FoldDiff: Folding in Point Cloud Diffusion

Anonymous authors
Paper under double-blind review

Abstract

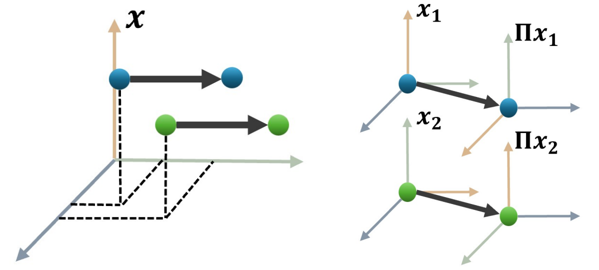
Diffusion denoising has emerged as a powerful approach for modeling data distributions, treating data as particles with their position and velocity modeled by a stochastic diffusion processes. While this framework assumes data resides in a fixed vector spaces (e.g., images as pixel-ordered vectors), point clouds present unique challenges due to their unordered representation. Existing point cloud diffusion methods often rely on voxelization to address this issue, but this approach is computationally expensive, with cubically scaling complexity. In this work, we investigate the misalignment between point cloud irregularity and diffusion models, analyzing it through the lens of denoising implicit priors. First, we demonstrate how the unknown permutations inherent in point cloud structures disrupt denoising implicit priors. To address this, we then propose a novel folding-based approach that reorders point clouds into a permutation-invariant grid, enabling diffusion to be performed directly on the structured representation. This construction is exploited both globally and locally. Globally, it can be used to represent point clouds in a fixed vector space (like images), therefore it enables us to extend the work of denoising as implicit priors to point clouds. On the other hand, exploiting this idea locally, allows us to create efficient and novel token representations that can improve existing transformer-based point cloud diffusion models. Our experiments show that the proposed folding operation integrates effectively with both denoising implicit priors as well as advanced diffusion architectures, such as UNet and Diffusion Transformers (DiTs). Notably, DiT with folded tokens achieves competitive generative performance compared to state-of-the-art models while significantly reducing training and inference costs relative to voxelization-based methods. Code is available at `http://anonymous.4open.science/r/FoldDiff-3B36/`.

1 Introduction

Modern representations of images and surfaces are often high-dimensional, encompassing thousands of pixels or 3D points. These representations typically reside on low-dimensional manifolds, either explicitly defined or inferred. Recently advancements in diffusion models (Ho et al., 2020; Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Song et al., 2021; Peebles & Xie, 2023) have demonstrated their efficacy in capturing implicit data distributions, and are conceptually linked to thermal systems. The forward diffusion process simulates thermal agitation, gradually transforming data into a standard Gaussian; while the reverse process mimics cooling, reconstructing data via Langevin-style probabilistic gradient ascent. However, this thermal-dynamics-inspired framework treats data as particles and assumes that the data reside in a structured vector space (e.g., images as pixel-ordered vectors). The irregularity and lack of inherent order in point clouds pose significant challenges to the direct application of diffusion or Langevin dynamics.

The problems are illustrated in Fig. 1, where a diffusion process or Langevin dynamics describes the motion of a particle in an arbitrary space. When we consider RGB images of height H and width W as particles, their positions are described by vectors in $\mathbb{R}^{H \times W \times 3}$ and updated by diffusion denoising or Langevin dynamics. Each entry in such vectors follows the same permutation across objects, and such a correspondence is preserved during denoising. This correspondence is missing in unordered point clouds. Thus, unlike image "particles," the motions of "particles" that represent the point cloud objects cannot be properly described due to the absence of this structured space. Permutation invariant choices for denoising loss further impose irretrievable permutation matrices on the noisy point clouds. This observation aligns with the findings reported in Zhou et al. (2021) and Mo et al. (2023), where similar challenges were encountered in designing diffusion models for unstructured point clouds.

The theoretical framework of "denoising implicit priors" investigates diffusion from its fundamental building blocks: Denoising Autoencoders (DAEs). As first shown in Vincent (2011), a DAE that minimizes the l_2 norm between a recovered signal and a clean one is equivalent to an energy model that tries to match its score to that of a non-parametric Parzen density estimator of the data. The theoretical connection between DAEs and score-matching (Vincent, 2011) guarantees that each denoising step moves the data towards higher probability regions. Thus, authentic samples can be drawn from the implicit priors that DAEs learned from l_2 noise regression. In this work, we employ the theoretical framework of denoising implicit priors to, for the first time, offer a comprehensive theoretical analysis of diffusion denoising in unstructured spaces. In Sec. 3, we theoretically prove that such unknown permutations breaks the proportional relationship between an l_2 denoising residual and the probability score, which are also empirically demonstrated in our experiments (Sec. 5.3).



(a) Structured space with consistent axis.

(b) Unstructured space with unknown permutation.

Figure 1: (a) Particle motions can be properly described in a shared and structured space (e.g., RGB images with the same height and width). (b) Particle motions cannot be properly described in unstructured spaces (e.g., unknown permutation between unordered point cloud objects) or during unstructured motion (e.g., unknown permutation imposed by permutation invariant denoising). Axes correspondence are noted by colors.

Recent diffusion models for point clouds have opted for voxel grids (Zhou et al., 2021; Zeng et al., 2022; Mo et al., 2023) or triplanes (Shue et al., 2023) as structured representations for the diffusion processes. However, these methods suffers from cubically or quadratically scaling computational complexities as well as quantization errors. In this work, we proposed a compact alternative based on the folding operation (Yang et al., 2018), which reconstructs a permutation-invariant, grid-like representation of the input point cloud. The folded reconstruction can be viewed as a fixed permutation that approximates the original geometry, providing a structured representation for point cloud diffusion. This novel framework, *FoldDiff*, is different from preview methods that performs diffusion on a voxelized space. It enjoys a linearly scaling computational complexity with the number of points in each object during diffusion denoising. The resulting grid-like representation, akin to a Geometry Image (GI) (Gu et al., 2002), pairs effectively with image denoisers to learn implicit priors of the original manifold of 3D objects. By combining image denoisers trained on folded objects with the Langevin-style sampling algorithm from Kadkhodaie & Simoncelli (2021), 3D objects can be sampled from the denoising implicit priors. Additionally, we test our framework on the Denoising Diffusion Probabilistic Models (DDPMs) (Ho et al., 2020) with both UNet (Ronneberger et al., 2014) and DiT (Peebles & Xie, 2023) backbones, and observed qualitatively and quantitatively competitive generative performances at significantly reduced cost. Notably, the use of folded tokens significantly reduces the token count in DiTs relative to its voxelization-based variant, leading to substantial improvements in training and sampling efficiency for point cloud diffusion.

Our main contributions can be summarized as follows:

- We present a theoretical analysis of the intractability of applying diffusion or Langevin dynamics to unordered data structures, grounded in the theory of "denoising implicit priors."

- We propose a novel folding-based point cloud diffusion framework, dubbed *FoldDiff*, offering greater efficiency than popular voxelization-based methods while enabling a seamless unification of 2D and 3D diffusion methods.

- We empirically validate our novel framework on denoising implicit priors, UNet-based DDPMs and DiT-based DDPMs, demonstrating competitive generative performance with a lower training and inferencing cost.

2 Related work

Diffusion models and denoising implicit priors. Diffusion models (Ho et al., 2020; Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Song et al., 2021; Peebles & Xie, 2023) have recently emerged as powerful generative models, outperforming earlier approaches like VAEs (Kingma & Welling, 2014), normalizing flows (Kobyzev et al., 2020), and GANs (Goodfellow et al., 2014). Denoising Diffusion Probabilistic Models (DDPMs) (Ho et al., 2020) utilize a fixed Markovian forward process that incrementally corrupts the data with Gaussian noise, paired with a learned noise-conditioned denoiser that constructs the reverse Markovian process. These models can be interpreted as Hierarchical Variational Autoencoders (HVAEs) (Kingma et al., 2016; Sønderby et al., 2016), Score-based Generative Models (SGMs) (Song & Ermon, 2019; Song et al., 2021), or stacks of Denoising Autoencoders (DAEs) (Kadkhodaie et al., 2024).

In this work, we explore diffusion models through the lens of DAEs, focusing on their connection to score matching and probability gradient ascent (Vincent et al., 2008). Central to their success is the use of DAEs to estimate the data distribution's score at varying noise levels, enabling denoising implicit priors on the data manifold. In particular, Kadkhodaie & Simoncelli (2021) proposed a stochastic gradient ascent procedure to sample from image denoisers. Our work builds upon this idea, but addresses the unique challenges in extending it for the first time to the point cloud domain. Building on the concept of denoising implicit priors, we propose a novel folding-based solution to address point cloud irregularity, which is compatible with models such as DDPMs using both UNet (Ronneberger et al., 2014) and DiT (Peebles & Xie, 2023) architectures.

Diffusion models for point clouds. Generative models for point clouds aims to capture "distribution (objects in $\mathbb{R}^{N\times 3}$) of distributions (points in $\mathbb{R}^3$)." Diffusion process was first adopted in Luo & Hu (2021b) to model the distribution of points in $\mathbb{R}^3$ conditioned on a PointNet (Qi et al., 2017a) encoded shape latent. Our work instead examines the diffusion models at the object level, where the "particles" are point cloud objects in $\mathbb{R}^{N\times 3}$.

Due to the irregularity of point clouds, mainstream 3D diffusion models perform diffusion either on a voxelized space following Point Voxel Diffusion (Zhou et al., 2021), or on a encoded triplane representation following Triplane diffusion (Shue et al., 2023). These two strctured representations are easy to optimize and intuitively analogous to 2D pixels, but they both suffer from quantization errors and higher computational complexity. Comparing to generative models modeled on raw point clouds in $\mathbb{R}^{N\times 3}$, voxelization-based methods (Zhou et al., 2021; Zeng et al., 2022; Mo et al., 2023) perform diffusion in the voxel space in $\mathbb{R}^{C\times V\times V\times V}$. Here, C denotes the number of feature channels per voxel, and V represents the resolution of the voxel grid along each spatial dimension. Similarly, triplane-based methods (Shue et al., 2023) perform diffusion in the space of encoded triplane latents in $\mathbb{R}^{3C\times H\times W}$, where H and W specify the spatial resolution of each triplane feature map of dimension C. Our proposed folded structure fundamentally differs from previous structured representations, offering computational complexity at the order of the original point cloud ($\mathbb{R}^{N\times 3}$) while being fully compatible with established 2D diffusion architectures.

Point cloud denoisers. The unordered nature of point clouds poses a key challenge in applying image-based deep learning methods to 3D data. PointNet (Qi et al., 2017a) pioneered using permutation invariant operations, e.g., pointwise convolutions followed by channel-wise pooling, to address this. Subsequent works improved the expressiveness (Qi et al., 2017b; Wang et al., 2018) or efficiency (Liu et al., 2019) of this approach. State-of-the-art point cloud denoisers (Rakotosaona et al., 2020; Luo & Hu, 2020; 2021a; de Silva Edirimuni et al., 2023) adopt a similar permutation invariant encoder-decoder architecture, but focus on recovering local surface details rather than global shape of an object. We do not consider these local denoisers since we are interested in modeling the distribution of point cloud objects globally. As described in Sec. 5.3, we performed experiments on a PointNet-based object-level denoiser paired with the Chamfer Distance loss. We provide theoretical and empirical analysis on why this object-level denoiser cannot be directly used to sample shapes using “denoiser as prior” techniques (Sec. 3.1). We solve this problem with the approach here introduced (Sec. 4.1), and demonstrate the empirical applications of these ideas (Sec. 4.3).

Folding-based autoencoders and geometry images. Folding-based decoders (Yang et al., 2018; Groueix et al., 2018; Pang et al., 2020) became a popular design choice due to their expressiveness power. In general, the decoding process simulates a deformation process from a genus-0 primitive surface (i.e., a 2D grid lattice) to a shape that reconstructs the inputting point cloud. This 2D-to-3D deformation is deeply connected with Geometry Images (GIs) (Gu et al., 2002), which aims to simplify shape analysis with image-processing tools including CNNs (Zhang et al., 2022; Sinha et al., 2016; 2017; Maron et al., 2017). In our work, folding is adopted to obtain a permutational-invariant order of the inputting point cloud, creating a structured vector space for diffusion systems.

3 Denoising implicit priors on point clouds

3.1 Denoising on images and point clouds.

We represent the original input data (image or point cloud) as a vector $\mathbf{x} \in \mathbb{R}^{N \times 3}$. The N dimensions, can be associated to an RGB image of N pixels or a point cloud of N (3D) points. The goal is to efficiently represent the prior probability $P(\mathbf{x})$ (i.e., the manifold on the original space where data lies). In both image and point cloud denoising, it is common to assume that the noisy observation is a distribution conditioned on the clean observation, expressively $\mathbf{y} = \mathbf{x} + \mathbf{z}$, where $\mathbf{y} \in \mathbb{R}^{N \times 3}$ is the noisy observation, $\mathbf{x} \in \mathbb{R}^{N \times 3}$ is the clean observation, and $\mathbf{z} \sim \mathcal{N}(0, \sigma^2 I_N)$ is the Gaussian noise modeled as both additive and Gaussian (Kadkhodaie & Simoncelli, 2021). Thus, a denoiser is optimized to approximate the residual $\mathbf{z}$ to recover the clean image $\mathbf{x}$ given $\mathbf{y}$. If $\mathbf{x}$ is drawn from a prior distribution $P(\mathbf{x})$, its created noisy version can be considered a Gaussian convolved distribution $P(\mathbf{y})$ that can be rewritten via marginalization,

$$P(\mathbf{y}) = \int P(\mathbf{y}|\mathbf{x})P(\mathbf{x})dx = \int g(\mathbf{y} - \mathbf{x})P(\mathbf{x})dx, \tag{1}$$

where $g(\mathbf{z})$ is the multivariate Gaussian PDF with variance σ^2 that models the noise residual $\mathbf{z}$.

Even though the additive noise is modeled by a Gaussian distribution in both scenarios, point cloud denoising differs from image denoising in its optimization objectives. A key observation is that point clouds are unstructured data while images are structured. In image denoising, each noisy pixel $\mathbf{y}_i$ has a one-to-one correspondence with its ground truth clean pixel $\mathbf{x}_i$. In this case, optimizing a denoiser $f_\theta(\mathbf{y}) = \mathbf{y} - \mathbf{x}$ is equivalent to minimizing the objective

$$\underset{\theta}{\operatorname{argmin}}\ \mathbf{E}_{\mathbf{y} \sim P(\mathbf{y})} \left[\mathcal{L}(\tilde{\mathbf{x}}, \mathbf{x}) \right], \tag{2}$$

where $\mathcal{L}$ is commonly the l_2 norm, and $\tilde{\mathbf{x}} = \mathbf{y} - f_\theta(\mathbf{y})$ is the recovered signal. Unlike images, a clean point cloud is a discrete set of samples from a surface $\mathcal{S}$ embedded in 3D space. Hence, each noisy point $\mathbf{y}_i$ can have multiple origins from the clean surface $\mathcal{S}$, which breaks the point-wise correspondence between noisy and clean point clouds. If we naively treat every point cloud with its given permutation as a vector and supervise denoising with the l_2 norm, the point-wise correspondence is still unknown during test time. As reported in Rakotosaona et al. (2020), a denoiser optimized to minimize the objective given in Equation 2 is not recovering the original clean point but an average of all possible candidates that the noisy point originates from. This leads to worse denoising performance as it does not guarantee the average to be a point on the surface. Another option is to minimize the distance from denoised points to the underlying surface through

$$\underset{\theta}{\operatorname{argmin}}\ \mathbf{E}_{\mathbf{y} \sim P(\mathbf{y})} \left[l(\tilde{\mathbf{x}}, \mathcal{S}) \right]. \tag{3}$$

In practice, we only have access to clean points $\mathbf{x}$ as discretized samples from the underlying surface. Proximity to the surface is thus the Euclidean distance between a noisy point $\mathbf{y}_i$ and its nearest neighbor (NN) in $\mathbf{x}$,

$$\left| \mathbf{y}_i - NN(\mathbf{y}_i, \mathbf{x}) \right|. \tag{4}$$

A common loss function $l(\tilde{\mathbf{x}}, \mathcal{S})$ (Rakotosaona et al., 2020; Roveri et al., 2018) is then

$$\|\tilde{\mathbf{x}} - NN(\mathbf{y}, \mathbf{x})\|_2^2 = \|f_\theta(\mathbf{y}) - (\mathbf{y} - NN(\mathbf{y}, \mathbf{x}))\|_2^2. \tag{5}$$

This strategy presents two main challenges: (1) nearest neighbors need to be computed online, which is computationally very demanding; and (2) the process of denoising introduces an unknown permutation in the points representations. Next, we discuss the challenges of this permutation matrix.

3.2 Denoising implicit priors in structured and unstructured spaces.

Denoising CNNs can be viewed as least squares estimators that recover the true signal by computing the conditional mean of the posterior,

$$\hat{\mathbf{x}}(\mathbf{y}) = \int \mathbf{x} P(\mathbf{x}|\mathbf{y}) dx = \int \mathbf{x} \frac{P(\mathbf{y}|\mathbf{x})P(\mathbf{x})}{P(\mathbf{y})} dx, \tag{6}$$

where $\hat{\mathbf{x}}(\mathbf{y})$ is the best (in the l_2 sense) approximation of the recovered signal. This solution can be expressed as (Miyasawa, 1961)

$$\hat{\mathbf{x}}(\mathbf{y}) = \mathbf{y} + \sigma^2 \nabla_{\mathbf{y}} \log P(\mathbf{y}). \tag{7}$$

The predicted residual $\hat{\mathbf{x}}(\mathbf{y}) - \mathbf{y}$ is then proportional to the score function $\nabla_{\mathbf{y}} \log P(\mathbf{y})$. This implies that the training objective of a CNN denoiser is equivalent to score matching. The noise residual provides a direction to move up a probability gradient toward a clean image density. Thus, as proposed in Kadkhodaie & Simoncelli (2021), we can gradually converge to the manifold $P(\mathbf{x})$ using Langevin style gradient ascent.

Unfortunately, Equation 7 no longer holds when the data is unstructured, and therefore the whole idea of “denoiser as prior” collapses. To formalize this problem, let’s consider a noisy observed point set $\mathcal{Y} = \{\mathbf{x}_i + \mathbf{z}_i : i = 1, \ldots, N\}$, where we assume $\mathbf{z}_i \sim \mathcal{N}(0, \sigma^2)$ for all i, and $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ is the “correct” vectorization of clean points. Any vectorization of $\mathcal{Y}$ can be rewritten as

$$\mathbf{y} = \mathbf{\Pi} \left(\mathbf{x} + \mathbf{z} \right), \tag{8}$$

where $\mathbf{\Pi}$ is a permutation matrix. Since $\mathbf{z}_i \sim \mathcal{N}(0, \sigma^2)$, we have $\mathbf{\Pi z} \sim \mathcal{N}(0, \sigma^2 \mathbf{\Pi})$, i.e., $\mathbf{z}$ and $\mathbf{\Pi z}$ have the same distribution, and since $\mathbf{z}$ is not directly observed, without loss of generality, we can write

$$\mathbf{y} = \tilde{\mathbf{x}} + \mathbf{z}, \tag{9}$$

where $\tilde{\mathbf{x}} = \mathbf{\Pi x}$ is a permutation of the original vectorization and $\mathbf{z} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$. Note that $\mathbf{\Pi} = \mathbf{I}_{N \times N}$ if and only if $\mathbf{y}_i = \mathbf{x}_i + \mathbf{z}_i$ for all i. Suppose we naively vectorize $\mathcal{Y}$ to obtain $\mathbf{y}$ by a random permutation. Given a permutation $\mathbf{\Pi}$, we can rewrite the marginal distribution of the observation $P(\mathbf{y}|\mathbf{\Pi})$ and the least squares estimate $\hat{\mathbf{x}}(\mathbf{y})$ as

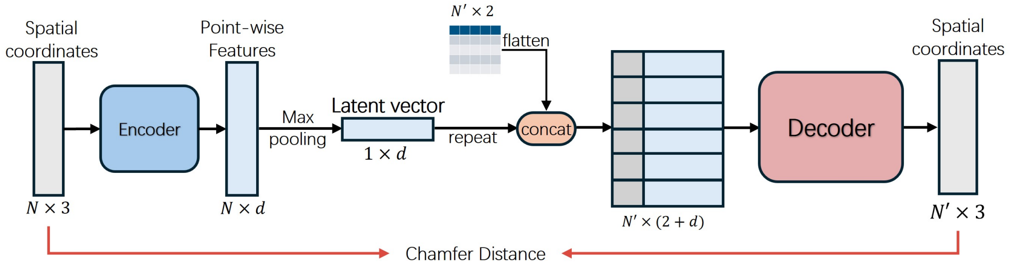
$$P(\mathbf{y}|\mathbf{\Pi}) = \int P(\mathbf{y}|\mathbf{x}, \mathbf{\Pi}) P(\mathbf{x}|\mathbf{\Pi}) d\mathbf{x} = \int g(\mathbf{y} - \tilde{\mathbf{x}}) P(\mathbf{x}|\mathbf{\Pi}) d\mathbf{x}, \tag{10}$$

$$\hat{\mathbf{x}}(\mathbf{y}) = \int \mathbf{x} P(\mathbf{x}|\mathbf{y}, \mathbf{\Pi}) d\mathbf{x} = \int \mathbf{x} \frac{P(\mathbf{y}|\mathbf{x}, \mathbf{\Pi}) P(\mathbf{x}|\mathbf{\Pi})}{P(\mathbf{y}|\mathbf{\Pi})} d\mathbf{x}. \tag{11}$$

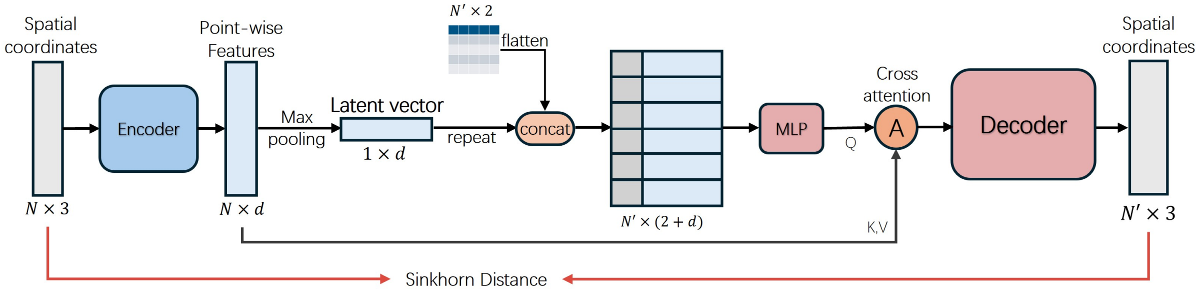
The following proposition then follows.

Proposition 1. *The denoiser residual* $f(\mathbf{y}) = \hat{\mathbf{x}} - \mathbf{y}$ *is proportional to* $\nabla_{\mathbf{y}} \log P(\mathbf{y})$ *if and only if* $\mathbf{\Pi} = \mathbf{I}_{N \times N}$.

The proof is presented in the Appendix A.1. This proposition implies that Equation 7 can only be applied to a structured data format that always follows its original permutation, which is not generally the case in point cloud denoising, validating the empirical observations in Sec. 5.3.



(a) Original FoldingNet.



(b) Modified FoldingNet.

Figure 2: Overviews of the original FoldingNet and our proposed modifications.

4 *FoldDiff* for 3D shape generation

4.1 FoldingNet as reordering module

To solve the challenge of permutation and irregularity in point cloud denoising, we choose to reorder the irregular inputting point cloud via a Folding-based autoencoder (Yang et al., 2018). The original FoldingNet architecture is illustrated in Fig. 2(a), which encompasses a permutation-invariant encoder, based on PointNet (Qi et al., 2017a), and a folding-based decoder. The encoder outputs a one-dimensional latent vector invariant to the input point cloud's ordering. This latent vector is concatenated with coordinates from a predefined 2D grid as a primitive structure. The decoder then models the "deforming forces" acting on the 2D grid to reconstruct the input point cloud. Since the decoder uses two inputs—the global latent vector and the predefined 2D grid—both of which are permutation-invariant to the original point cloud, the reconstructed output also inherits this property. Consequently, the folding operation produces a reordered point cloud reconstruction embedded in a structured vector space, enabling us to treat the reconstruction as "particles" suitable for diffusion systems.

4.2 Modified FoldingNet

In our framework, we introduce two modifications to the original FoldingNet. Fig. 2 shows an overview of the original FoldingNet and our proposed alternative. First, while FoldingNet was originally optimized using Chamfer Distance (CD), we replace it with the Sinkhorn Distance (Cuturi, 2013; Feydy et al., 2019). Since our goal is to use the FoldingNet to find a fixed permutation that reconstructs the inputting point clouds, it can be formulated as an optimal transport problem that aims to transport the source unordered points to the target reordered reconstructions, and the Sinkhorn Distance is a differentiable solution. Second, we incorporate an attention block following Wen et al. (2020) to improve the reconstruction quality without breaking the permutation-invariant property. As illustrated in Fig. 2(b), the queries are generated by a multilayer perceptron (MLP) that processes the 2D coordinates concatenated with the max-pooled global feature. Subsequently, the attention module automatically selects the most informative encoded point-wise features corresponding to these queries, thereby establishing a mapping—a permutation—from an input

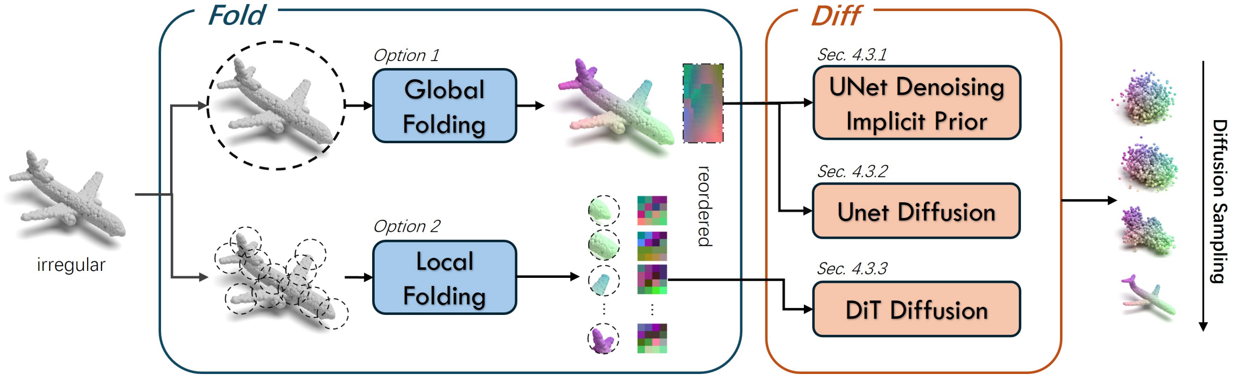


Figure 3: An overview of the proposed *FoldDiff* framework. Folding operation provides a reordered reconstruction that is permutation invariant to the input. The resulting structured global or local representation can be paired with various 2D diffusion techniques and enables efficient point cloud diffusion sampling without the need for voxelization.

point to a specific position on the 2D grid. This process yields integrated features that guide the decoder in producing more precise reconstructions, while the decoder in the original design (Fig. 2(a)) relies solely on the global feature. Given that the attention mechanism is permutation-equivariant with respect to the queries, and considering that our queries are permutation-invariant features, the permutation-invariance of the reordered reconstruction is preserved.

To validate our just introduced modifications to the local FoldingNet, we report the reconstruction errors in Chamfer Distances (CD) and Earth Moving Distances (EMD) on small local patches. Training local patches are extracted from chairs, cars, and airplanes in the ShapeNet dataset, where each object contains 2048 uniformly sampled points via farthest point sampling. In our default setting, each local patch resembles a $k \times k$ geometry image that aggregates 16 nearest neighbors of a randomly chosen center point. The models were trained for 1,000 epochs with a batch size of 65,536 and a learning rate annealing from 2×10^{-4} to 2×10^{-6} with a cosine schedule. The experimental results in Table 1 demonstrates that our modifications boost the reconstruction quality of folded local patches with only a 0.2 Gflops overhead.

Table 1: FoldingNet performance on small local patches of 16 points with different modifications.

Attention	Gflops	loss	CD(10^{-3})	EMD(10^{-2})
✗	1.6799	CD	3.9188	7.5219
✓	1.8716	CD	3.1273	7.0969
✓	1.8716	Sinkhorn	**2.9870**	**0.4456**

While the modified FoldingNet demonstrated in Fig. 2 is sufficient for tokenizing local patches, we use a hierarchical FoldingNet to better reconstruct point clouds globally. The architecture is inspired by PointNet++ (Qi et al., 2017b) and the Laplacian Pyramid, and the details are covered in Appendix A.2. The resulting reconstructions is then reshaped into a $H \times W$ geometry image (GI), where each pixel encodes the (x, y, z) coordinates of the reconstructed geometry. The GI is then naturally compatible with 2D UNet denoiser and UNet-DDPMs trained with l_2 noise residuals.

4.3 The *FoldDiff* framework

To address the irregularity in point clouds for diffusion models without incurring the high computational complexities and quantization errors of voxelization-based approaches, we propose *FoldDiff* (demonstrated in Fig. 3), which performs denoising diffusion on folded point clouds. We explore three variants of our framework: (1) 2D UNet denoising implicit priors paired with folded objects; (2) 2D UNet-based DDPMs paired with folded objects; and (3) DiT-based DDPMs paired with folded tokens.

Sampling folded objects from UNet implicit priors. Modeling the manifold of reordered reconstructions instead of unordered inputs circumvents the challenges posed by permutation matrices as shown in Proposition 1. The folded reconstructions can be trivially reshaped into an image-like representation (i.e., Geometry Image), making them compatible with 2D CNN-based denoisers. As demonstrated in Equation 7, the predicted noise residual gives us a direction (i.e., score estimation) toward the clean manifold, thereby providing access to the density $p(\mathbf{x})$. A general denoiser can be considered a score approximator adaptive to different noise levels. It's connection to denoising diffusion provides us a proof-of-concept algorithm to illustrate the necessity of structured permutation. Following Kadkhodaie & Simoncelli (2021), we trained a 2D CNN UNet denoiser (Ronneberger et al., 2014) as our implicit prior. Architectural details of the UNet are covered in Appendix A.2. The training objective is the l_2-norm between the denoised residuals and the ground truth additive Gaussian noise on each reordered point. To sample high-probability objects from a denoiser, we follow the Langevin-style stochastic gradient ascent algorithm in Kadkhodaie & Simoncelli (2021). A detailed explanation of the sampling algorithm is included in Appendix A.3.

Sampling folded objects from Unet-based DDPMs. Extending this approach, we perform Denoising Diffusion Probabilistic Models (DDPMs) sampling using noise-conditioned UNet denoisers trained on globally folded point clouds. Unlike general denoisers, noise-conditioned UNets are specifically trained to predict the score at fixed noise levels rather than adapting to a wide range of noise levels. This specialization allows for more accurate score estimation, enhancing the sampling quality and diversity.

In DDPMs, the forward diffusion process corrupts data $\mathbf{x}_0$ by adding Gaussian noise over T timesteps, forming latent variables $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T$. The joint distribution of the forward process is defined as $q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$, where $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})$. With the reparametrization trick, samples $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x}_{t-1})$ can be rewritten as $\sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\epsilon_0$, where $\epsilon_0 \sim \mathcal{N}(0, \mathbf{I})$. The reverse process learns a score estimation network $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ such that $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, where $p(\mathbf{x}_T) \sim \mathcal{N}(0, \mathbf{I})$. The parameters θ are optimized to minimize the evidence lower bound (ELBO) on a negative log-likelihood of $\mathbf{x}_0$ under $p_\theta(\mathbf{x}_{0:T})$. The ELBO then reduces to minimizing the l_2-norm between the ground truth noise and the predicted noise up to a irrelevant constant multiplier, which coincides with the denoising score matching results as introduced in Sec 3.

Sampling from DiT-based DDPMs with folded tokens. If UNet based diffusion predicts the dynamics of "a particle" (i.e., an image or a point cloud), Diffusion Transformer (DiT) (Peebles & Xie, 2023) predicts the dynamics of "a particle group" (i.e., a group of tokens in an image or a point cloud). Here, an indestructible particle is no longer the entire object, but the tokens. In this case, the movement direction (i.e., probability gradient) of each "particle" (i.e., each token) is dependent not only on noise levels, but also on the interactions within a "particle group" (i.e., group of tokens), and such interactions are captured by the transformer (Vaswani et al., 2017).

The original DiT for images consists of two main components. The first is a pretrained **tokenizer**, which compresses the high-resolution image $\mathbf{x}$ into a lower-resolution latent image $\mathbf{z}$. This latent image is then divided into patch embeddings in the raster order. The second component is the **latent diffusion transformer**, which replaces the UNet in DDPMs. In the latent diffusion transformer, the l_2 noise on each token is predicted conditioned on token interactions captured by self-attention. The original DiT added positional embeddings to the image patch embeddings to retain positional information.

We dub our DiT as DiT-fold, where the tokenizer is a lightweight FoldingNet as described in Sec. 4.2, and the folded reconstructions serve as our tokens. We applied sinusoidal positional embeddings on the barycenter of each folded token, allowing the transformer to capture the spatial relationship between tokens. Additionally, without the constraints of voxel grids, our approach allows for any integer number of token length L, with each token a $k \times k$ local geometry image. We considered choices such that $Lk^2 \approx 2N$ for objects of N points. This design offers greater flexibility in sequence length compared to DiT architectures based on voxelization such as Mo et al. (2023), which is restricted to voxel dimensions $V \in \{16, 32, 64\}$, and patch widths $p \in \{2, 4, 8\}$, resulting in token lengths $L = (V/p)^3 \in \{2^3, 4^3, 8^3, 16^3, 32^3\}$. In their largest configuration ($V = 64, p = 2$), the token count reaches 32,768—significantly exceeding the number of points ($N = 2048$) in the experiments—leading to highly inefficient training and inference. Our proposed *FoldDiff* provides a more compact and efficient alternative to point-voxel representations for point cloud diffusion, despite some computational overhead for training a lightweight folding tokenizer. Such overhead can be minimized by training a universal local patch tokenizer applicable across all object categories. A well-

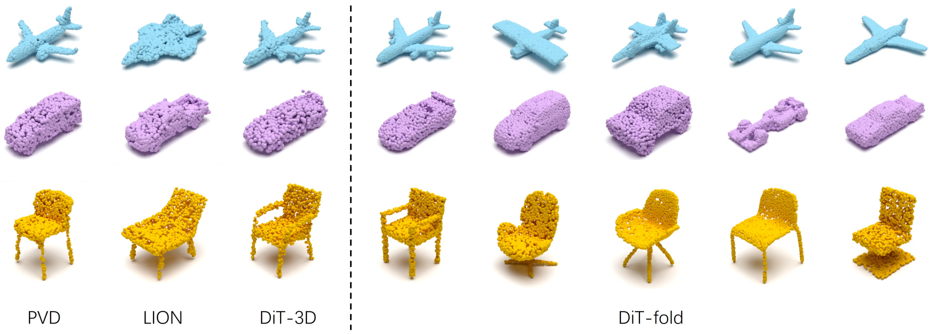


Figure 4: Single-class point cloud generation for *airplanes*, *cars*, and *chairs*. The proposed DiT-fold avoids the quantization error introduced from voxelization, thus generates smoother shapes.

optimized, lightweight tokenizer can then be directly paired with DiTs of any size to model diverse target distributions efficiently. Aside from the above modifications, our implementations align with the original DiT (Peebles & Xie, 2023).

5 Experiments

5.1 Experimental setup

Datasets. We compare the performance of different methods on single-category 3D shape generation using ShapeNet (Chang et al., 2015) chairs, cars, and airplanes as primary datasets. We use the same dataset splits of previous works (Luo & Hu, 2021b; Yang et al., 2019; Zhou et al., 2021; Zeng et al., 2022; Mo et al., 2023). Throughout our experiments, each object contains 2048 uniformly sampled points. During evaluation, both the generated shapes and the reference shapes are inversely transformed with the global mean and variance of the training set.

Implementation details. All models are trained with PyTorch. UNet implicit priors and DDPMs are trained on globally folded point clouds. DiT-based DDPMs use the folded local patches with a lightweight FoldingNet (see Sec. 4.2). Model details are covered in Appendix A.2.

Evaluation metrics. Following previous works (Luo & Hu, 2021b; Yang et al., 2019; Zhou et al., 2021; Zeng et al., 2022; Mo et al., 2023), we adopt coverage score (COV) and 1-Nearest-Neighbor classifier Accuracy (1-NNA) to quantitatively evaluate the sampling quality and diversity of different methods. Specifically, 1-NNA evaluates whether the generated distribution is identical to the reference distribution (e.g. test-set distribution). Closer to 50% implies better similarity between distributions, thus capturing both the sampling quality and diversity. COV calculates the proportion of shapes from the reference dataset to be matched with at least one generated shape. Higher COV implies better diversity. 1-NNAs and COVs are computed from the full pair-wise distance matrix of elements from both test set and the sampled set. Chamfer Distance (CD) and Earth Mover's Distance (EMD) were used as our distance metrics to compute 1-NNA and COV. Yang et al. (2019) provided a more detailed discussion over different evaluation metrics.

5.2 3D shape generation

We evaluate the generative performance and efficiency of DiT-DDPM-based *FoldDiff* against previous point cloud generative frameworks, as demonstrated in Table 2. We compared previous works with similar Gflops (Yang et al., 2019; Kim et al., 2020; 2021; Klokov et al., 2020; Luo & Hu, 2021b; Zhou et al., 2021; Mo et al., 2023). Among them, point-voxel architectures (Zhou et al., 2021; Zeng et al., 2022; Mo et al., 2023) demonstrate the most competitive generative performance, with DiT-3D-XL (Mo et al., 2023) recently achieving state-of-the-art results. Since LION (Zeng et al., 2022) has Gflops of 247.38 and DiT-3D-XL(64/2) in Mo et al. (2023) has Gflops of 1.12×10^4, they are omitted in our experiments for fair comparison. The point-voxel representation provides a structured space for point cloud diffusion, yet suffers from a cubically scaling complexity and quantization errors. In contrast, our folded representation avoids the above restrictions. As illustrated in Fig. 4, our framework reduces quantization noise, producing smoother point

Table 2: Comparison results (%) on shape metrics of our DiT-fold and baseline models. (*) denotes re-implemented performances on 2048 uniformly sampled points. Here we considered the methods with Gflops from 0 to 100. We report the average performance of 3 generative runs.

Method	Gflops	Chair				Airplane				Car			
		1-NNA (↓)		COV (↑)		1-NNA (↓)		COV (↑)		1-NNA (↓)		COV (↑)	
		CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
r-GAN Achlioptas et al. (2017)	–	83.69	99.70	24.27	15.13	98.40	96.79	30.12	14.32	94.46	99.01	19.03	6.539
l-GAN (CD) Achlioptas et al. (2017)	–	68.58	83.84	41.99	29.31	87.30	93.95	38.52	21.23	66.49	88.78	38.92	23.58
l-GAN (EMD) Achlioptas et al. (2017)	–	71.90	64.65	38.07	44.86	89.49	76.91	38.27	38.52	71.16	66.19	37.78	45.17
PointFlow Yang et al. (2019)	2.10	62.84	60.57	42.90	50.00	75.68	70.74	47.90	46.41	58.10	56.25	46.88	50.00
SoftFlow Kim et al. (2020)	11.29	59.21	60.05	41.39	47.43	76.05	65.80	46.91	47.90	64.77	60.09	42.90	44.60
SetVAE Kim et al. (2021)	1.61	58.84	60.57	46.83	44.26	76.54	67.65	43.70	48.40	59.94	59.94	**49.15**	46.59
DPF-Net Klokov et al. (2020)	3.02	62.00	58.53	44.71	48.79	75.18	65.55	46.17	48.89	62.35	54.48	45.74	49.43
DPM Luo & Hu (2021b)	2.09	60.05	74.77	44.86	35.50	76.42	86.91	48.64	33.83	68.89	79.97	44.03	34.94
PVD Zhou et al. (2021)	81.12	57.09	60.87	36.68	49.24	73.82	64.81	48.88	**52.09**	**54.55**	53.83	41.19	50.56
PVD* Zhou et al. (2021)	81.12	56.85	55.91	46.68	51.41	73.57	67.97	47.22	48.20	57.79	53.07	47.51	53.30
DiT-3D-S(32/4) Mo et al. (2023)	21.95	56.31	55.82	47.21	50.75	–	–	–	–	–	–	–	–
DiT-3D-B(32/4) Mo et al. (2023)	87.42	55.59	54.91	50.09	52.80	–	–	–	–	–	–	–	–
DiT-3D-S(32/4)* Mo et al. (2023)	21.95	59.19	55.82	44.96	51.16	78.57	67.86	45.57	47.78	63.99	61.65	40.06	50.28
DiT-fold-S(256/16) (Ours)	**12.80**	55.48	53.56	**48.03**	51.69	72.19	66.06	47.28	51.11	58.74	**52.26**	48.53	**53.79**
DiT-fold-S(512/9) (Ours)	23.92	**55.06**	**53.25**	46.83	**53.78**	**67.24**	**64.78**	**50.99**	51.97	56.63	52.85	47.73	52.40

clouds than PVD (Zhou et al., 2021), LION (Zeng et al., 2022), and DiT-3D (Mo et al., 2023). Moreover, we observed a generalized generative behavior as covered in Appendix A.4 by comparing a generated shape to its nearest neighbors in the training set.

Due to constraints on training budget, we compare the point-voxel framework with our *FoldDiff* framework under the same model size: DiT-S. We denote different configurations of DiT-fold as DiT-fold(L/k^2), where L stands for token length, and each token is a $k \times k$ local geometry image. We denote different configurations of our baseline, DiT-3D (Mo et al., 2023), as DiT-3D(V/p), where V is the voxel dimension, and p is the patch width. We re-implement point-voxel diffusion methods with similar Gflops including PVD and DiT-3D-S for a more comprehensive comparison in ShapeNet (Chang et al., 2015) chairs, cars, and airplanes, while the results for other methods are reported as in their original paper. For fairness, we report both the available published performance metrics and re-implemented results (denoted with *).

In our default configuration, we use DiT-S with 256 tokens, each containing 16 folded points, which corresponds to 4×4 local geometry images. During training, each token is captured from 16 nearest neighbors of 256 farthest sampled centers. During sampling, tokens are directly inferred and contains 3D position of points. We train the DiT-S with folded tokens for 10,000 epochs with AdamW optimizers using a learning rate of 2×10^{-4}. Following DiT (Peebles & Xie, 2023), we maintain an exponential moving average (EMA) of model weights over training with a decay of 0.9999 and the EMA weights were used during sampling for evaluation. This technique was also used in re-implemented DiT-3D.

As shown in Table 2, with similar or lower computational costs, our default configuration DiT-fold-S(256/16) demonstrates superior performance over traditional methods (Yang et al., 2019; Kim et al., 2020; 2021; Klokov et al., 2020; Luo & Hu, 2021b). More importantly, DiT-fold-S(256/16) surpasses the performances of point-voxel methods, PVD (Zhou et al., 2021) and DiT-3D-S (Mo et al., 2023), with equivalent or less Gflops. In particular, DiT-fold-S(256/16) outperforms re-implemented DiT-3D-S(32/4) with only 1/2 of the Gflops. It also achieves on-par performance on ShapeNet chairs with the reported DiT-3D-B(32/4) performance with only 1/8 of the Gflops and smaller model. We further boost the performance of DiT-fold by scaling to 512 tokens. Due to restrictions on our training budget, we didn't scale our models to DiT-B, DiT-L, and DiT-XL. Nevertheless, *FoldDiff* demonstrates its superiority over the point-voxel diffusion framework. Please refer to Appendix A.5.1 for more visualizations.

5.3 Implicit priors and DDPMs

Here we demonstrate the necessity of a structured space in both denoising implicit priors and DDPMs, also showcasing the flexibility of *FoldDiff* as a general framework. In total, we experiment on four different options: (1) PointNet denoising implicit prior with unstructured space; (2) UNet denoising implicit prior with

Table 3: Generative performances of denoising implicit priors with unstructured and structured spaces on ShapeNet objects. Structured spaces means we use globally folded reconstructions for training and sampling.

		Chair				Airplane				Car			
Denoiser	Structured Space	1-NNA (↓)		COV (↑)		1-NNA (↓)		COV (↑)		1-NNA (↓)		COV (↑)	
		CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
PointNet	✗	Fails to generate any visible shapes											
2D UNet	✓	**85.73**	**87.24**	**28.85**	**28.40**	**96.13**	**96.09**	**20.82**	**22.63**	**90.62**	**92.14**	**34.09**	**36.93**

Table 4: Generative performances of UNet DDPMs and DiT DDPMs with unstructured and structured spaces on ShapeNet objects. Structured spaces means we use globally or locally folded points for training and sampling.

		Chair				Airplane				Car			
Denoiser	Structured Space	1-NNA (↓)		COV (↑)		1-NNA (↓)		COV (↑)		1-NNA (↓)		COV (↑)	
		CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD	CD	EMD
PointNet++	✗	Fails to generate any visible shapes											
2D UNet	✓	59.29	62.54	42.13	45.92	84.44	85.31	39.51	38.52	67.33	70.03	42.05	40.91
DiT-raw	✗	Fails to generate any visible shapes											
DiT-fold	✓	**55.48**	**53.56**	**48.03**	**51.69**	**72.19**	**66.06**	**47.28**	**51.11**	**58.74**	**52.26**	**48.53**	**53.79**

globally folded structured space; (3) UNet DDPM with globally folded structured space; and (4) DiT DDPM with locally folded structured space. Denoising implicit priors works with slightly different preprocessing steps, where objects are individually normalized into a bounding box of $[-1, 1]^3$. DDPM methods follows the default data preprocessing steps.

Implicit priors with unstructured and structured space. In Sec. 3, we demonstrated that the theoretical foundation of denoising implicit priors fails with unknown permutations. It supports the failures reported by Zhou et al. (2021) and Mo et al. (2023) when they perform diffusion on unstructured point clouds. Our theoretical and empirical results provides for the first time an explanation to the problems they encountered. Here we show the empirical results using object-level point cloud denoisers. Our object-level denoiser is implemented with the segmentation-variant of PointNet (Qi et al., 2017a). Implementation details are covered in Appendix A.2. We use the surface proximity (Equation 5) as training objective, and the model predicts a denoising direction for each point. Langevin stochastic gradient ascent fails to generate any visible shape on such unstructured space of point clouds, agreeing with the empirical findings in Zhou et al. (2021) and Mo et al. (2023). Generative performances are covered in Table 3.

The *FoldDiff* framework offers a structured space for denoising implicit priors. We can prepare a globally folded dataset from ShapeNet Chang et al. (2015) chairs, airplanes, and cars. On each subset, a 2D UNet (with architecture described in Appendix A.2) was trained to denoise the folded objects, treating them as geometry images of $(H \times W \times 3)$. The denoiser is trained to remove a gaussian noise of variance from $[0, 0.8]$. Finally, we apply the Langevin stochastic gradient ascent (Algorithm 1) to sample 3D shapes from the denoiser, with initial parameters $\sigma_0 = 1.0, \sigma_L = 0.01$, $h_0 = 0.01$, and $\beta = 0.1$. As reported in Table 3, a structured space is necessary to sample authentic and diverse shapes from denoising implicit priors.

DDPMs with unstructured and structured space. Following previous works, here we confirm again that DDPMs on unstructured space with PointNet++ as in Zhou et al. (2021) and DiT on raw points as in Mo et al. (2023) fail to generate any visible shapes. Moreover, as discussed in Sec. 4, noise-conditioned denoisers gives a more accurate score estimation for diffusion sampling. We performed experiments on two versions of DDPMs using folded point clouds: (1) 2D UNet-based DDPMs and (2) DiT-based DDPMs. We use the default DiT-fold-S(256/16).

The 2D UNet-based DDPM uses a similar UNet architecture as in 2D UNet-based denoising implicit priors, except being conditioned on noise levels. As reported in Table 4, DDPMs provides better sampling quality and diversity than denoising implicit priors in Table 3. However, since the UNet DDPMs are modeling the manifold of globally folded shapes, the sampling performance is limited by the performance of the

Table 5: Ablation studies. MMD-CD is multiplied with 1×10^{-3}, MMD-EMD is multiplied with 1×10^{-2}.

Folding Configs			1-NNA↓(%)		COV↑(%)		MMD↓	
token source	attention	loss	CD	EMD	CD	EMD	CD	EMD
Chair	✗	CD	56.38	55.83	47.83	**53.45**	2.55	1.45
Chair	✓	CD	56.35	54.39	47.71	51.90	2.59	1.47
Chair,Airplane,Car	✓	CD	55.93	54.15	47.53	52.52	**2.51**	**1.43**
Chair,Airplane,Car	✓	CD	**55.48**	**53.56**	**48.03**	51.69	**2.51**	1.44

(a) Ablating FoldingNets.

Configs		1-NNA↓(%)		COV↑(%)		MMD↓	
Method	EMA	CD	EMD	CD	EMD	CD	EMD
DiT-3D-S(32/4)	✗	68.35	69.18	39.73	40.33	2.90	1.63
DiT-3D-S(32/4)	✓	59.19	55.82	44.96	51.16	2.54	1.49
DiT-fold-S(256/16)	✗	56.85	55.20	45.78	50.60	2.62	1.48
DiT-fold-S(256/16)	✓	**55.48**	**53.56**	**48.03**	**51.69**	**2.51**	**1.44**

(b) Ablating EMA.

autoencoder. DiT-based *FoldDiff* avoids rediscovering a globally autoencoded manifold. As discussed in Sec. 4, DiTs predict the denoising direction for "particle groups" instead of the autoencoded "particles," thus is more flexible than the 2D UNet-based DDPMs with globally folded reconstructions. As reported in Table 4, DiTs generates more authentic and diverse shapes comparing to UNets.

5.4 Ablation studies

Local FoldingNet. Here we ablate our DiT-fold with different local FoldingNets discussed in Sec. 4.2. We evaluate the default DiT-fold-S(256/16) on the chair subset with different FoldingNet configurations. Minimum Matching Distance (MMD) were included as additional metrics to evaluate the quality of generated shapes for experiments. Lower MMD implies better fidelity. The experimental results, summarized in Table 5a, reveal several key insights. As observed, introducing attention mechanisms into the local folding tokenizers is necessary for accurately reconstructing local patches, leading to improved overall performance. Changing the optimization goal from the Chamfer Distance (CD) to the Sinkhorn Distance further boosts the performance, as the Sinkhorn Distance is better at continuously approximating a permutation from reconstructions. Moreover, training the local folding tokenizer on a diverse set of object categories (e.g., chairs, airplanes, and cars) enhances both the accuracy and generalization capability of the learned tokens. This diversity proves to be a critical factor in achieving better performance in DiT-fold.

Exponential Moving Average. Exponential Moving Average (EMA) is a widely adopted weight averaging technique in deep learning, where an EMA of the raw weights is maintained during training and used for evaluation or inference. This approach typically leads to improved generalization compared to using the raw, last-step weights. In our experiments, both the baseline (DiT-3D-S) and our model (DiT-fold-S) utilize EMA to obtain a more robust noise estimator, resulting in better quality and diversity in the generated shapes. As observed in Table 5b, EMA is crucial for enhancing generative quality and diversity. Notably, without using EMA weights, our DiT-fold exhibits more robust performance compared to the point-voxel-based DiT-3D.

6 Conclusion

In this work, we provide a theoretical explanation to the impact of irregularity in point cloud diffusion. To solve this problem, we propose *FoldDiff*, a novel framework for 3D point cloud diffusion with linear complexity. By transforming point cloud diffusion from an unstructured space to a reordered structured space, *FoldDiff* eliminates the need for voxelization and integrates seamlessly with 2D denoising priors and DDPM-based methods. The framework leverages the folding operation to output a reordered reconstruction of the inputting geometry, enabling direct integration with off-the-shelf 2D denoising implicit priors and DDPM-based methods. This facilitates efficient modeling of the distribution of clean 3D objects and high-probability sampling from the learned distribution. Leveraging folded tokens with Diffusion Transformers, *FoldDiff* outperforms voxel-based approaches in both performance and efficiency, offering a new direction for tokenizers and transformer architectures in point cloud modeling.

Future work includes exploring similar frameworks for meshes, as well as integrating texture information into the generative process. Additionally, our evaluations were conducted under a relatively small model setting due to computational constraints (i.e., DiT-S), and while sufficient to show the power of the proposed framework, it mainly lays the groundwork for future research to scale up our results or design optimized DiT variants tailored to the folding operation, potentially unifying approaches for 3D generative modeling with 2D methods.

Acknowledgments

Work supported by NSF, ONR, Simons Foundation, Google, Apple, and Amazon.

References

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning Representations and Generative Models for 3D Point Clouds. In *International Conference on Machine Learning*, 2017. URL `https://api.semanticscholar.org/CorpusID:23102425`.

Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. *CoRR*, abs/1512.03012, 2015.

Marco Cuturi. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Dasith de Silva Edirimuni, Xuequan Lu, Zhiwen Shao, Gang Li, Antonio Robles-Kelly, and Ying He. IterativePFN: True Iterative Point Cloud Filtering. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13530–13539, 2023.

Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré. Interpolating between Optimal Transport and MMD using Sinkhorn Divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2681–2690, 2019.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative Adversarial Networks. *Communications of the ACM*, 63:139 – 144, 2014.

Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. A Papier-Mache Approach to Learning 3D Surface Generation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 216–224, 2018.

Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. Geometry Images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Zahra Kadkhodaie and Eero Simoncelli. Stochastic Solutions for Linear Inverse Problems using the Prior Implicit in a Denoiser. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13242–13254, 2021.

Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. Generalization in Diffusion Models Arises from Geometry-adaptive Harmonic Representations. In *The Twelfth International Conference on Learning Representations*, 2024.

Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. SoftFlow: Probabilistic Framework for Normalizing Flow on Manifolds. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 16388–16397. Curran Associates, Inc., 2020.

Jinwoo Kim, Jae Hyeon Yoo, Juho Lee, and Seunghoon Hong. SetVAE: Learning Hierarchical Composition for Generative Modeling of Set-Structured Data. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15054–15063, 2021.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations*, 2014.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved Variational Inference with Inverse Autoregressive Flow. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Roman Klokov, Edmond Boyer, and Jakob J. Verbeek. Discrete Point Flow Networks for Efficient Point Cloud Generation. In *European Conference on Computer Vision*, 2020.

Ivan Kobyzev, Simon Prince, and Marcus A. Brubaker. Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:3964–3979, 2020.

Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-Voxel CNN for Efficient 3D Deep Learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Shitong Luo and Wei Hu. Differentiable Manifold Reconstruction for Point Cloud Denoising. *Proceedings of the 28th ACM International Conference on Multimedia*, 2020.

Shitong Luo and Wei Hu. Score-Based Point Cloud Denoising. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4563–4572, 2021a.

Shitong Luo and Wei Hu. Diffusion Probabilistic Models for 3D Point Cloud Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021b.

Haggai Maron, Meirav Galun, Noam Aigerman, M. Trope, Nadav Dym, Ersin Yumer, Vladimir G. Kim, and Yaron Lipman. Convolutional Neural Networks on Surfaces via Seamless Toric Covers. *ACM Transactions on Graphics (TOG)*, 36:1 – 10, 2017.

Koichi Miyasawa. An Empirical Bayes Estimator of the Mean of a Normal Population. *Bull. Inst. Internat. Statist.*, 38:181–188, 1961.

Shentong Mo, Enze Xie, Ruihang Chu, Lanqing Hong, Matthias Niessner, and Zhenguo Li. DiT-3D: Exploring Plain Diffusion Transformers for 3D Shape Generation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 67960–67971. Curran Associates, Inc., 2023.

Jiahao Pang, Duanshun Li, and Dong Tian. TearingNet: Point Cloud Autoencoder to Learn Topology-Friendly Representations. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7449–7458, 2020.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Charles R. Qi, Hao Su, Mo Kaichun, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 77–85, 2017a. doi: 10.1109/CVPR.2017.16.

Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 5105–5114, 2017b.

Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J Mitra, and Maks Ovsjanikov. POINT-CLEANNET: Learning to Denoise and Remove Outliers from Dense Point Clouds. In *Computer Graphics Forum*, volume 39, pp. 185–203, 2020.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI*, pp. 234–241, 2014. doi: 10.1007/978-3-319-24574-4_28.

Riccardo Roveri, A. Cengiz Öztireli, Ioana Pandele, and Markus H. Gross. PointProNets: Consolidation of Point Clouds with Convolutional Neural Networks. *Computer Graphics Forum*, 37, 2018.

J. Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3D Neural Field Generation Using Triplane Diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20875–20886, June 2023.

Ayan Sinha, Jing Bai, and Karthik Ramani. Deep Learning 3D Shape Surfaces Using Geometry Images. In *European Conference on Computer Vision*, 2016.

Ayan Sinha, Asim Unmesh, Qi-Xing Huang, and Karthik Ramani. SurfNet: Generating 3D Shape Surfaces Using Deep Residual Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 791–800, 2017.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 2256–2265, 2015.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 3745–3753, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *9th International Conference on Learning Representations, ICLR 2021*, 2021.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, 2017.

Pascal Vincent. A Connection Between Score Matching and Denoising Autoencoders. *Neural Computation*, 23:1661–1674, 2011.

Pascal Vincent, H. Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *International Conference on Machine Learning*, 2008.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions on Graphics (TOG)*, 38:1 – 12, 2018.

Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. Point Cloud Completion by Skip-Attention Network With Hierarchical Folding. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1936–1945, 2020.

Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan. PointFlow: 3D Point Cloud Generation With Continuous Normalizing Flows. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4540–4549, 2019.

Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point Cloud Auto-encoder via Deep Grid Deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 206–215, 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. *ArXiv*, abs/2210.06978, 2022.

Qijian Zhang, Junhui Hou, Yu Qian, Yiming Zeng, Juyong Zhang, and Ying He. Flattening-Net: Deep Regular 2D Representation for 3D Point Cloud Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:9726–9742, 2022.

Linqi Zhou, Yilun Du, and Jiajun Wu. 3D Shape Generation and Completion through Point-Voxel Diffusion. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5806–5815, 2021.

A Appendix

A.1 Proof of Proposition 1

In Sec. 3.2 we explained why gradient denoising in Equation 7 cannot be naively extended to unstructured data. This is formalized through our Proposition 1 below, which implies that Equation 7 holds only if we pick a permutation of the point set that is exactly equal to a true underlying structure. For unstructured data, such permutation matrix is non-retrievable, thus Equation 7 no longer holds. In this appendix we include the proof of this proposition.

Proposition 1. *The denoiser residual* $f(\mathbf{y}) = \hat{\mathbf{x}} - \mathbf{y}$ *is proportional to* $\nabla_{\mathbf{y}} \log P(\mathbf{y})$ *if and only if* $\mathbf{\Pi} = \mathbf{I}_{N\times N}$.

Proof. Starting from Equation 10, we can write

$$\begin{aligned}\nabla_y P(\mathbf{y}|\mathbf{\Pi}) &= \sigma^{-2}\int (\tilde{\mathbf{x}} - \mathbf{y})\, g(y - \tilde{\mathbf{x}}) P(\mathbf{x}) d\mathbf{x}, \\ &= \sigma^{-2}\int (\mathbf{\Pi}\mathbf{x} - \mathbf{y})\, P(\mathbf{y}, \mathbf{x}|\mathbf{\Pi}) d\mathbf{x},\end{aligned}$$

where we use $\tilde{\mathbf{x}} = \mathbf{\Pi}\mathbf{x}$ and replace $g(\mathbf{y} - \mathbf{\Pi}\mathbf{x})$ with $P(\mathbf{y}|\mathbf{x}, \mathbf{\Pi})$ to obtain the second row. Dividing by $\sigma^{-2}P(\mathbf{y}|\mathbf{\Pi})$ gives

$$\begin{aligned}\sigma^2 \nabla_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{\Pi}) &= \int \mathbf{\Pi}\mathbf{x} P(\mathbf{x}|\mathbf{y}, \mathbf{\Pi}) d\mathbf{x} - \int \mathbf{y} P(\mathbf{x}|\mathbf{y}, \mathbf{\Pi}) d\mathbf{x}, \\ &= \mathbf{\Pi}\hat{\mathbf{x}} - \mathbf{y},\end{aligned}$$

where in the first row we used $\nabla_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{\Pi}) = \frac{\nabla_{\mathbf{y}} P(\mathbf{y}|\mathbf{\Pi})}{P(\mathbf{y}|\mathbf{\Pi})}$, and in the second row we used Equation 11. This gives us the estimator $\hat{\mathbf{x}}(\mathbf{y})$ as

$$\hat{\mathbf{x}}(\mathbf{y}) = \mathbf{\Pi}^T \left[\mathbf{y} + \sigma^2 \nabla_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{\Pi})\right]. \tag{12}$$

Therefore, we can evaluate how much $f(\mathbf{y}) = \hat{\mathbf{x}} - \mathbf{y}$ differs from being proportional to $\nabla_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{\Pi})$ as

$$f(\mathbf{y}) - \sigma^2 \nabla_{\mathbf{y}} \log P(\mathbf{y}|\mathbf{\Pi}) = (\mathbf{I} - \mathbf{\Pi})\, \hat{\mathbf{x}}(\mathbf{y}),$$

which is 0 if and only if $\mathbf{\Pi} = \mathbf{I}$. ☐

A.2 Implementation details

Here we cover the implementation details of each model we used for experiments. We will discuss important components for each model. For more details, please refer to our public repo.

Architectural details of unstructured PointNet denoiser. We use a PointNet-based denoiser for denoising implicit priors experiments in unstructured spaces as discussed in Sec. 5.3. The network design follows the segmentation-variant of PointNet (Qi et al., 2017a). The encoder is composed of MLP layers with an input dimension of 3 (spatial coordinates) and output dimensions of [64, 64, 64, 128, 512]. The $N \times 512$-dimensional point-wise features then maxpooled over the N-points to achieve a 512-dimensional 1D vector. The 1D vector is then concatenated with the local feature at dimension 128, creating a global-local mixture of point-wise latent code in $N \times (512 + 128)$. The point-wise latent code is then fed into a decoder of output dimensions [256, 128, 64, 3] to predict the point-wise denoising direction. The loss is computed with the mean-squared-error between the predicted denoising direction and the displacement from the noisy point to the nearest clean point.

Architectural details of hierarchical FoldingNet. This section continues Sec. 4.2 to provide more architectural and training details of the proposed hierarchical folding autoencoder. We first adopt a stack of Set Abstraction modules as proposed in PointNet++ Qi et al. (2017b) to encode information in different levels of resolution. After three Set Abstraction operations, we acquire 3 sets of points in decreasing resolutions N_1,N_2,N_3 and their aggregated features $N_1 \times C_1$,$N_2 \times C_2$,$N_3 \times C_3$. During decoding, the first folding block will reconstruct N_3 points as our base geometry image (GI) I_0. The last-layer features before reconstruction are fed forward. Then we expand base GI P_0 to the same resolution as N_2, and use the second folding block that outputs a difference image to add on $Up(I_0)$ to achieve a higher resolution GI I_1. We iterate this process in our decoder until we get a GI with the same resolution as our inputting point cloud. In our implementation, $N_1 = 512$, $N_2 = 256$, and $N_3 = 64$. The corresponding resolution of GIs are $I_0 = (8, 8)$, $I_1 = (16, 16)$, and $I_2 = (32, 16)$. Set Abstraction blocks and Folding blocks follow the same design in SA-Net Wen et al. (2020). Note that in our architecture, we follow SA-Net to apply cross-attention layers Vaswani et al. (2017) analogously as skip-connections Ronneberger et al. (2014). This design preserves permutation invariance of resulting GIs, as the queries of the cross-attention layers are permutation-invariant features extracted by the encoder. Note that we train seperate hierarchical foldingnet for each category. This is less efficient than the DiT case where a lightweight folding tokenizer can be trained on multi-class objects as a generalized tokenizer and be used for different DiTs.

Architectural details of 2D UNet. This section continues Sec. 4 to provide more architectural and training details of the UNet (Ronneberger et al., 2014) model we applied to model denoiser implicit priors. The UNet contains 4 pairs of downsampling-upampling modules with feature dimensions [32, 64, 128, 256], each pair connected by skip connections. Each module contains 2 convolutional layers with residual connections. The same architecture is used in all experiments for fair comparison. We first construct a GI dataset converted from point clouds that contains 512 points farthest sampled from ShapeNet (Chang et al., 2015) single-class objects, which further normalized into a bounding-box of $[-1, 1]^3$ across the dataset. In Kadkhodaie & Simoncelli (2021), the UNet is trained with Gaussian noise with standard deviations drawn from $\sigma \sim \mathcal{U}[0, 0.4]$ (relative to image intensity range $[0, 1]$). Thus, given our GIs have an intensity range $[-1, 1]$, we train the UNet with absolute noise intensity $\sigma \sim \mathcal{U}[0, 0.8]$.

The UNet used in the diffusion model have similar feature dimensions and architecture, except each layer is conditioned on diffusion time step $t \sim sigmoid(\mathcal{U}[0, 1])$.

Architectural details of DiT-fold and the folding tokenizer. This section continues Sec. 5 and covers implementation details of DiT with our FoldDiff framework. The lightweight folding tokenizer follows the modified architecture as shown in Sec. 4.2. We train a generalized tokenizer across the chair, airplane, car subsets of ShapeNet (Chang et al., 2015), and use this tokenizer for each of the following experiments related to DiT. The tokenizer contains a shallow PointNet-based (Qi et al., 2017a) encoder of output dimensions [64,128,128]. The max-pooled 128-dimensional 1D vector is then concatenated with the coordinates of a $k \times k$ 2D grid, and fed into a MLP to output a point-wise latent code of dimension 128 that mixes the primitive information and the global information. Then we compute the attention scores between the point-wise latent code and the 128-dimensional point-wise feature before max-pooling, further merging local and global information of the point cloud patch. Two folding layers then takes the merged point-wise latent as input and output a reordered reconstruction of the inputting local patch.

A.3 Langevin gradient ascent details

This section continues Sec. 4 to provide a more detailed explanation for the Langevine dynamics sampling algorithm in Kadkhodaie & Simoncelli (2021) which is detailed in Algorithm 1.

In the hyperparameters, σ_0 is the initial noise level, σ_L is the convergence threshold, h_0 controls the step size of each denoising correction, and β controls the proportion of injected noise in each iteration.

As shown in the algorithm, we update noisy image y_t after each denoising step with

$$\mathbf{y}_{t-1} + h_t \mathbf{d}_t + \gamma_t \mathbf{z}_t.$$

Here h_t controls the step size of the denoising correction $\mathbf{d}_t$, while γ_t controls the magnitude of noise injection after each denoising step. Thus the effective noise variance σ_t^2 of $\mathbf{y}_t$ after each iteration is

$$\begin{aligned} \sigma_t^2 &= (1-h_t)^2\sigma_{t-1}^2 + \gamma_t^2, \\ &= (1-\beta h_t)^2\sigma_{t-1}^2. \end{aligned} \tag{13}$$

In the first expression, the first term is the remaining noise variance after the denoiser correction, and the second term is the additional variance from the injected noise. To ensure convergence, σ_t^2 can be rewritten as the second expression in Equation 13 by enforcing $\beta h_t \leq 1$, with $h_t = \frac{h_0 t}{1+h_0(t-1)}$ given an initial parameter $h_0 \in [0,1]$, and $\beta \in [0,1]$ that controls the proportion of injected noise. When $\beta = 1$, we have $\gamma_t^2 = 0$, which indicates no noise injection; when $\beta = 0$, a noise with a variance equivalent to the removed noise is injected. The noise injection amplitude γ_t is relevant to both h_t and β with the expression:

$$\begin{aligned} \gamma_t^2 &= \left[(1-\beta h_t)^2 - (1-h_t)^2\right]\sigma_{t-1}^2, \\ &= \left[(1-\beta h_t)^2 - (1-h_t)^2\right]\left\|f(\mathbf{y}_{t-1})\right\|^2/N, \end{aligned} \tag{14}$$

where $\left\|f(\mathbf{y}_{t-1})\right\|^2/N = \sigma_{t-1}^2$ denotes the magnitude of predicted noise variance, and N is the number of pixels in the image.

To sample high-probability objects from a denoiser, we first sample from random Gaussian noise $y_0 \sim \mathcal{N}(0, \sigma_0^2 I)$. Then the residual of our 2D UNet denoiser is applied as the score $\nabla_y \log p(y)$. In each iteration, we take a small step toward the suggested direction, thus moving closer to the manifold of clean folded reconstructions, then inject an additional Gaussian perturbation to avoid getting stuck in local maxima (Kadkhodaie & Simoncelli, 2021).

Algorithm 1: Coarse-to-fine stochastic ascent method for sampling from the implicit prior of a denoiser, using denoiser residual $f(\mathbf{y}) = \hat{\mathbf{x}}(\mathbf{y}) - \mathbf{y}$. (Kadkhodaie & Simoncelli, 2021)

```
Parameters: $\sigma_0$, $\sigma_L$, $h_0$, $\beta$
Initialization: $t = 1$, draw $\mathbf{y}_0 \sim \mathcal{N}(0, \sigma_0^2 I)$
while $\sigma_{t-1} \leq \sigma_L$ do
    $h_t = \frac{h_0 t}{1+h_0(t-1)}$ ;                                        // step size for denoising step
    $d_t = f(\mathbf{y}_{t-1})$ ;                                             // denoising direction
    $\sigma_t^2 = \frac{||\mathbf{d}_t||^2}{N}$ ;                             // effective noise variance
    $\gamma_t^2 = \left((1-\beta h_t)^2 - (1-h_t)^2\right)\sigma_t^2$ ;       // shrinking noise variance
    Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$ ;                              // sampling Gaussian noise
    $\mathbf{y}_t \leftarrow \mathbf{y}_{t-1} + h_t\mathbf{d}_t + \gamma_t\mathbf{z}_t$ ;   // update data with Langevin dynamics
    $t \leftarrow t + 1$
end
```

A.4 Nearest neighbors of generated shapes

Fig. 5 demonstrates the top 5 nearest neighbors of a sampled chair using both Chamfer Distance (CD) and Earth Moving Distance (EMD). As one can notice, the sampled chair is novel comparing to its nearest neighbors, showcasing a generalized generative behavior of our DiT-fold.

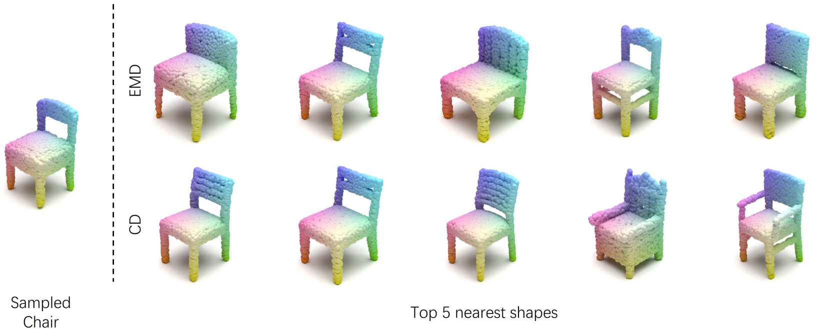


Figure 5: Top 5 nearest neighbors of the sampled chair from the training set. Top: nearest neighbors of Chamfer Distance (CD). Bottom: nearest neighbors of Earth Moving Distance (EMD).

A.5 Qualitative visualizations

A.5.1 More visualizations of synthesized shapes

Here we show more abundant synthesized shapes from DiT-fold trained on ShapeNet Chang et al. (2015) Airplane (Fig. 6), Car (Fig. 7), and Chair (Fig. 8).

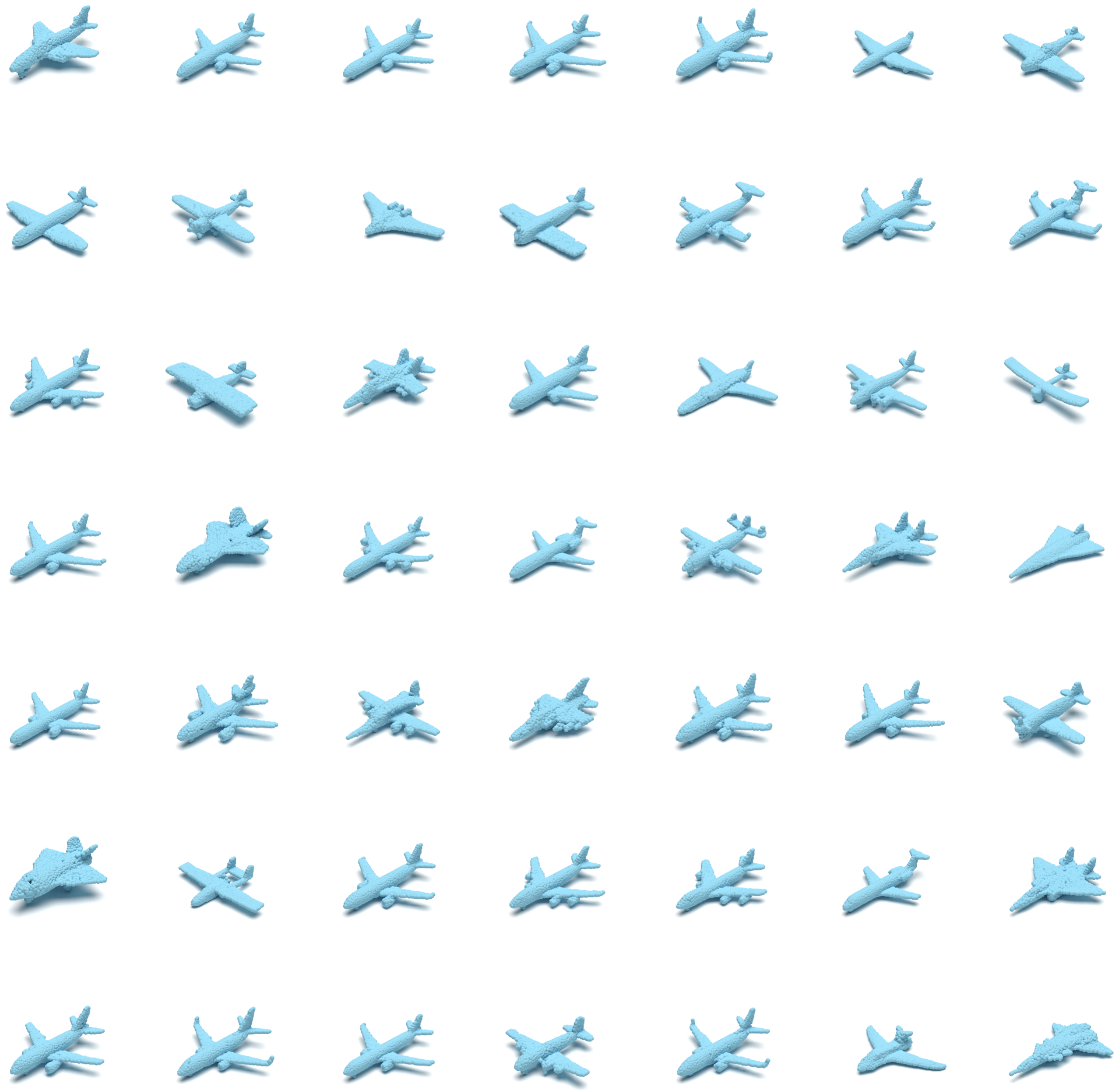


Figure 6: Synthesized airplanes using DiT based on the proposed *FoldDiff* framework.

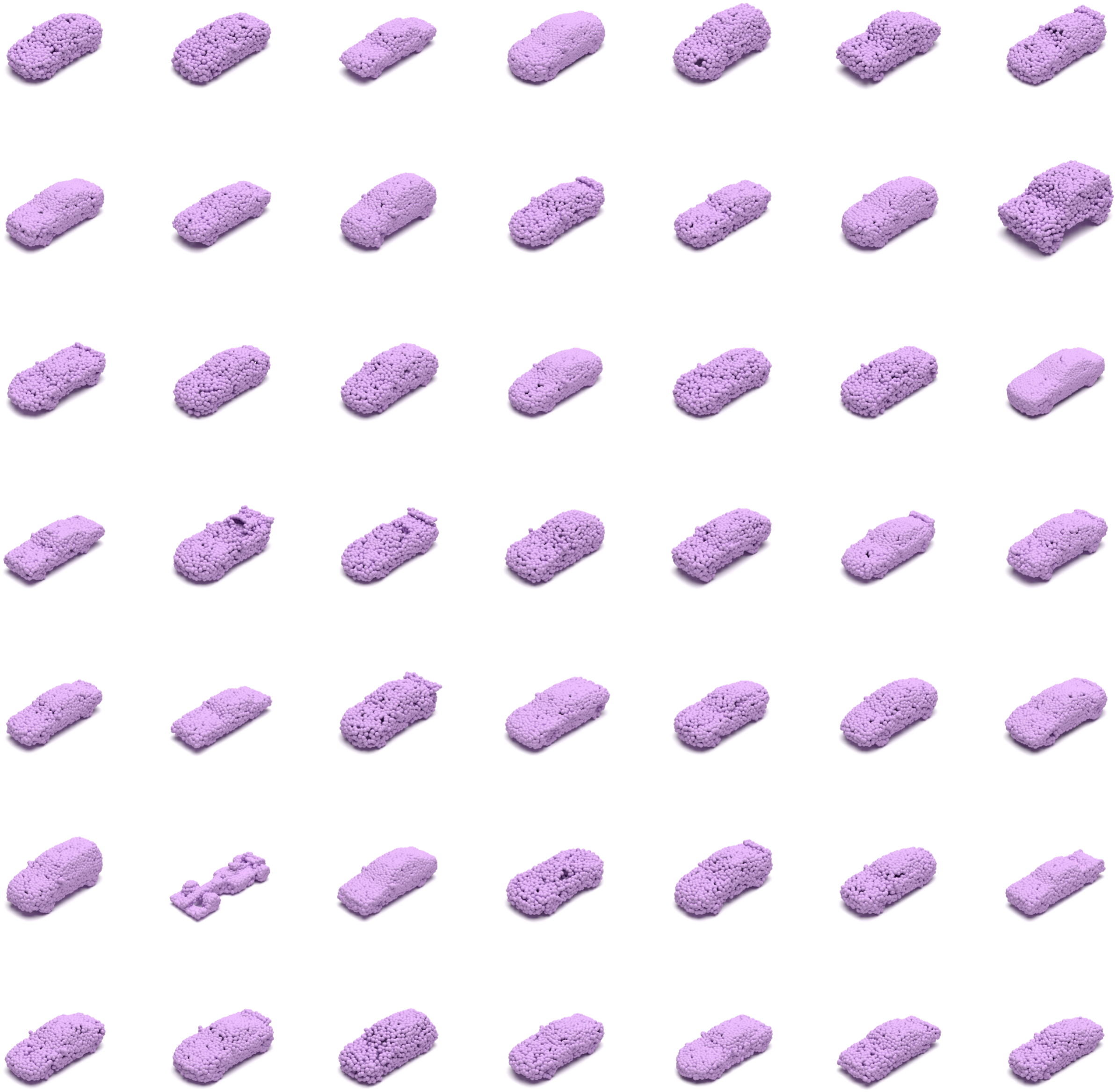


Figure 7: Synthesized cars using DiT based on the proposed *FoldDiff* framework.

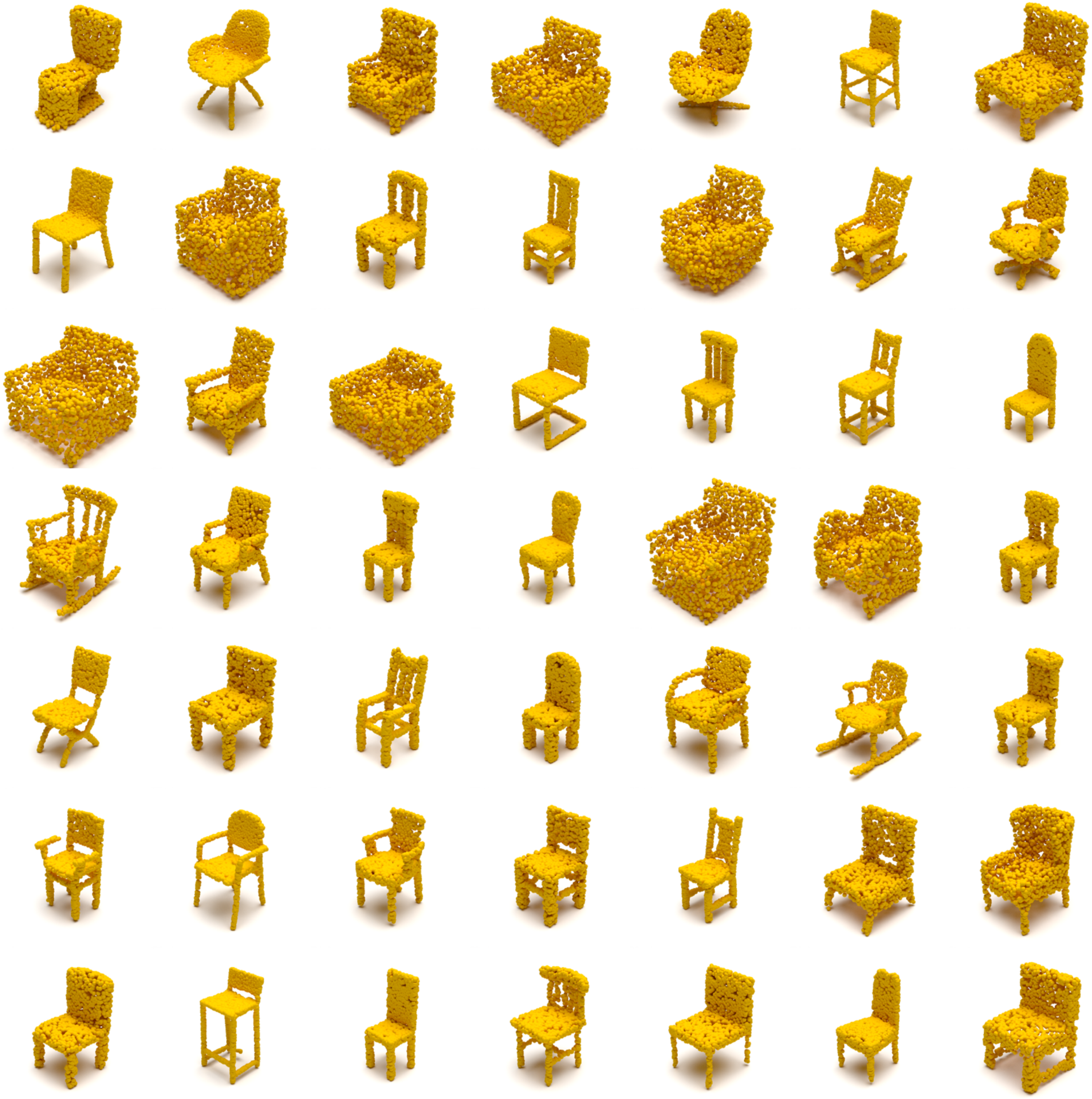


Figure 8: Synthesized chairs using DiT based on the proposed *FoldDiff* framework.