverse diffusion generates a new Fourier latent code, which is subsequently decoded into a 3D point cloud using the shape-conditioned vector field learned in Stage I.

3.3 Generation

At inference time, generation composes the latent diffusion model with the shape-conditioned decoder learned in Stage I. We first sample a Fourier latent representation using the standard reverse diffusion process. Starting from Gaussian noise $z_T \sim \mathcal{N}(0, I_{2M})$, the frequency-conditioned DiT iteratively denoises the latent, with each noisy Fourier coefficient paired with its fixed spatial frequency k_m, to obtain a generated latent $z^\star \in \mathbb{R}^{M\times 2}$. The resulting $z^\star$ represents a generated shape in the learned Fourier latent space.

We then realize this shape as a point cloud using the shape-conditioned vector-field decoder. Given a desired output cardinality n, we independently initialize $x_{0,j} \sim \mathcal{N}(0, I_3)$ for $j = 1, \ldots, n$ and evolve each particle according to $\frac{dx_{t,j}}{dt} = v_\psi(x_{t,j}, t, z^\star)$ for $t \in [0, 1]$. Integrating the decoder dynamics yields $\hat{X}_n = \{x_{1,j}\}_{j=1}^{n}$.

The target cardinality n enters only through the number of particles initialized for decoding and does not affect the Fourier latent representation or the latent diffusion model. Consequently, the same generated latent $z^\star$ can be realized at different point-cloud cardinalities without retraining either stage. Conceptually, the latent diffusion model generates a shape-level Fourier representation, while the decoder converts this representation into point samples in continuous 3D space. The complete generation procedure is summarized in Algorithm 3.

4 Experiments

We evaluate FourierDiT on unconditional 3D point cloud generation, examining generation quality, training efficiency, cardinality scaling, latent-space properties, and key design choices.

4.1 Experimental Setup

Datasets and metrics. We evaluate FourierDiT on ShapeNetv2 using the airplane, chair, and car categories. We generate and evaluate point clouds with $2{,}048$ points. Unless otherwise specified, we use $M = 128$ Fourier tokens. Generation quality is measured by 1-nearest-neighbor accuracy (1-NNA) under Chamfer Distance (CD) and Earth Mover's Distance (EMD), where scores closer to 50% indicate better distribution matching.

Table 1: Performance under the standard 2,048-point setup on ShapeNet using two test-set protocols: the top block corresponds to the conventional random split, and the bottom block corresponds to the LION split. Category-wise baseline results are quoted from PointNSP, while mean results are computed over the three categories. All values are 1-NNA scores (%), where values closer to 50% indicate better distribution matching. Best results are in bold and second-best results are underlined within each block.

Method	Type	Airplane		Chair		Car		Mean	
		CD	EMD	CD	EMD	CD	EMD	CD	EMD
ShapeGF	Diffusion	80.00	76.17	68.96	65.48	63.20	56.53	70.72	66.06
DPM	Diffusion	76.42	86.91	60.05	74.77	68.89	79.97	68.45	80.55
PVD	Diffusion	73.82	64.81	56.26	53.32	54.55	53.83	61.54	57.32
LION	Diffusion	72.99	64.21	55.67	53.82	53.47	<u>53.21</u>	60.71	<u>57.08</u>
TIGER	Diffusion	73.02	64.10	55.15	53.18	<u>53.21</u>	53.95	60.46	<u>57.08</u>
PointGrow	Autoregressive	82.20	78.54	63.14	61.87	67.56	65.89	70.97	68.77
CanonicalVAE	Autoregressive	80.15	76.27	62.78	61.05	63.23	61.56	68.72	66.29
PointGPT	Autoregressive	74.85	65.61	57.24	55.01	55.91	54.24	62.67	58.29
PointNSP	Autoregressive	<u>72.24</u>	<u>63.69</u>	<u>54.54</u>	<u>52.85</u>	**52.17**	**51.85**	<u>59.65</u>	**56.13**
FourierDiT	Diffusion	**64.20**	**61.85**	**54.31**	**52.57**	54.69	53.98	**57.73**	**56.13**
LION	Diffusion	67.41	61.23	**53.70**	52.34	53.41	51.14	58.17	54.90
TIGER	Diffusion	67.21	<u>56.26</u>	54.32	<u>51.71</u>	54.12	<u>50.24</u>	58.55	<u>52.74</u>
PointNSP	Autoregressive	<u>66.98</u>	**56.05**	<u>54.01</u>	**51.05**	<u>53.12</u>	**50.09**	<u>58.04</u>	**52.40**
FourierDiT	Diffusion	**63.95**	59.26	54.61	53.10	**52.98**	52.13	**57.18**	54.83

Baselines and protocol. We compare FourierDiT with ShapeGF (Cai et al., 2020), DPM (Luo and Hu, 2021), PVD (Zhou et al., 2021), LION (Zeng et al., 2022), TIGER (Ren et al., 2024), PointGrow (Sun et al., 2020), CanonicalVAE (Cheng et al., 2022), PointGPT (Chen et al., 2023), and PointNSP (Meng et al., 2026). Evaluation is conducted using the test sets defined by the conventional random split and the LION split. For each category, we generate the same number of samples as in the corresponding test set.

4.2 Quantitative Generation Results

Generation comparison. Tab. 1 reports results under the standard 2,048-point setting. On the conventional random split, FourierDiT achieves the best mean CD-based 1-NNA (57.73%) and matches the best mean EMD-based score (56.13%). It also obtains the best CD- and EMD-based results on airplane and chair. On the LION split, FourierDiT again achieves the best mean CD-based 1-NNA (57.18%), including the best CD scores on airplane and car. Its EMD performance remains competitive, although PointNSP and TIGER achieve lower mean scores. Overall, FourierDiT provides strong distribution matching across both test-set protocols.

Qualitative comparison. Fig. 2 presents qualitative comparisons of point clouds generated by different methods across the three categories. FourierDiT produces coherent shapes with recognizable global structures and characteristic local geometry.

4.3 Training efficiency and Effect of training cardinality

Tab. 2 compares training costs at 2,048 and 8,192 points. FourierDiT requires only 48 GPU hours in both settings, substantially less than the reported costs of competing methods. Its training cost remains nearly unchanged as the input cardinality increases, demonstrating its efficiency in scaling to larger point clouds. We further investigate the effect of training cardinality on airplane by varying the number of training points N, while keeping the model architecture and Fourier latent dimensionality unchanged. As shown in Tab. 3, increasing N from 2,048 to 15,000 reduces CD-based 1-NNA from 69.38% to 64.20% and EMD-based 1-NNA from 64.44% to 61.85%.

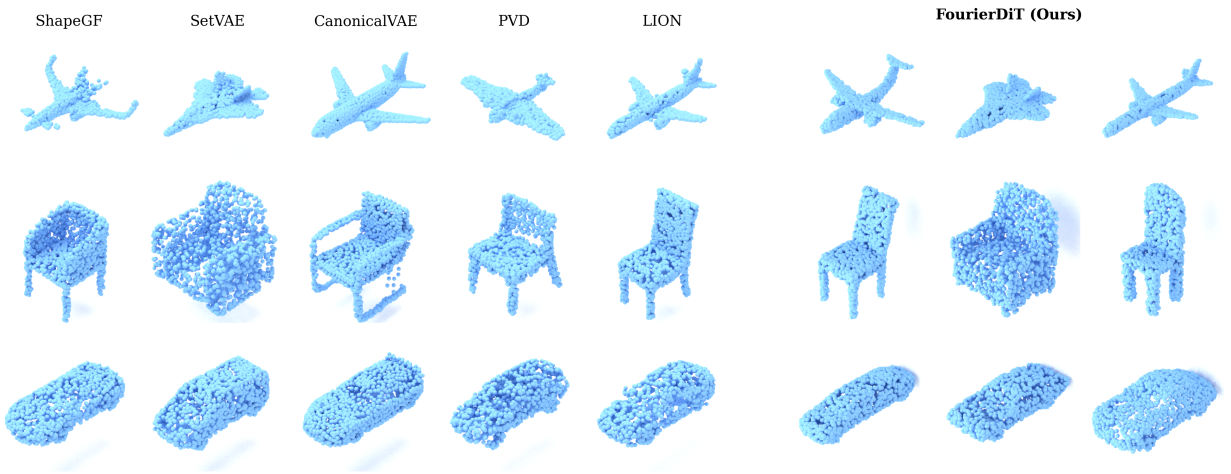


Figure 2: Qualitative comparison of generated point clouds across different methods. FourierDiT generates coherent shapes with well-preserved global structure and local geometry.

Table 2: Training efficiency under the 2,048-point and 8,192-point settings. Training time is reported in GPU hours and averaged over the three categories.

Method	2,048 pts ↓	8,192 pts ↓
PointGPT	185	296
PVD	142	201
LION	550	610
TIGER	164	320
PointNSP	178	190
FourierDiT	48	48

Table 3: Effect of the number of training points N on the Airplane category. All values are 1-NNA scores (%), where values closer to 50% indicate better distribution matching.

N	CD	EMD
2,048	69.38	64.44
8,192	67.28	64.69
15,000	64.20	61.85

4.4 Analysis of the Fourier Latent Space

Reconstruction. Fig. 3 shows point cloud reconstructions obtained by encoding input shapes into Fourier latent codes and decoding them with the learned vector field. The reconstructed point clouds preserve global shape structure and characteristic local geometry, demonstrating the representational capacity of the learned Fourier latent space.

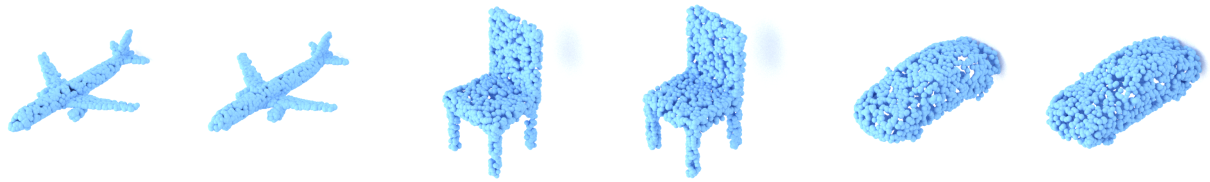
Latent interpolation. We perform linear interpolation between two encoded Fourier latent representations,

$$z_\alpha = (1-\alpha)z_a + \alpha z_b, \qquad \alpha \in [0,1].$$

As shown in Fig. 4, the decoded shapes exhibit smooth geometric transitions, suggesting that the learned decoder effectively captures the relationship between Fourier latent representations and the underlying shape geometry.

4.5 Ablation Studies

Learnable frequencies and frequency conditioning. Tab. 4 evaluates two key design choices of FourierDiT. Fixing the Fourier frequencies increases the mean CD- and EMD-based 1-NNA from 57.73% and 56.13% to 60.98% and 57.72%, respectively, demonstrating the benefit of jointly optimizing the frequency vectors and the decoder. Separately, removing frequency conditioning from the latent DiT increases the corresponding scores to 59.33% and 58.84%. This suggests that providing each Fourier coefficient with its associated spatial frequency helps the DiT model the relationships among coefficients at different frequencies.



(a) Airplane (b) Chair (c) Car

Figure 3: Point cloud reconstruction from Fourier latent codes. Each pair shows the input point cloud and its reconstruction. The reconstructed shapes preserve both global structure and characteristic local geometry.



Figure 4: Linear interpolation between encoded Fourier latent codes. Intermediate codes decode into coherent shapes with smooth geometric transitions.

Table 4: Ablation studies of FourierDiT on ShapeNet. All values are 1-NNA scores (%). Mean results are averaged over the Airplane, Chair, and Car categories. Numbers in parentheses denote the change relative to the full model.

Variant	Airplane		Chair		Car		Mean	
	CD	EMD	CD	EMD	CD	EMD	CD	EMD
Fixed Fourier frequencies	67.16 (+2.96)	61.98 (+0.13)	57.40 (+3.09)	55.51 (+2.94)	58.38 (+3.69)	55.68 (+1.70)	60.98 (+3.25)	57.72 (+1.59)
w/o frequency-conditioned DiT	65.31 (+1.11)	63.46 (+1.61)	56.42 (+2.11)	57.25 (+4.68)	56.25 (+1.56)	55.82 (+1.84)	59.33 (+1.60)	58.84 (+2.71)
Full FourierDiT	**64.20**	**61.85**	**54.31**	**52.57**	**54.69**	**53.98**	**57.73**	**56.13**

Number of Fourier tokens. Tab. 5 investigates the effect of the number of Fourier tokens M on airplane. Increasing M from 64 to 128 improves both metrics, while further increases yield marginal and non-monotonic improvements. Although $M = 256$ achieves the lowest EMD score and $M = 512$ the lowest CD score, the training cost increases substantially, reaching $4.22\times$ that of $M = 128$. We therefore adopt $M = 128$ as the default setting, balancing generation quality and training efficiency.

Table 5: Effect of the number of Fourier tokens M on the Airplane category. All values are 1-NNA scores (%).

M	CD	EMD	Training Cost
64	66.42	62.35	$0.51\times$
128	64.20	61.85	$1.00\times$
256	64.32	61.36	$2.05\times$
512	63.70	62.22	$4.22\times$

5 Conclusion

We introduced FourierDiT, a two-stage generative framework for 3D point clouds that separates shape representation from point sampling. Point clouds are encoded as fixed-dimensional Fourier latents through characteristic-function evaluations at learnable spatial frequencies. A shape-conditioned flow-matching decoder maps each latent to a continuous 3D point distribution, while a frequency-conditioned diffusion transformer models the latent distribution across shapes. This design yields a permutation-invariant and cardinality-independent shape representation.

A limitation of our approach is that the latent code retains only M evaluations of the characteristic function, so geometric detail whose spectral content lies outside the selected frequency set cannot be recovered by the decoder. Future work includes characterizing the approximation error as a function of the number and choice of frequencies, and extending the framework to conditional generation settings such as text- or image-guided synthesis.

Reproducibility Statement

An anonymized codebase for our training and generation pipeline is submitted with the paper, including the implementation and configurations used for training and evaluation. All experiments use publicly available data, and the corresponding data preprocessing and experimental settings are provided in the paper and code.

AI Use Statement

In this work, we used generative AI tools to assist with code development and debugging. We did not use generative AI tools to develop theoretical models or conceptual frameworks, formulate or prove mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean or reformat datasets, or interpret experimental results; the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for literature search, identification of relevant prior work, and improving the clarity and presentation of the manuscript. We reviewed all AI-assisted work: AI-assisted code was tested and verified for correctness by the authors, and suggested references were checked against the original sources. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

References

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 40–49. PMLR, 2018.

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 364–381. Springer International Publishing, 2020.

Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. PointGPT: Auto-regressively generative pre-training from point clouds. *Advances in Neural Information Processing Systems*, 36:29667–29679, 2023.

Ricky TQ Chen and Yaron Lipman. Flow matching on general geometries. In *International Conference on Learning Representations*, volume 2024, pages 47922–47945, 2024.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31, 2018.

Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6526–6534. IEEE, 2017.

An-Chieh Cheng, Xueting Li, Sifei Liu, Min Sun, and Ming-Hsuan Yang. Autoregressive 3d shape generation via canonical mapping. In *European Conference on Computer Vision*, pages 89–104. Springer, 2022.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

Hangfeng He and Weijie J Su. A law of next-token prediction in large language models. *Physical Review E*, 112(3): 035317, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. In *Proceedings of the International Conference on Learning Representations (ICLR)*, volume 2025, pages 23378–23402, 2025.

Chieh-Hsin Lai, Yang Song, Dongjun Kim, Yuki Mitsufuji, and Stefano Ermon. The principles of diffusion models. *arXiv preprint arXiv:2510.21890*, 2025.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37:56424–56445, 2024.

Yaron Lipman, Ricky T Q Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Jiaxu Liu, Li Li, Hubert PH Shum, and Toby P Breckon. TFDM: Time-variant frequency-based point cloud diffusion with state space model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 4667–4676, 2026.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2023.

Zhijun Liu, Shuai Wang, Sho Inoue, Qibing Bai, and Haizhou Li. Autoregressive diffusion transformer for text-to-speech synthesis. *arXiv preprint arXiv:2406.05551*, 2024.

Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

Shitong Luo and Wei Hu. Diffusion probabilistic models for 3D point cloud generation. In *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2836–2844. IEEE, 2021.

Ziqiao Meng, Qichao Wang, Zhiyang Dou, Zixing Song, Zhipeng Zhou, Irwin King, and Peilin Zhao. PointNSP: Autoregressive 3D point cloud generation with next-scale level-of-detail prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 31450–31461, 2026.

Achraf Oussidi and Azeddine Elhassouny. Deep generative models: Survey. In *Proceedings of the 2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–8. IEEE, 2018.

Ajay Patel, Bryan Li, Mohammad Sadegh Rasooli, Noah Constant, Colin Raffel, and Chris Callison-Burch. Bidirectional language models are also few-shot learners. *arXiv preprint arXiv:2209.14500*, 2022.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4172–4182. IEEE, 2023.

Sameera Ramasinghe, Salman Khan, Nick Barnes, and Stephen Gould. Spectral-GANs for high-resolution 3D point-cloud generation. In *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8169–8176. IEEE, 2020.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 5389–5400. PMLR, 2019.

Zhiyuan Ren, Minchul Kim, Feng Liu, and Xiaoming Liu. TIGER: Time-varying denoising model for 3D point cloud generation with diffusion process. In *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9462–9471. IEEE, 2024.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 1530–1538. PMLR, 2015.

Lars Ruthotto and Eldad Haber. An introduction to deep generative modeling. *GAMM-Mitteilungen*, 44(2): e202100008, 2021.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021b.

Maojiang Su, Mingcheng Lu, Jerry Yao-Chieh Hu, Shang Wu, Zhao Song, Alex Reneau, and Han Liu. A theoretical analysis of discrete flow matching generative models. *arXiv preprint arXiv:2509.22623*, 2025.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Yongbin Sun, Yue Wang, Ziwei Liu, Joshua Siegel, and Sanjay Sarma. PointGrow: Autoregressively learned point cloud generation with self-attention. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 61–70, 2020.

Abdel Aziz Taha and Allan Hanbury. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Medical Imaging*, 15(1):29, 2015.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in Neural Information Processing Systems*, 37:84839–84865, 2024.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

Rafael Valle, Kevin Shih, Ryan Prenger, and Bryan Catanzaro. Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. *arXiv preprint arXiv:2005.05957*, 2020.

Jacob Varley, Chad DeChant, Adam Richardson, Joaquín Ruales, and Peter Allen. Shape completion enabled robotic grasping. In *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2442–2447. IEEE, 2017.

Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

Lemeng Wu, Dilin Wang, Chengyue Gong, Xingchao Liu, Yunyang Xiong, Rakesh Ranjan, Raghuraman Krishnamoorthi, Vikas Chandra, and Qiang Liu. Fast point cloud generation with straight flows. In *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9445–9454. IEEE, 2023.

Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer V2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer V3: Simpler, faster, stronger. In *Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4840–4851. IEEE, 2024.

Desai Xie, Zhan Xu, Yicong Hong, Hao Tan, Difan Liu, Feng Liu, Arie Kaufman, and Yang Zhou. Progressive autoregressive video diffusion models. In *Proceedings of the 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 6312–6322. IEEE, 2025.

Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. PointFlow: 3D point cloud generation with continuous normalizing flows. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4540–4549. IEEE, 2019.

Xiangyu Yue, Bichen Wu, Sanjit A Seshia, Kurt Keutzer, and Alberto L Sangiovanni-Vincentelli. A LiDAR point cloud generator: from a virtual world to autonomous driving. In *Proceedings of the 2018 ACM International Conference on Multimedia Retrieval*, pages 458–464. ACM, 2018.

Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. LION: Latent point diffusion models for 3D shape generation. In *Advances in Neural Information Processing Systems*, volume 35, pages 10021–10039, 2022.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16259–16268, 2021.

Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pages 42390–42402. PMLR, 2023.

Boxuan Zhong, He Huang, and Edgar Lobaton. Reliable vision-based grasping target recognition for upper limb prostheses. *IEEE transactions on cybernetics*, 52(3):1750–1762, 2020.

Chenliang Zhou, Fangcheng Zhong, Param Hanji, Zhilin Guo, Kyle Fogarty, Alejandro Sztrajman, Hongyun Gao, and Cengiz Oztireli. FrePolad: Frequency-rectified point latent diffusion for point cloud generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 434–453. Springer Nature Switzerland, 2024.

Linqi Zhou, Yilun Du, and Jiajun Wu. 3D shape generation and completion through point-voxel diffusion. In *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5806–5815. IEEE, 2021.

Zhengyu Zhou and Weiwei Liu. An error analysis of flow matching for deep generative modeling. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

A Additional Related Work

Autoregressive generative modeling. Another common but comparatively less-explored paradigm is autoregressive generative modeling for 3D point cloud generation (Sun et al., 2020; Chen et al., 2023; Meng et al., 2026). The core principle of autoregressive generative models is to synthesize outputs sequentially, with each new segment generated conditioned on those already produced. This principle has driven remarkable success in language modeling via next-token prediction (Patel et al., 2022; Wang et al., 2024; He and Su, 2025), and has since been extended to a variety of other modalities, including speech (Valle et al., 2020; Liu et al., 2024), images (Sun et al., 2024; Li et al., 2024; Tian et al., 2024), and videos (Xie et al., 2025). In practice, these modalities are often continuous in nature and must first be transformed into discrete token representations, with generation subsequently performed over the resulting tokens in a predefined raster-scan order.

Continuous normalizing flow. CNFs arise from viewing the transport map as the solution of an ODE, but remained impractical at scale until the introduction of flow matching, whose key idea is to learn the constructed velocity fields through deep least-squares regression rather than ODE simulation. Several variants differ in how the interpolant and probability path are constructed: rectified flow (Liu et al., 2023) proposes a linear interpolant emphasizing straight paths to improve sampling efficiency, Lipman et al. (2023) build the interpolant from conditional probability paths connecting to a Gaussian, Tong et al. (2023) introduce a simulation-free objective conditioned on a general distribution and show that lifting static optimal transport to the dynamic setting lowers objective variance and simplifies flows, and Chen and Lipman (2024) extend flow matching to the Riemannian setting. A related line of work (Albergo and Vanden-Eijnden, 2023; Albergo et al., 2025; Lipman et al., 2023) notes that the probability path of CNFs encompasses that of diffusion models (Song et al., 2021b; Ho et al., 2020; Song et al., 2021a), and that ODE-based methods offer advantages absent for SDEs, including an exact, tractable likelihood formula, adaptive integration schemes for sampling, and compatibility with distillation (Liu et al., 2023; Salimans and Ho, 2022; Zheng et al., 2023; Luhman and Luhman, 2021), which requires deterministic samplers.

B Additional Preliminaries on flow-based generative model

Continuous normalizing flows. Continuous normalizing flows (CNFs) extend classical normalizing flows, which are discrete in time, to the continuous-time setting. A CNF generates samples from a target distribution by continuously deforming samples drawn from a source distribution. The transformation is described by an invertible continuous-time flow map

$$\phi_t(x) \doteq \phi(t, x) : [0, 1] \times \mathbb{R}^d \to \mathbb{R}^d,$$

where $t \in [0, 1]$ indexes time and $x \in \mathbb{R}^d$ is the initial point, with $\phi_0(x) = x$ and $\phi_1(x)$ the endpoint. This flow is the solution of an ordinary differential equation driven by a velocity field $u_t(x)$:

$$\mathrm{d}\phi_t(x) = u_t(\phi_t(x))\,\mathrm{d}t.$$

Setting $X_0 \sim p$ and $X_t \doteq \phi_t(X_0) \sim p_t$, the induced probability path $\{p_t\}_{t=0}^1$ follows from the change-of-variables formula

$$p_t(x) = p_0\big(\phi_t^{-1}(x)\big)\left|\det \partial_x \phi_t^{-1}(x)\right|.$$

Because this yields a closed-form expression for $p_1(x)$, early CNF approaches parameterize $u_t(x)$ with a neural network and train via maximum likelihood estimation. However, this requires solving the ODE exactly and differentiating through the solution at every iteration, which is both computationally expensive and prone to instability, motivating the simulation-free flow matching framework discussed next.

Flow matching. To avoid the per-iteration ODE solving required by maximum-likelihood training of CNFs, subsequent work developed the flow matching (FM) framework (Lipman et al., 2023; Dao et al., 2023; Jin et al., 2025; Zhou and Liu, 2025; Su et al., 2025). Rather than maximizing likelihood, FM directly regresses a parameterized vector field u_t^θ onto a prescribed target velocity field u_t that induces a flow transporting p_0 into p_1. The flow map $\phi_t(x)$ can be expressed as the solution to an ordinary differential equation driven by a velocity field $u_t : \mathbb{R}^d \to \mathbb{R}^d$ for $t \in [0, 1]$

$$\frac{d\phi_t(x)}{dt} = u_t(\phi_t(x)) \tag{3}$$

with some initial condition $\phi_0(x)$. The initial condition is a random variable X_0 drawn from some distribution p_0. This naturally defines a trajectory $X_t = \phi_t(X_0) \sim p_t$ as the solution to the ODE

$$\frac{dX_t}{dt} = u_t(X_t). \tag{4}$$

In flow matching, the goal is to learn the velocity field u_t via some neural network so that it induces a probability path from p_0 to p_1 via $\{p_t\}_{t=0}^1$. More precisely, we want to minimize the loss function

$$\min_\theta \mathbb{E}_{t\sim[0,1],X_t\sim p_t}\|u_t^\theta(X_t) - u_t(X_t)\|^2 \tag{5}$$

empirically on some finite samples. However, training via (5) is rarely used in practice, as the velocity field u_t is often intractable in practice. Instead, one would consider the conditional flow matching training objective

$$\min_\theta \mathbb{E}_{t\sim[0,1],X_1\sim p_1,X_t\sim p_t(\cdot|X_1)}\|u_t^\theta(X_t) - u_t(X_t|X_1)\|^2, \tag{6}$$

where $p(\cdot|X_1)$ is the conditional probability path associated with the conditional velocity field $u_t(\cdot|X_1)$, which is often tractable. It is worth noting that one can easily show that the loss in (5) is equivalent to the loss in (6) up to an additive constant independent of the neural network training parameters θ (Lipman et al., 2024).

One popular way of designing the probability path $\{p_t\}_{t=0}^1$ is via linear interpolation between p_0 and p_1:

$$X_t = (1-t)X_0 + tX_1 \tag{7}$$

for $t \in [0,1]$. This leads to linear conditional flow (Lipman et al., 2024). Under this, the conditional velocity field is simply $u_t(\cdot|X_1) = X_1 - X_0$, and the conditional flow matching training objective reduces to

$$\min_\theta \mathbb{E}_{t\sim[0,1],X_0\sim p_0,X_1\sim p_1}\|u_t^\theta(X_t) - (X_1 - X_0)\|^2. \tag{8}$$

It is worth noting that diffusion models (DMs) are an important class of flow-based models closely connected to FM (Lipman et al., 2024; Lai et al., 2025).

C Algorithms

Algorithm 1 Stage I: Training the Fourier Encoder and Flow-Matching Decoder

Require: Training dataset $\mathcal{D}$, number of Fourier frequencies M, shape minibatch size S, flow-matching point batch size B

Ensure: Learned frequency set $K = \{k_m\}_{m=1}^M$ and shape-conditioned velocity field v_ψ

Initialize $K = \{k_m\}_{m=1}^M$ and decoder parameters ψ

for each training iteration **do**

Sample a minibatch of point clouds $\{X^{(s)}\}_{s=1}^S \sim \mathcal{D}$, where $X^{(s)} = \{x_j^{(s)}\}_{j=1}^N$

Compute the Fourier latent of each shape using all N points:

$$z_N^{(s)} = \Phi_K(\hat{\mu}_N^{(s)}) \in \mathbb{R}^{M\times 2}, \qquad s = 1,\dots,S$$

For each shape s, sample a single time $t^{(s)} \sim \mathrm{Unif}[0,1]$

Sample B shape–point pairs $\{(s_\ell, j_\ell)\}_{\ell=1}^B$ from the points in the current shape minibatch

For each sampled pair, set

$$x_{1,\ell} = x_{j_\ell}^{(s_\ell)}, \qquad z_\ell = z_N^{(s_\ell)}, \qquad t_\ell = t^{(s_\ell)}$$

Sample $x_{0,\ell} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_3)$ for $\ell = 1,\dots,B$

Construct

$$x_{t,\ell} = (1-t_\ell)x_{0,\ell} + t_\ell x_{1,\ell}$$

Compute

$$\mathcal{L}_{\text{decoder}}(\psi, K) = \frac{1}{B}\sum_{\ell=1}^B \|v_\psi(x_{t,\ell}, t_\ell, z_\ell) - (x_{1,\ell} - x_{0,\ell})\|_2^2$$

Update ψ and K by minimizing $\mathcal{L}_{\text{decoder}}(\psi, K)$

end for

return K and v_ψ

D Statistical Properties of the Fourier Representation

We investigate the statistical properties of the Fourier representation, focusing on its consistency across different input cardinalities and its stability under Gaussian input perturbations.

Algorithm 2 Stage II: Training the Frequency-Conditioned Latent DiT

Require: Training dataset $\mathcal{D}$, fixed frequency set $K = \{k_m\}_{m=1}^{M}$ learned in Stage I, and diffusion schedule $\{\alpha_t, \sigma_t\}_{t=1}^{T}$
Ensure: Trained frequency-conditioned DiT D_θ

Initialize DiT parameters θ
for each training iteration **do**
  Sample a point cloud $X = \{x_j\}_{j=1}^{N} \sim \mathcal{D}$
  Encode X into its empirical Fourier latent code

$$z = \Phi_K(\hat{\mu}_N) = \{z_m\}_{m=1}^{M} \in \mathbb{R}^{M\times 2}$$

  Sample $t \sim \mathrm{Unif}\{1, \dots, T\}$ and $\epsilon \sim \mathcal{N}(0, I_{2M})$
  Construct the noisy Fourier latent

$$z_t = \alpha_t z + \sigma_t \epsilon$$

  Construct frequency-conditioned tokens

$$h_{t,m} = \mathrm{Embed}([z_{t,m}; k_m]), \qquad m = 1, \dots, M$$

  Compute the v-prediction target

$$v_t = \alpha_t \epsilon - \sigma_t z$$

  Predict the latent denoising target

$$\hat{v}_t = D_\theta\big(\{h_{t,m}\}_{m=1}^{M}, t\big)$$

  Compute the diffusion loss

$$\mathcal{L}_{\text{diff}}(\theta) = \|\hat{v}_t - v_t\|_2^2$$

  Update θ by minimizing $\mathcal{L}_{\text{diff}}(\theta)$
end for
return D_θ

Algorithm 3 Point Cloud Generation with FourierDiT

Require: Fixed frequency set $K = \{k_m\}_{m=1}^{M}$, trained frequency-conditioned DiT D_θ, shape-conditioned decoder v_ψ, target cardinality n
Ensure: Generated point cloud $\hat{X}_n$

Step 1: Generate a Fourier latent
Sample $z_T \sim \mathcal{N}(0, I_{2M})$
for $t = T, \dots, 1$ **do**
  Construct frequency-conditioned tokens $\{[z_{t,m}; k_m]\}_{m=1}^{M}$
  Apply one standard reverse-diffusion step using D_θ to obtain z_{t-1}
end for
Set $z^\star \leftarrow z_0$
Step 2: Decode the generated shape
Sample $X_0 = \{x_{0,j}\}_{j=1}^{n}$, where $x_{0,j} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_3)$
Integrate $\frac{dx_{t,j}}{dt} = v_\psi(x_{t,j}, t, z^\star)$ from $t = 0$ to $t = 1$ for $j = 1, \dots, n$
return $\hat{X}_n = \{x_{1,j}\}_{j=1}^{n}$

D.1 Sampling Consistency

We examine how the empirical Fourier representation varies with the number of input points. For each of the Airplane, Chair, and Car categories, we select 100 validation shapes and use the Fourier representation computed from the corresponding 15,000-point cloud as a finite reference. For each shape, we randomly downsample the point cloud to different cardinalities while keeping the learned frequency set fixed with $M = 128$. We repeat the downsampling 10 times for each cardinality.

Let $z_N^{(s,r)}$ denote the Fourier representation of shape s obtained from the r-th random downsampling to N points, and let $z_{\text{ref}}^{(s)}$ denote the representation computed from its full 15,000-point cloud. We measure the relative encoding error

as

$$E(N) = \frac{1}{SR} \sum_{s=1}^{S} \sum_{r=1}^{R} \frac{\left\| z_N^{(s,r)} - z_{\text{ref}}^{(s)} \right\|_F}{\left\| z_{\text{ref}}^{(s)} \right\|_F}, \tag{9}$$

where $S = 100$ and $R = 10$.

The results are shown in Fig. 5.

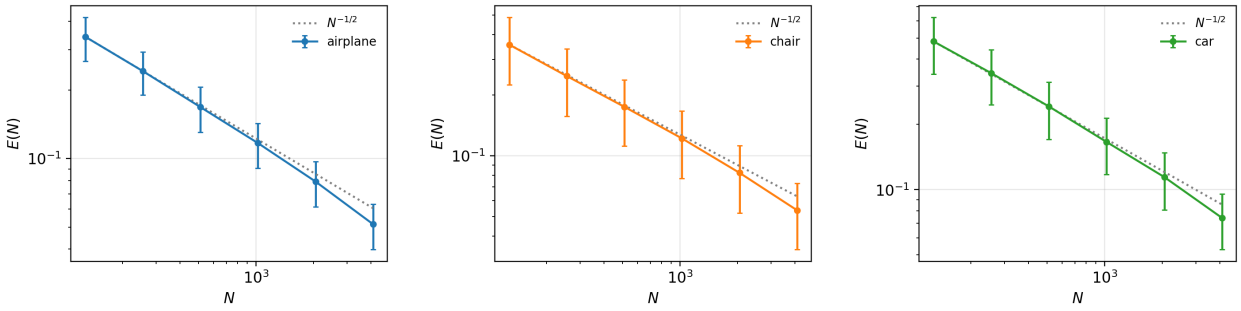


Figure 5: Sampling consistency of the empirical Fourier representation across different input cardinalities for Airplane, Chair, and Car (left to right). Relative encoding errors are measured with respect to the representations computed from the corresponding 15,000-point clouds.

As shown in Fig. 5, the relative encoding error decreases consistently as the number of sampled points increases across all three categories. The observed log–log scaling is close to the $N^{-1/2}$ rate expected from Monte Carlo estimation of empirical characteristic-function values.

Because the reference representation is itself computed from a finite 15,000-point cloud rather than the underlying population distribution, the measured error also reflects finite-reference effects and approaches zero as the sampled point set approaches the reference cloud. Overall, the results show that the empirical Fourier representation becomes increasingly consistent with its finite-sample reference as more input points are used.

D.2 Stability under Gaussian Jitter

We examine the stability of the empirical Fourier representation under Gaussian perturbations of the input points. Given a point cloud $X = \{x_i\}_{i=1}^N$, we construct a perturbed point cloud $\widetilde{X} = \{\widetilde{x}_i\}_{i=1}^N$ as

$$\widetilde{x}_i = x_i + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0, \sigma^2 I_3),$$

while keeping the learned frequency set fixed. We measure the overall relative change in the Fourier representation by

$$D_{\text{latent}} = \frac{\|\widetilde{z} - z\|_F}{\|z\|_F}, \tag{10}$$

where z and $\widetilde{z}$ denote the Fourier representations of the original and perturbed point clouds, respectively.

Gaussian perturbations admit a simple characterization in the Fourier domain. Let $\widetilde{\mu}$ denote the convolution of the original shape distribution μ with isotropic Gaussian noise of variance σ^2. Its characteristic function satisfies

$$\varphi_{\widetilde{\mu}}(k) = \varphi_\mu(k) \exp\left(-\frac{\sigma^2 \|k\|_2^2}{2}\right). \tag{11}$$

For a finite point cloud, the corresponding empirical characteristic function satisfies

$$\mathbb{E}_\epsilon \left[\widehat{\varphi}_{\widetilde{X}}(k) \,\middle|\, X\right] = \widehat{\varphi}_X(k) \exp\left(-\frac{\sigma^2 \|k\|_2^2}{2}\right). \tag{12}$$

Thus, Gaussian jitter attenuates Fourier coefficients in a frequency-dependent manner, with higher spatial frequencies undergoing stronger attenuation in expectation.

The overall latent drift under different perturbation strengths is shown in Fig. 6.

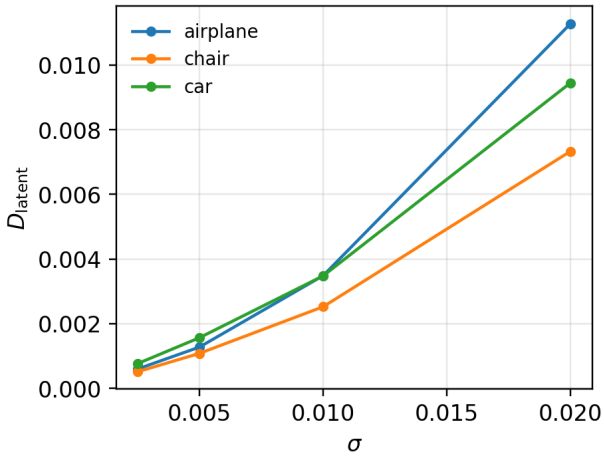


Figure 6: Stability of the Fourier representation under Gaussian jitter. Relative latent drift is measured as a function of the input noise standard deviation for Airplane, Chair, and Car.

As shown in Fig. 6, the Fourier representation changes gradually as the perturbation strength increases. At $\sigma = 0.02$, the mean relative latent drift is 0.94%.

To examine this frequency dependence more directly, we sort the learned frequencies by their magnitudes $r_m = \|k_m\|_2$ and partition them into four equal-size frequency bands $B_1, \ldots, B_4$, each containing $M/4 = 32$ tokens. For each band, we measure the average empirical coefficient drift

$$D_b(\sigma) = \frac{1}{S|B_b|} \sum_{s=1}^{S} \sum_{m \in B_b} \left\| \tilde{z}_m^{(s)} - z_m^{(s)} \right\|_2 , \tag{13}$$

where S is the number of evaluated shapes, and $z_m^{(s)}, \tilde{z}_m^{(s)} \in \mathbb{R}^2$ denote the original and perturbed Fourier coefficients of token m for shape s, respectively.

From Eq. (12), the attenuation factor associated with frequency k_m is

$$a_m(\sigma) = \exp\left(-\frac{\sigma^2 r_m^2}{2}\right). \tag{14}$$

Accordingly, the conditional mean of the perturbed Fourier coefficient is $a_m(\sigma) z_m^{(s)}$, so the magnitude of the attenuation-induced shift in its mean is

$$\left\| z_m^{(s)} \right\|_2 (1 - a_m(\sigma)) .$$

Using the same aggregation as in Eq. (13), we define the corresponding theoretical band-wise prediction as

$$D_b^{\text{theory}}(\sigma) = \frac{1}{S|B_b|} \sum_{s=1}^{S} \sum_{m \in B_b} \left\| z_m^{(s)} \right\|_2 (1 - a_m(\sigma)) . \tag{15}$$

The empirical and theoretical frequency-dependent responses are compared in Fig. 7.

As illustrated in Fig. 7, higher-frequency bands exhibit larger coefficient drift under Gaussian perturbations than lower-frequency bands. The empirical responses also closely follow the frequency-dependent trends predicted by the characteristic-function attenuation model across the evaluated noise levels.

Overall, these results show that the Fourier representation responds to Gaussian input perturbations in a frequency-dependent manner, with higher spatial frequencies exhibiting greater sensitivity to noise.

E Analysis of Latent Diffusion

We further analyze the learned Fourier latent space through PCA visualizations of real and generated representations and an examination of frequency-resolved denoising behavior.

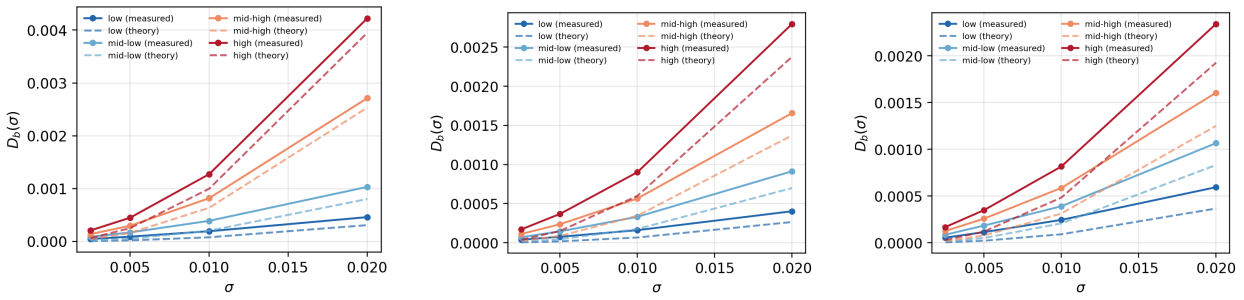


Figure 7: Frequency-dependent sensitivity to Gaussian jitter for Airplane, Chair, and Car (left to right). Learned frequencies are partitioned into four equal-size bands according to their magnitudes. The empirical band-wise coefficient drift is compared with the corresponding theoretical prediction induced by Gaussian attenuation of the characteristic function.

E.1 PCA Visualization of Real and Generated Latents

To qualitatively examine the learned latent distribution, we compare Fourier representations obtained from real point clouds with those sampled directly from the trained latent diffusion model.

For each category, real and generated latents are normalized using the statistics employed by the diffusion model and projected onto a shared two-dimensional PCA space. The generated latents are evaluated directly, without passing through the point-cloud decoder. The resulting projections are shown in Fig. 8.

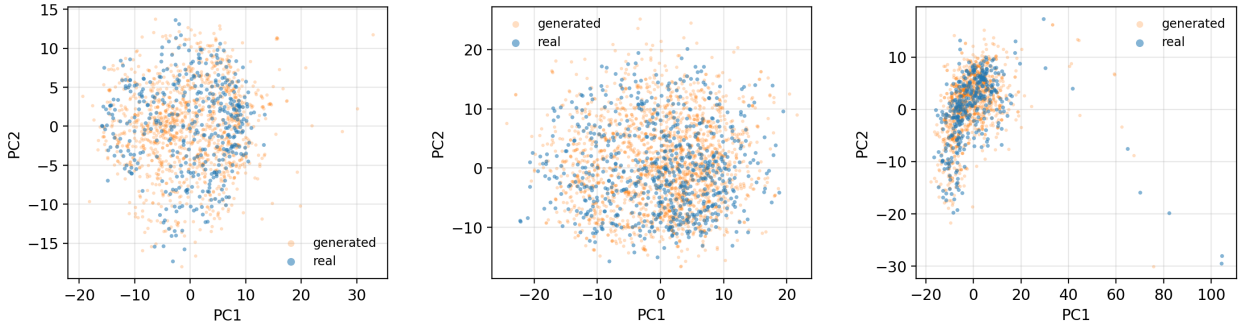


Figure 8: PCA visualization of real and generated Fourier latents for Airplane, Chair, and Car (left to right). Real and generated representations are projected onto the same PCA basis within each category.

As illustrated in Fig. 8, the projected distributions exhibit broadly overlapping regions between real and generated latents across the three categories, without visually obvious missing clusters in the displayed subspaces.

These visualizations provide a qualitative view of the latent distribution learned by the diffusion model, although two-dimensional projections do not fully characterize distributional agreement in the original latent space.

E.2 Frequency-Resolved Denoising Behavior

We further examine how the latent diffusion model recovers Fourier coefficients at different spatial frequencies. We apply the forward diffusion process to real validation latents at different noise levels and use the trained model to recover an estimate $\widehat{z}_0$ of the corresponding clean latent z_0. All latents in this analysis are expressed in the same normalized space used to train the DiT.

We use the same frequency partition as in Sec. D.2. Specifically, the learned frequencies are sorted by their magnitudes $\|k_m\|_2$ and divided into four equal-size bands $B_1, \ldots, B_4$, each containing $M/4 = 32$ Fourier tokens.

To make prediction errors comparable across tokens with different latent scales, we estimate a token-wise scale s_m from the training-split latents. For each token m, we first compute the unbiased across-shape variance of its real and

imaginary channels and define

$$s_m^2 = \frac{1}{2} \sum_{c \in \{\mathrm{Re},\mathrm{Im}\}} \widehat{\mathrm{Var}}_s \left[z_{m,c}^{(s)} \right]. \tag{16}$$

Thus, a single scalar s_m is shared by the two channels of token m. For each frequency band, we measure the normalized clean-latent prediction error

$$E_b(t) = \frac{1}{S|B_b|} \sum_{s=1}^{S} \sum_{m \in B_b} \left\| \frac{\hat{z}_{0,m}^{(s,t)} - z_{0,m}^{(s)}}{s_m} \right\|_2^2, \tag{17}$$

where S denotes the number of validation shapes. The same scale s_m is applied to both real and imaginary channels before their squared errors are summed.

We evaluate six normalized diffusion timesteps,

$$t/T \in \{1.0, 0.8, 0.6, 0.4, 0.2, 0.05\},$$

where larger t/T corresponds to stronger corruption in the forward diffusion process. For reproducibility, we use a fixed random seed and a single forward-noise realization at each evaluated timestep. The resulting frequency-resolved prediction errors are shown in Fig. 9.

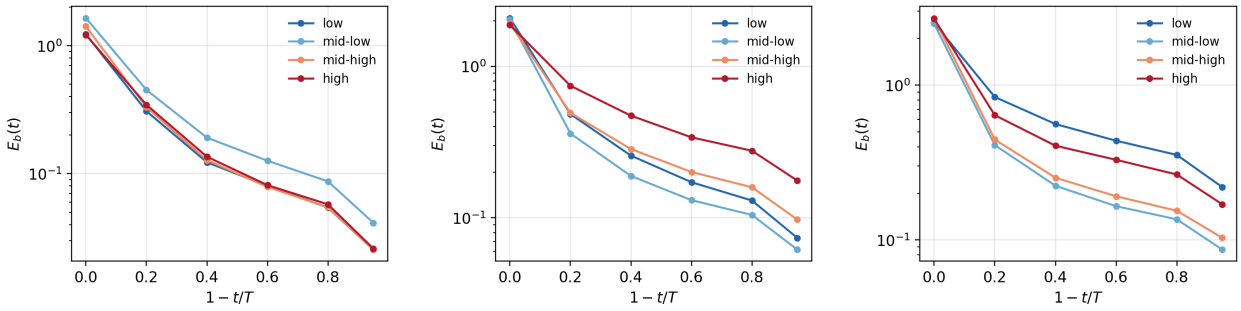


Figure 9: Frequency-resolved denoising behavior of the latent diffusion model for Airplane, Chair, and Car (left to right). Normalized clean-latent prediction errors are reported for four equal-size frequency bands at different diffusion noise levels.

As shown in Fig. 9, the prediction error decreases substantially across all frequency bands as the noise level decreases, with reductions of approximately 15–90$\times$ from $t/T = 1.0$ to $t/T = 0.05$.

The residual errors, however, exhibit category-dependent spectral patterns. Chair shows a larger residual error in the highest-frequency band, whereas Car exhibits its largest normalized residual error in the lowest-frequency band. Airplane shows comparatively more uniform residual errors across frequency bands.

These results do not indicate a universal coarse-to-fine denoising order. Instead, the relative difficulty of recovering different Fourier components varies across the evaluated shape categories.

F Unconditional 55-Class Generation

We further evaluate FourierDiT on unconditional generation across all 55 ShapeNet categories to investigate its ability to model a more diverse distribution of 3D shapes.

We employ a single shared Fourier frequency bank with $M = 128$ and a single unconditional latent diffusion model, without category labels, class embeddings, or classifier guidance. We adopt the ShapeNet normalization convention used in the LION multiclass setting.

We compare FourierDiT with the published 55-class results of LION using pooled evaluation metrics. For FourierDiT, 1-NNA is computed under Chamfer Distance (CD) and Earth Mover's Distance (EMD), using 1,000 generated samples and 1,000 reference samples. We additionally report coverage (COV) under both distances. The results are summarized in Table 6.

Table 6: Unconditional generation on 55 ShapeNet categories. Pooled 1-NNA scores (%) are better when closer to 50%, while higher coverage (COV, %) is better. LION results are taken from its published evaluation.

	1-NNA (%)		COV (%)	
Method	CD	EMD	CD	EMD
LION	58.25	57.75	48.0	52.2
FourierDiT	57.40	56.95	48.1	50.1

As shown in Table 6, FourierDiT obtains pooled 1-NNA scores of 57.40% under CD and 56.95% under EMD, compared with 58.25% and 57.75% reported by LION, respectively.

The two methods obtain comparable CD-based coverage, while LION reports higher EMD-based coverage.

These results demonstrate the applicability of FourierDiT to multiclass unconditional generation using a shared Fourier representation and a single latent diffusion model to capture a mixture of 55 shape categories.