# LION: Latent Point Diffusion Models for 3D Shape Generation

**Xiaohui Zeng**[1,2,3,*] **Arash Vahdat**[1] **Francis Williams**[1]

**Zan Gojcic**[1] **Or Litany**[1] **Sanja Fidler**[1,2,3] **Karsten Kreis**[1]

[1]NVIDIA [2]University of Toronto [3]Vector Institute

{xzeng,avahdat,fwilliams,zgojcic,olitany,sfidler,kkreis}@nvidia.com

## Abstract

Denoising diffusion models (DDMs) have shown promising results in 3D point cloud synthesis. To advance 3D DDMs and make them useful for digital artists, we require *(i)* high generation quality, *(ii)* flexibility for manipulation and applications such as conditional synthesis and shape interpolation, and *(iii)* the ability to output smooth surfaces or meshes. To this end, we introduce the hierarchical *Latent Point Diffusion Model (LION)* for 3D shape generation. LION is set up as a variational autoencoder (VAE) with a hierarchical latent space that combines a global shape latent representation with a point-structured latent space. For generation, we train two hierarchical DDMs in these latent spaces. The hierarchical VAE approach boosts performance compared to DDMs that operate on point clouds directly, while the point-structured latents are still ideally suited for DDM-based modeling. Experimentally, LION achieves state-of-the-art generation performance on multiple ShapeNet benchmarks. Furthermore, our VAE framework allows us to easily use LION for different relevant tasks: LION excels at multimodal shape denoising and voxel-conditioned synthesis, and it can be adapted for text- and image-driven 3D generation. We also demonstrate shape autoencoding and latent shape interpolation, and we augment LION with modern surface reconstruction techniques to generate smooth 3D meshes. We hope that LION provides a powerful tool for artists working with 3D shapes due to its high-quality generation, flexibility, and surface reconstruction. Project page and code: https://nv-tlabs.github.io/LION.

## 1 Introduction

Generative modeling of 3D shapes has extensive applications in 3D content creation and has become an active area of research [1–52]. However, to be useful as a tool for digital artists, generative models of 3D shapes have to fulfill several criteria: **(i)** Generated shapes need to be realistic and of high-quality without artifacts. **(ii)** The model should enable flexible and interactive use and refinement: For example, a user may want to refine a generated shape and synthesize versions with varying details. Or an artist may provide a coarse or noisy input shape, thereby guiding the model to produce multiple realistic high-quality outputs. Similarly, a user may want to interpolate different shapes. **(iii)** The model should output smooth meshes, which are the standard representation in most graphics software.

Existing 3D generative models build on various frameworks, including generative adversarial networks (GANs) [1–23], variational autoencoders (VAEs) [24–30], normalizing flows [31–34], autoregressive models [35–38], and more [39–44]. Most recently, denoising diffusion models (DDMs)

[*] Work done during internship at NVIDIA.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

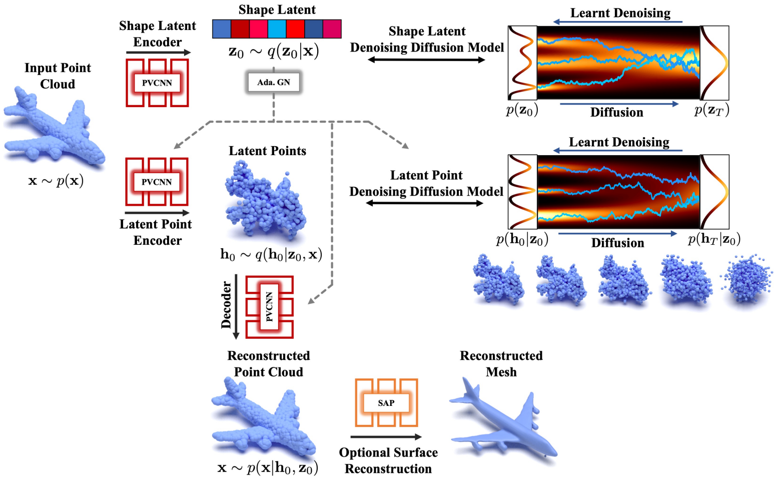

Input Point Cloud
$\mathbf{x} \sim p(\mathbf{x})$
Shape Latent Encoder
PVCNN
Shape Latent
$\mathbf{z}_0 \sim q(\mathbf{z}_0|\mathbf{x})$
Ada. GN
Shape Latent Denoising Diffusion Model
Learnt Denoising
Diffusion
$p(\mathbf{z}_0)$
$p(\mathbf{z}_T)$
PVCNN
Latent Point Encoder
Latent Points
$\mathbf{h}_0 \sim q(\mathbf{h}_0|\mathbf{z}_0, \mathbf{x})$
Latent Point Denoising Diffusion Model
Learnt Denoising
Diffusion
$p(\mathbf{h}_0|\mathbf{z}_0)$
$p(\mathbf{h}_T|\mathbf{z}_0)$
Decoder
PVCNN
Reconstructed Point Cloud
$\mathbf{x} \sim p(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0)$
SAP
Optional Surface Reconstruction
Reconstructed Mesh

Figure 1: *LION* is set up as a hierarchical point cloud VAE with denoising diffusion models over the shape latent and latent point distributions. Point-Voxel CNNs (PVCNN) with adaptive Group Normalization (Ada. GN) are used as neural networks. The latent points can be interpreted as a smoothed version of the input point cloud. *Shape As Points* (SAP) is optionally used for mesh reconstruction.

have emerged as powerful generative models, achieving outstanding results not only on image synthesis [53–64] but also for point cloud-based 3D shape generation [45–47]. In DDMs, the data is gradually perturbed by a diffusion process, while a deep neural network is trained to denoise. This network can then be used to synthesize novel data in an iterative fashion when initialized from random noise [53, 65–67]. However, existing DDMs for 3D shape synthesis struggle with simultaneously satisfying all criteria discussed above for practically useful 3D generative models.

Here, we aim to develop a DDM-based generative model of 3D shapes overcoming these limitations. We introduce the *Latent Point Diffusion Model (LION)* for 3D shape generation (see Fig. 1). Similar to previous 3D DDMs, LION operates on point clouds, but it is constructed as a VAE with DDMs in latent space. LION comprises a hierarchical latent space with a vector-valued global shape latent and another point-structured latent space. The latent representations are predicted with point cloud processing encoders, and two latent DDMs are trained in these latent spaces. Synthesis in LION proceeds by drawing novel latent samples from the hierarchical latent DDMs and decoding back to the original point cloud space. Importantly, we also demonstrate how to augment LION with modern surface reconstruction methods [68] to synthesize smooth shapes as desired by artists. LION has multiple advantages:

**Expressivity:** By mapping point clouds into regularized latent spaces, the DDMs in latent space are effectively tasked with learning a smoothed distribution. This is easier than training on potentially complex point clouds directly [58], thereby improving expressivity. However, point clouds are, in principle, an ideal representation for DDMs. Because of that, we use *latent points*, this is, we keep a point cloud structure for our main latent representation. Augmenting the model with an additional global shape latent variable in a hierarchical manner further boosts expressivity. We validate LION on several popular ShapeNet benchmarks and achieve state-of-the-art synthesis performance.

**Varying Output Types:** Extending LION with *Shape As Points* (SAP) [68] geometry reconstruction allows us to also output smooth meshes. Fine-tuning SAP on data generated by LION's autoencoder reduces synthesis noise and enables us to generate high-quality geometry. LION combines (latent) point cloud-based modeling, ideal for DDMs, with surface reconstruction, desired by artists.

**Flexibility:** Since LION is set up as a VAE, it can be easily adapted for different tasks without retraining the latent DDMs: We can efficiently fine-tune LION's encoders on voxelized or noisy inputs, which a user can provide for guidance. This enables multimodal voxel-guided synthesis and shape denoising. We also leverage LION's latent spaces for shape interpolation and autoencoding. Optionally training the DDMs conditioned on CLIP embeddings enables image- and text-driven 3D generation.

In summary, we make the following contributions: **(i)** We introduce LION, a novel generative model for 3D shape synthesis, which operates on point clouds and is built on a hierarchical VAE framework with two latent DDMs. **(ii)** We validate LION's high synthesis quality by reaching state-of-the-art performance on widely used ShapeNet benchmarks. **(iii)** We achieve high-quality and diverse 3D shape synthesis with LION even when trained jointly over many classes without conditioning. **(iv)** We propose to combine LION with SAP-based surface reconstruction. **(v)** We demonstrate the flexibility of our framework by adapting it to relevant tasks such as multimodal voxel-guided synthesis.

## 2 Background

Traditionally, DDMs were introduced in a discrete-step fashion: Given samples $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ from a data distribution, DDMs use a Markovian fixed forward diffusion process defined as [65, 53]

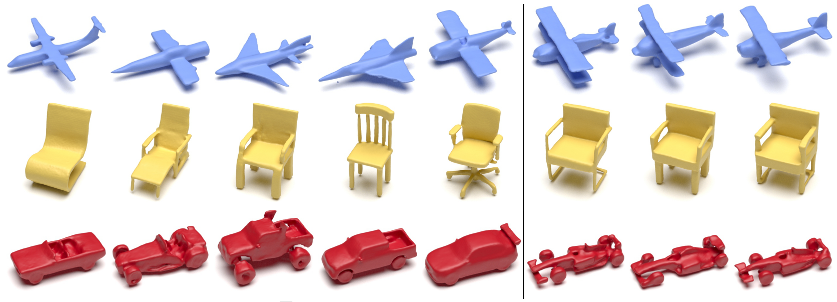


Figure 2: Generated meshes with LION. *Right*: Synthesizing different details by *diffuse-denoise* (see Sec. 3.1) in latent space, while preserving overall shapes.

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \qquad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\boldsymbol{I}), \tag{1}$$

where $T$ denotes the number of steps and $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is a Gaussian transition kernel, which gradually adds noise to the input with a variance schedule $\beta_1, ..., \beta_T$. The $\beta_t$ are chosen such that the chain approximately converges to a standard Gaussian distribution after $T$ steps, $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \boldsymbol{I})$. DDMs learn a parametrized reverse process (model parameters $\boldsymbol{\theta}$) that inverts the forward diffusion:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \mu_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \rho_t^2\boldsymbol{I}). \tag{2}$$

This generative reverse process is also Markovian with Gaussian transition kernels, which use fixed variances $\rho_t^2$. DDMs can be interpreted as latent variable models, where $\mathbf{x}_1, ..., \mathbf{x}_T$ are latents, and the forward process $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$ acts as a fixed approximate posterior, to which the generative $p_{\boldsymbol{\theta}}(\mathbf{x}_{0:T})$ is fit. DDMs are trained by minimizing the variational upper bound on the negative log-likelihood of the data $\mathbf{x}_0$ under $p_{\boldsymbol{\theta}}(\mathbf{x}_{0:T})$. Up to irrelevant constant terms, this objective can be expressed as [53]

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim U\{1,T\},\mathbf{x}_0\sim p(\mathbf{x}_0),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[w(t)||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\alpha_t\mathbf{x}_0 + \sigma_t\boldsymbol{\epsilon}, t)||_2^2\right], \quad w(t) = \frac{\beta_t^2}{2\rho_t^2(1-\beta_t)(1-\alpha_t^2)}, \tag{3}$$

where $\alpha_t = \sqrt{\prod_{s=1}^{t}(1-\beta_s)}$ and $\sigma_t = \sqrt{1-\alpha_t^2}$ are the parameters of the tractable diffused distribution after $t$ steps $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \alpha_t\mathbf{x}_0, \sigma_t^2\boldsymbol{I})$. Furthermore, Eq. (3) employs the widely used parametrization $\mu_{\boldsymbol{\theta}}(\mathbf{x}_t, t) := \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right)$. It is common practice to set $w(t) = 1$, instead of the one in Eq. (3), which often promotes perceptual quality of the generated output. In the objective of Eq. (3), the model $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ is, for all possible steps $t$ along the diffusion process, effectively trained to predict the noise vector $\boldsymbol{\epsilon}$ that is necessary to denoise an observed diffused sample $\mathbf{x}_t$. After training, the DDM can be sampled with ancestral sampling in an iterative fashion:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)) + \rho_t\boldsymbol{\eta}, \tag{4}$$

where $\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{\eta}; \mathbf{0}, \boldsymbol{I})$. This sampling chain is initialized from a random sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \boldsymbol{I})$. Furthermore, the noise injection in Eq. 4 is usually omitted in the last sampling step.

DDMs can also be expressed with a continuous-time framework [67, 69]. In this formulation, the diffusion and reverse generative processes are described by differential equations. This approach allows for deterministic sampling and encoding schemes based on ordinary differential equations (ODEs). We make use of this framework in Sec. 3.1 and we review this approach in more detail in App. B.

## 3 Hierarchical Latent Point Diffusion Models

We first formally introduce LION, then discuss various applications and extensions in Sec. 3.1, and finally recapitulate its unique advantages in Sec. 3.2. See Fig. 1 for a visualization of LION.

We are modeling point clouds $\mathbf{x} \in \mathbb{R}^{3\times N}$, consisting of $N$ points with $xyz$-coordinates in $\mathbb{R}^3$. LION is set up as a hierarchical VAE with DDMs in latent space. It uses a vector-valued global shape latent $\mathbf{z}_0 \in \mathbb{R}^{D_\mathbf{z}}$ and a point cloud-structured latent $\mathbf{h}_0 \in \mathbb{R}^{(3+D_\mathbf{h})\times N}$. Specifically, $\mathbf{h}_0$ is a *latent point cloud* consisting of $N$ points with $xyz$-coordinates in $\mathbb{R}^3$. In addition, each latent point can carry additional $D_\mathbf{h}$ latent features. Training of LION is then performed in two stages—first, we train it as a regular VAE with standard Gaussian priors; then, we train the latent DDMs on the latent encodings.

**First Stage Training.** Initially, LION is trained by maximizing a modified variational lower bound on the data log-likelihood (ELBO) with respect to the encoder and decoder parameters $\boldsymbol{\phi}$ and $\boldsymbol{\xi}$ [70, 71]:

$$\begin{aligned}\mathcal{L}_{\text{ELBO}}(\boldsymbol{\phi}, \boldsymbol{\xi}) = \mathbb{E}_{p(\mathbf{x}), q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x}), q_{\boldsymbol{\phi}}(\mathbf{h}_0|\mathbf{x},\mathbf{z}_0)} \big[&\log p_{\boldsymbol{\xi}}(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0) \\ &- \lambda_\mathbf{z} D_{\text{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})|p(\mathbf{z}_0)\right) - \lambda_\mathbf{h} D_{\text{KL}}\left(q_{\boldsymbol{\phi}}(\mathbf{h}_0|\mathbf{x}, \mathbf{z}_0)|p(\mathbf{h}_0)\right)\big].\end{aligned} \tag{5}$$

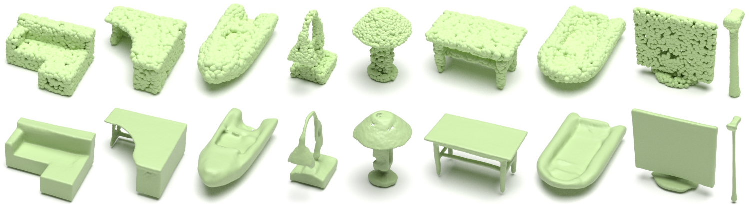


Figure 3: Generated shapes (*top*: point clouds, *bottom*: corresponding meshes) from LION trained jointly over 13 classes of ShapeNet-vol without conditioning (Sec. 5.2).

Here, the global shape latent $\mathbf{z}_0$ is sampled from the posterior distribution $q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})$, which is parametrized by factorial Gaussians, whose means and variances are predicted via an encoder network. The point cloud latent $\mathbf{h}_0$ is sampled from a similarly parametrized posterior $q_{\boldsymbol{\phi}}(\mathbf{h}_0|\mathbf{x}, \mathbf{z}_0)$, while also conditioning on $\mathbf{z}_0$ ($\boldsymbol{\phi}$ denotes the parameters of both encoders). Furthermore, $p_{\boldsymbol{\xi}}(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0)$ denotes the decoder, parametrized as a factorial Laplace distribution with predicted means and fixed unit scale parameter (corresponding to an $L_1$ reconstruction loss). $\lambda_{\mathbf{z}}$ and $\lambda_{\mathbf{h}}$ are hyperparameters balancing reconstruction accuracy and Kullback-Leibler regularization (note that only for $\lambda_{\mathbf{z}} = \lambda_{\mathbf{h}} = 1$ we are optimizing a rigorous ELBO). The priors $p(\mathbf{z}_0)$ and $p(\mathbf{h}_0)$ are $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Also see Fig. 1 again.

**Second Stage Training.** In principle, we could use the VAE's priors to sample encodings and generate new shapes. However, the simple Gaussian priors will not accurately match the encoding distribution from the training data and therefore produce poor samples (*prior hole problem* [58, 72–79]). This motivates training highly expressive latent DDMs. In particular, in the second stage we freeze the VAE's encoder and decoder networks and train two latent DDMs on the encodings $\mathbf{z}_0$ and $\mathbf{h}_0$ sampled from $q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x})$ and $q_{\boldsymbol{\phi}}(\mathbf{h}_0|\mathbf{x}, \mathbf{z}_0)$, minimizing score matching (SM) objectives similar to Eq. (2):

$$\mathcal{L}_{\mathrm{SM}^{\mathbf{z}}}(\boldsymbol{\theta}) = \mathbb{E}_{t\sim U\{1,T\}, p(\mathbf{x}), q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x}), \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)||_2^2, \tag{6}$$

$$\mathcal{L}_{\mathrm{SM}^{\mathbf{h}}}(\boldsymbol{\psi}) = \mathbb{E}_{t\sim U\{1,T\}, p(\mathbf{x}), q_{\boldsymbol{\phi}}(\mathbf{z}_0|\mathbf{x}), q_{\boldsymbol{\phi}}(\mathbf{h}_0|\mathbf{x},\mathbf{z}_0), \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} ||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\psi}}(\mathbf{h}_t, \mathbf{z}_0, t)||_2^2, \tag{7}$$

where $\mathbf{z}_t = \alpha_t \mathbf{z}_0 + \sigma_t \boldsymbol{\epsilon}$ and $\mathbf{h}_t = \alpha_t \mathbf{h}_0 + \sigma_t \boldsymbol{\epsilon}$ are the diffused latent encodings. Furthermore, $\boldsymbol{\theta}$ denotes the parameters of the global shape latent DDM $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)$, and $\boldsymbol{\psi}$ refers to the parameters of the conditional DDM $\boldsymbol{\epsilon}_{\boldsymbol{\psi}}(\mathbf{h}_t, \mathbf{z}_0, t)$ trained over the latent point cloud (note the conditioning on $\mathbf{z}_0$).

**Generation.** With the latent DDMs, we can formally define a hierarchical generative model $p_{\boldsymbol{\xi},\boldsymbol{\psi},\boldsymbol{\theta}}(\mathbf{x}, \mathbf{h}_0, \mathbf{z}_0) = p_{\boldsymbol{\xi}}(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0) p_{\boldsymbol{\psi}}(\mathbf{h}_0|\mathbf{z}_0) p_{\boldsymbol{\theta}}(\mathbf{z}_0)$, where $p_{\boldsymbol{\theta}}(\mathbf{z}_0)$ denotes the distribution of the global shape latent DDM, $p_{\boldsymbol{\psi}}(\mathbf{h}_0|\mathbf{z}_0)$ refers to the DDM modeling the point cloud-structured latents, and $p_{\boldsymbol{\xi}}(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0)$ is LION's decoder. We can hierarchically sample the latent DDMs following Eq. (4) and then translate the latent points back to the original point cloud space with the decoder.

**Network Architectures and DDM Parametrization.** Let us briefly summarize key implementation choices. The encoder networks, as well as the decoder and the latent point DDM, operating on point clouds $\mathbf{x}$, are all implemented based on Point-Voxel CNNs (PVCNNs) [80], following Zhou et al. [46]. PVCNNs efficiently combine the point-based processing of PointNets [81, 82] with the strong spatial inductive bias of convolutions. The DDM modeling the global shape latent uses a ResNet [83] structure with fully-connected layers (implemented as $1\times1$-convolutions). All conditionings on the global shape latent are implemented via adaptive Group Normalization [84] in the PVCNN layers. Furthermore, following Vahdat et al. [58] we use a *mixed score parametrization* in both latent DDMs. This means that the score models are parametrized to predict a residual correction to an analytic standard Gaussian score. This is beneficial since the latent encodings are regularized towards a standard Gaussian distribution during the first training stage (see App. D for all details).

### 3.1 Applications and Extensions

Here, we discuss how LION can be used and extended for different relevant applications.

**Multimodal Generation.** We can synthesize different variations of a given shape, enabling multimodal generation in a controlled manner: Given a shape, i.e., its point cloud $\mathbf{x}$, we encode it into latent space. Then, we diffuse its encodings $\mathbf{z}_0$ and $\mathbf{h}_0$ for a small number of steps $\tau < T$ towards intermediate $\mathbf{z}_\tau$ and $\mathbf{h}_\tau$ along the diffusion process such that only local details are destroyed. Running the reverse generation process from this intermediate $\tau$, starting at $\mathbf{z}_\tau$ and $\mathbf{h}_\tau$, leads to variations of the original shape with different details (see, for instance, Fig. 2). We refer to this procedure as *diffuse-denoise* (details in App. C.1). Similar techniques have been used for image editing [85].

**Encoder Fine-tuning for Voxel-Conditioned Synthesis and Denoising.** In practice, an artist using a 3D generative model may have a rough idea of the desired shape. For instance, they may be able to quickly construct a coarse voxelized shape, to which the generative model then adds realistic details.

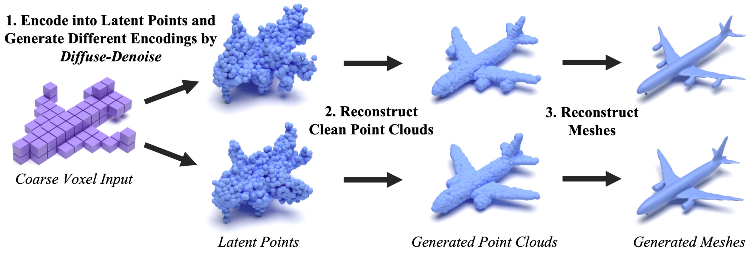

1. Encode into Latent Points and Generate Different Encodings by *Diffuse-Denoise*
2. Reconstruct Clean Point Clouds
3. Reconstruct Meshes
Coarse Voxel Input
Latent Points
Generated Point Clouds
Generated Meshes

Figure 4: Voxel-guided synthesis with *LION*. We run *diffuse-denoise* in latent space (see Sec. 3.1) to generate diverse plausible clean shapes.

In LION, we can support such applications: using a similar ELBO as in Eq. (5), but with a frozen decoder, we can fine-tune LION's encoder networks to take voxelized shapes as input (we simply place points at the voxelized shape's surface) and map them to the corresponding latent encodings $\mathbf{z}_0$ and $\mathbf{h}_0$ that reconstruct the original non-voxelized point cloud. Now, a user can utilize the fine-tuned encoders to encode voxelized shapes and generate plausible detailed shapes. Importantly, this can be naturally combined with the *diffuse-denoise* procedure to clean up imperfect encodings and to generate different possible detailed shapes (see Fig. 4).

Furthermore, this approach is general. Instead of voxel-conditioned synthesis, we can also fine-tune the encoder networks on noisy shapes to perform multimodal shape denoising, also potentially combined with *diffuse-denoise*. LION supports these applications easily without re-training the latent DDMs due to its VAE framework with additional encoders and decoders, in contrast to previous works that train DDMs on point clouds directly [46, 47]. See App. C.2 for technical details.

**Shape Interpolation.** LION also enables shape interpolation: We can encode different point clouds into LION's hierarchical latent space and use the *probability flow ODE* (see App. B) to further encode into the latent DDMs' Gaussian priors, where we can safely perform spherical interpolation and expect valid shapes along the interpolation path. We can use the intermediate encodings to generate the interpolated shapes (see Fig. 7; details in App. C.3).

**Surface Reconstruction.** While point clouds are an ideal 3D representation for DDMs, artists may prefer meshed outputs. Hence, we propose to combine LION with modern geometry reconstruction methods (see Figs. 2, 4 and 5). We use *Shape As Points* (SAP) [68], which is based on differentiable Poisson surface reconstruction and can be trained to extract smooth meshes from noisy point clouds. Moreover, we fine-tune SAP on training data generated by LION's autoencoder to better adjust SAP to the noise distribution in point clouds generated by LION. Specifically, we take clean shapes, encode them into latent space, run a few steps of *diffuse-denoise* that only slightly modify some details, and decode back. The *diffuse-denoise* in latent space results in noise in the generated point clouds similar to what is observed during unconditional synthesis (details in App. C.4).

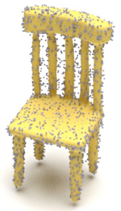


Figure 5: Reconstructing a mesh from LION's generated points.

### 3.2 LION's Advantages

We now recapitulate LION's unique advantages. LION's structure as a hierarchical VAE with latent DDMs is inspired by latent DDMs on images [57, 58, 77]. This framework has key benefits:

**(i) Expressivity:** First training a VAE that regularizes the latent encodings to approximately fall under standard Gaussian distributions, which are also the DDMs' equilibrium distributions towards which the diffusion processes converge, results in an easier modeling task for the DDMs: They have to model only the remaining mismatch between the actual encoding distributions and their own Gaussian priors [58]. This translates into improved expressivity, which is further enhanced by the additional decoder network. However, point clouds are, in principle, an ideal representation for the DDM framework, because they can be diffused and denoised easily and powerful point cloud processing architectures exist. Therefore, LION uses *point cloud latents* that combine the advantages of both latent DDMs and 3D point clouds. Our point cloud latents can be interpreted as smoothed versions of the original point clouds that are easier to model (see Fig. 1). Moreover, the hierarchical VAE setup with an additional global shape latent increases LION's expressivity even further and results in natural disentanglement between overall shape and local details captured by the shape latents and latent points (Sec. 5.2).

**(ii) Flexibility:** Another advantage of LION's VAE framework is that its encoders can be fine-tuned for various relevant tasks, as discussed previously, and it also enables easy shape interpolation. Other 3D point cloud DDMs operating on point clouds directly [47, 46] do not offer simultaneously as much flexibility and expressivity out-of-the-box (see quantitative comparisons in Secs. 5.1 and 5.4).

**(iii) Mesh Reconstruction:** As discussed, while point clouds are ideal for DDMs, artists likely prefer meshed outputs. As explained above, we propose to use LION together with modern surface recon-

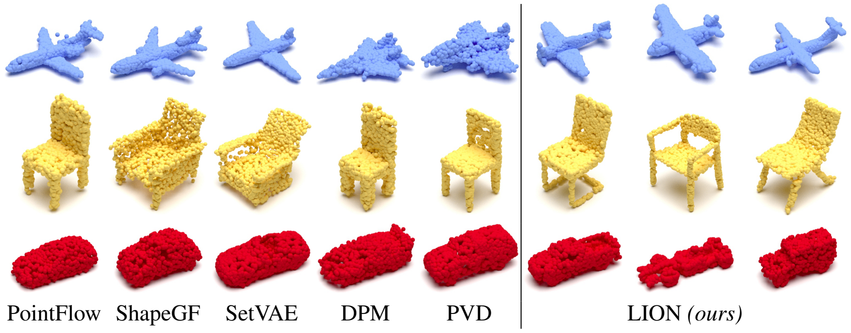


Figure 6: Unconditional shape generation with 2,048 points for *airplane*, *car* and *chair* classes (class-specific models trained on PointFlow's ShapeNet data with global normalization).

struction techniques [68], again combining the best of both worlds—a point cloud-based VAE backbone ideal for DDMs, and smooth geometry reconstruction methods operating on the synthesized point clouds to generate practically useful smooth surfaces, which can be easily transformed into meshes.

## 4 Related Work

We are building on DDMs [53, 65–67], which have been used most prominently for image [53–63] and speech synthesis [86–91]. We train DDMs in latent space, an idea that has been explored for image [57, 58, 77] and music [92] generation, too. However, these works did not train separate conditional DDMs. Hierarchical DDM training has been used for generative image upsampling [54], text-to-image generation [63, 64], and semantic image modeling [60]. Most relevant among these works is Preechakul et al. [60], which extracts a high-level semantic representation of an image with an auxiliary encoder and then trains a DDM that adds details directly in image space. We are the first to explore related concepts for 3D shape synthesis and we also train both DDMs in latent space. Furthermore, DDMs and VAEs have also been combined in such a way that the DDM improves the output of the VAE [93].

Most related to LION are "Point-Voxel Diffusion" (PVD) [46] and "Diffusion Probabilistic Models for 3D Point Cloud Generation" (DPM) [47]. PVD trains a DDM directly on point clouds, and our decision to use PVCNNs is inspired by this work. DPM, like LION, uses a shape latent variable, but models its distribution with Normalizing Flows [94, 95], and then trains a weaker point-wise conditional DDM directly on the point cloud data (this allows DPM to learn useful representations in its latent variable, but sacrifices generation quality). As we show below, neither PVD nor DPM easily enables applications such as multimodal voxel-conditioned synthesis and denoising. Furthermore, LION achieves significantly stronger generation performance. Finally, neither PVD nor DPM reconstructs meshes from the generated point clouds. Point cloud and 3D shape generation have also been explored with other generative models: PointFlow [31], DPF-Net [33] and SoftFlow [32] rely on Normalizing Flows [94–97]. SetVAE [29] treats point cloud synthesis as set generation and uses VAEs. ShapeGF [45] learns distributions over gradient fields that model shape surfaces. Both IM-GAN [7], which models shapes as neural fields, and l-GAN [2] train GANs over latent variables that encode the shapes, similar to other works [3], while r-GAN [2] generates point clouds directly. PDGN [52] proposes progressive deconvolutional networks within a point cloud GAN. SP-GAN [19] uses a spherical point cloud prior. Other progressive [22, 37] and graph-based architectures [4, 6] have been used, too. Also generative cellular automata (GCAs) can be employed for voxel-based 3D shape generation [43]. In orthogonal work, point cloud DDMs have been used for generative shape completion [46, 98].

Recently, image-driven [8–16, 44] training of 3D generative models as well as text-driven 3D generation [34, 49–51] have received much attention. These are complementary directions to ours; in fact, augmenting LION with additional image-based training or including text-guidance are promising future directions. Finally, we are relying on SAP [68] for mesh generation. Strong alternative approaches for reconstructing smooth surfaces from point clouds exist [99–103].

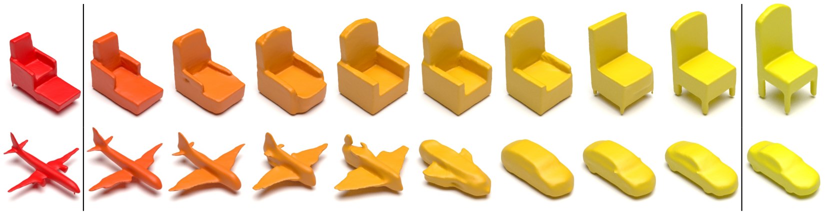


Figure 7: Interpolating different shapes by interpolating their encodings in the standard Gaussian priors of LION's latent DDMs (details in App. C.3).

Table 1: Generation metrics (1-NNA↓) on *airplane*, *chair*, *car* categories from ShapeNet dataset from PointFlow [31]. Training and test data normalized globally into [-1, 1].

| | Airplane | | Chair | | Car | |
|---|---|---|---|---|---|---|
| | CD | EMD | CD | EMD | CD | EMD |
| r-GAN [2] | 98.40 | 96.79 | 83.69 | 99.70 | 94.46 | 99.01 |
| l-GAN (CD) [2] | 87.30 | 93.95 | 68.58 | 83.84 | 66.49 | 88.78 |
| l-GAN (EMD) [2] | 89.49 | 76.91 | 71.90 | 64.65 | 71.16 | 66.19 |
| PointFlow [31] | 75.68 | 70.74 | 62.84 | 60.57 | 58.10 | 56.25 |
| SoftFlow [32] | 76.05 | 65.80 | 59.21 | 60.05 | 64.77 | 60.09 |
| SetVAE [29] | 76.54 | 67.65 | 58.84 | 60.57 | 59.94 | 59.94 |
| DPF-Net [33] | 75.18 | 65.55 | 62.00 | 58.53 | 62.35 | 54.48 |
| DPM [47] | 76.42 | 86.91 | 60.05 | 74.77 | 68.89 | 79.97 |
| PVD [46] | 73.82 | 64.81 | 56.26 | 53.32 | 54.55 | 53.83 |
| LION *(ours)* | **67.41** | **61.23** | **53.70** | **52.34** | **53.41** | **51.14** |

Table 2: Generation results (1-NNA↓) on ShapeNet dataset from PointFlow [31]. All data normalized individually into [-1, 1].

| | Airplane | | Chair | | Car | |
|---|---|---|---|---|---|---|
| | CD | EMD | CD | EMD | CD | EMD |
| TreeGAN [6] | 97.53 | 99.88 | 88.37 | 96.37 | 89.77 | 94.89 |
| ShapeGF [45] | 81.23 | 80.86 | 58.01 | 61.25 | 61.79 | 57.24 |
| SP-GAN [19] | 94.69 | 93.95 | 72.58 | 83.69 | 87.36 | 85.94 |
| PDGN [52] | 94.94 | 91.73 | 71.83 | 79.00 | 89.35 | 87.22 |
| GCA [43] | 88.15 | 85.93 | 64.27 | 64.50 | 70.45 | 64.20 |
| LION *(ours)* | **76.30** | **67.04** | **56.50** | **53.85** | **59.52** | **49.29** |

Table 3: Results (1-NNA↓) on ShapeNet-vol.

| | Airplane | | Chair | | Car | |
|---|---|---|---|---|---|---|
| | CD | EMD | CD | EMD | CD | EMD |
| IM-GAN [7] | 79.70 | 77.85 | 57.09 | 58.20 | 88.92 | 84.58 |
| DPM [47] | 83.04 | 96.04 | 61.96 | 74.96 | 77.30 | 87.12 |
| PVD [46] | 66.46 | 56.06 | 61.89 | 57.90 | 64.49 | 55.74 |
| LION *(ours)* | **53.47** | **53.84** | **52.07** | **48.67** | **54.81** | **50.53** |

## 5 Experiments

We provide an overview of our most interesting experimental results in the main paper. All experiment details and extensive additional experiments can be found in App. E and App. F, respectively.

### 5.1 Single-Class 3D Shape Generation

**Datasets.** To compare LION against existing methods, we use ShapeNet [104], the most widely used dataset to benchmark 3D shape generative models. Following previous works [31, 46, 47], we train on three categories: *airplane*, *chair*, *car*. Also like previous methods, we primarily rely on PointFlow's [31] dataset splits and preprocsssing. It normalizes the data globally across the whole dataset. However, some baselines require per-shape normalization [19, 43, 45, 52]; hence, we also train on such data. Furthermore, training SAP requires signed distance fields (SDFs) for volumetric supervision, which the PointFlow data does not offer. Hence, for simplicity we follow Peng et al. [68, 101] and also use their data splits and preprocessing, which includes SDFs.We train LION, DPM, PVD, and IM-GAN (which synthesizes shapes as SDFs) also on this dataset version (denoted as *ShapeNet-vol* here). This data is also per-shape normalized. Dataset details in App. E.1.

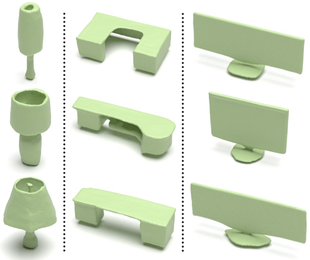


Figure 8: Samples from our unconditional 13-class model: In each column, we use the same global shape latent $\mathbf{z}_0$.

**Evaluation.** Model evaluation follows previous works [31, 46]. Various metrics to evaluate point cloud generative models exist, with different advantages and disadvantages, discussed in detail by Yang et al. [31]. Following recent works [31, 46], we use *1-NNA* (with both Chamfer distance (CD) and earth mover distance (EMD)) as our main metric. It quantifies the distributional similarity between generated shapes and validation set and measures both quality and diversity [31]. For fair comparisons, all metrics are computed on point clouds, not meshed outputs (App. E.2 discusses different metrics; further results on coverage (COV) and minimum matching distance (MMD) in App. F.2).

**Results.** Samples from LION are shown in Fig. 6 and quantitative results in Tabs. 1-3 (see Sec. 4 for details about baselines—to reduce the number of baselines to train, we are focusing on the most recent and competitive ones). LION outperforms all baselines and achieves state-of-the-art performance on all classes and dataset versions. Importantly, we outperform both PVD and DPM, which also leverage DDMs, by large margins. Our samples are diverse and appear visually pleasing.

Table 4: Generation results (1-NNA↓) of LION trained jointly on 13 classes of ShapeNet-vol.

| Model | CD | EMD |
|---|---|---|
| TreeGAN [6] | 96.80 | 96.60 |
| PointFlow [31] | 63.25 | 66.05 |
| ShapeGF [45] | 55.65 | 59.00 |
| SetVAE [29] | 79.25 | 95.25 |
| PDGN [52] | 71.05 | 86.00 |
| DPF-Net [33] | 67.10 | 64.75 |
| DPM [47] | 62.30 | 86.50 |
| PVD [46] | 58.65 | 57.85 |
| LION *(ours)* | **51.85** | **48.95** |

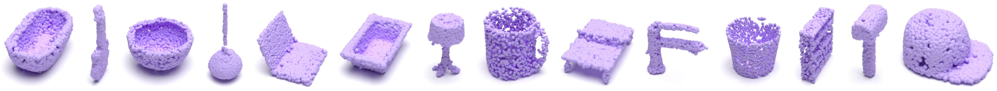


Figure 9: Generated point clouds from LION trained jointly over 55 classes of ShapeNet-vol (no conditioning).

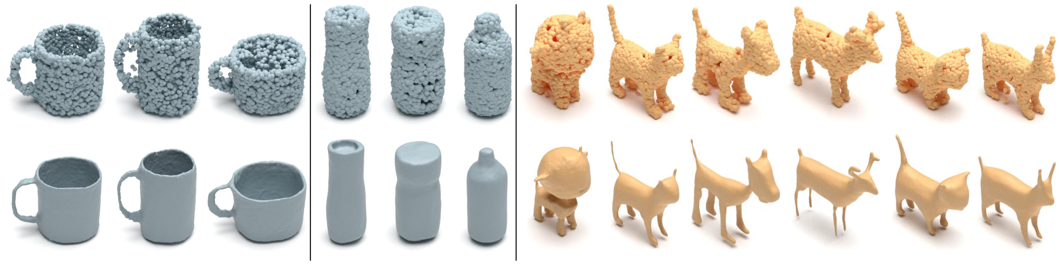


Figure 10: Samples from LION trained on ShapeNet's Mug and Bottle classes, and on Turbosquid animals.

**Mesh Reconstruction.** As explained in Sec. 3.1, we combine LION with mesh reconstruction, to directly synthesize practically useful meshes. We show generated meshes in Fig. 2, which look smooth and of high quality. In Fig. 2, we also visually demonstrate how we can vary the local details of synthesized shapes while preserving the overall shape with our *diffuse-denoise* technique (Sec. 3.1). Details about the number of diffusion steps for all *diffuse-denoise* experiments are in App. E.

**Shape Interpolation.** As discussed in Sec. 3.1, LION also enables shape interpolation, potentially useful for shape editing applications. We show this in Fig. 7, combined with mesh reconstruction. The generated shapes are clean and semantically plausible along the entire interpolation path. In App. F.12.1, we also show interpolations from PVD [46] and DPM [47] for comparison.

### 5.2 Many-class Unconditional 3D Shape Generation

**13-Class LION Model.** We train a LION model *jointly without any class conditioning* on 13 different categories (*airplane, chair, car, lamp, table, sofa, cabinet, bench, telephone, loudspeaker, display, watercraft, rifle*) from ShapeNet (ShapeNet-vol version). Training a single model without conditioning over such diverse shapes is challenging, as the data distribution is highly complex and multimodal. We show LION's generated samples in Fig. 3, including meshes: LION synthesizes high-quality and diverse plausible shapes even when trained on such complex data. We report the model's quantitative generation performance in Tab. 4, and we also trained various strong baseline methods under the same setting for comparison. We find that LION significantly outperforms all baselines by a large margin. We further observe that the hierarchical VAE architecture of LION becomes crucial: The shape latent variable $\mathbf{z}_0$ captures global shape, while the latent points $\mathbf{h}_0$ model details. This can be seen in Fig. 8: we show samples when fixing the global shape latent $\mathbf{z}_0$ and only sample $\mathbf{h}_0$ (details in App. F.3).

**55-Class LION Model.** Encouraged by these results, we also trained a LION model again *jointly without any class conditioning* on all 55 different categories from ShapeNet. Note that we did on purpose not use class-conditioning in these experiments to create a difficult 3D generation task and thereby explore LION's scalability to highly complex and multimodal datasets. We show generated point cloud samples in Fig. 9 (we did not train an SAP model on the 55 classes data): LION synthesizes high-quality and diverse shapes. It can even generate samples from the *cap* class, which contributes with only 39 training data samples, indicating that LION has an excellent mode coverage that even includes the very rare classes. To the best of our knowledge no previous 3D shape generative models have demonstrated satisfactory generation performance for such diverse and multimodal 3D data without relying on conditioning information (details in App. F.4). In conclusion, we observe that LION out-of-the-box easily scales to highly complex multi-category shape generation.

### 5.3 Training LION on Small Datasets

Next, we explore whether LION can also be trained successfully on very small datasets. To this end, we train models on the Mug and Bottle ShapeNet classes. The number of training samples

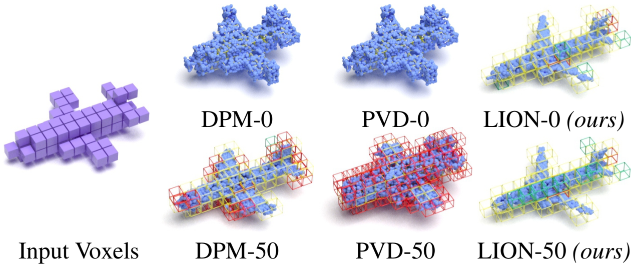


Figure 11: Voxel-guided synthesis. We show different methods with 0 and 50 steps of *diffuse-denoise*. Voxelizations of generated points are also shown: Yellow boxes indicate generated points correctly fill input voxels, green boxes indicate voxels should be filled but are left empty, red boxes indicate extra voxels.

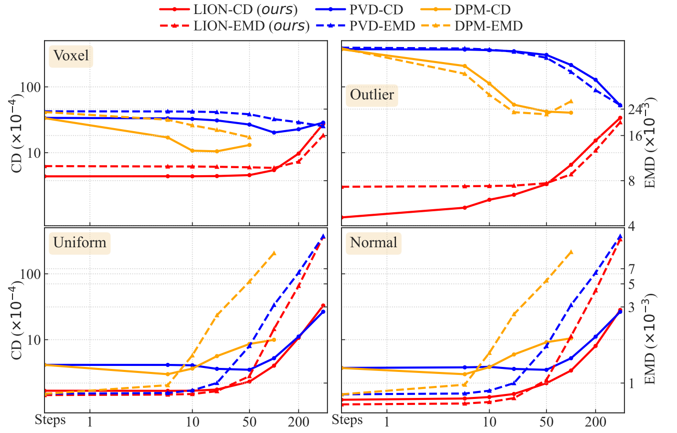


Figure 12: Reconstruction metrics with respect to clean inputs for *airplane* category (lower is better) when guiding synthesis with voxelized or noisy inputs (using uniform, outlier, and normal noise, see App. F.7). $x$-axes denote number of *diffuse-denoise* steps.

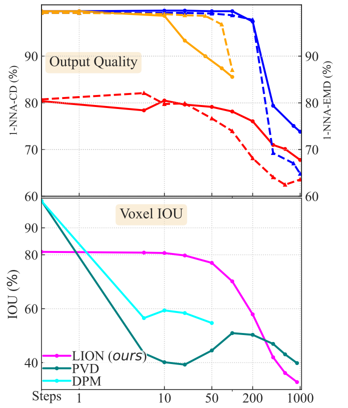


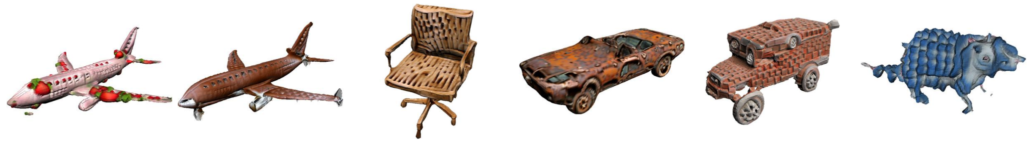
Figure 13: Voxel-guided generation. Quality metrics for output points (lower is better) and voxel IOU with respect to input (higher is better). $x$-axes denote *diffuse-denoise* steps.

is 149 and 340, respectively, which is much smaller than the common classes like chair, car and airplane. Furthermore, we also train LION on 553 animal assets from the TurboSquid data repository. Generated shapes from the three models are shown in Fig. 10. LION is able to generate correct mugs and bottles as well as diverse and high-quality animal shapes. We conclude that LION also performs well even when training in the challenging low-data setting (details in Apps. F.5 and F.6).

### 5.4 Voxel-guided Shape Synthesis and Denoising with Fine-tuned Encoders

Next, we test our strategy for multimodal voxel-guided shape synthesis (see Sec. 3.1) using the *airplane* class LION model (experiment details in App. E, more experiments in App. F.7). We first voxelize our training set and fine-tune our encoder networks to produce the correct encodings to decode back the original shapes. When processing voxelized shapes with our point-cloud networks, we sample points on the surface of the voxels. As discussed, we can use different numbers of *diffuse-denoise* steps in latent space to generate various plausible shapes and correct for poor encodings. Instead of voxelizations, we can also consider different noisy inputs (we use *normal*, *uniform*, and *outlier* noise, see App. F.7) and achieve multimodal denoising with the same approach. The same tasks can be attempted with the important DDM-based baselines PVD and DPM, by directly—not in a latent space—diffusing and denoising voxelized (converted to point clouds) or noisy point clouds.

Fig. 12 shows the reconstruction performance of LION, DPM and PVD for different numbers of *diffuse-denoise* steps (we voxelized or noised the validation set to measure this). We see that for almost all inputs—voxelized or different noises—LION performs best. PVD and DPM perform acceptably for normal and uniform noise, which is similar to the noise injected during training of their DDMs, but perform very poorly for outlier noise or voxel inputs, which is the most relevant case to us, because voxels can be easily placed by users. It is LION's unique framework with additional fine-tuned encoders in its VAE and only latent DDMs that makes this possible. Performing more *diffuse-denoise* steps means that more independent, novel shapes are generated. These will be cleaner and of higher quality, but also correspond less to the noisy or voxel inputs used for guidance. In Fig. 13, we show this trade-off for the voxel-guidance experiment (other experiments in App. F.7), where *(top)* we measured the outputs' synthesis quality by calculating 1-NNA with respect to the validation set, and *(bottom)* the average intersection over union (IOU) between the input voxels and the voxelized outputs. We gener-



*an airplane made of strawberries* *an airplane made of fabric leather* *a chair made of wood* *a car made of* rusty metal *a car made of brick* *a denim fabric animal*

Figure 14: We apply Text2Mesh [49] on meshes generated by LION. In Text2Mesh, textures are generated and meshes refined such that rendered images of the 3D objects are aligned with user-provided text prompts [105].

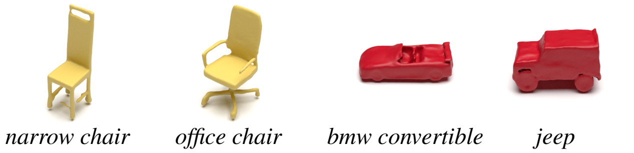


Figure 16: Text-driven shape generation of chairs and cars with LION. Bottom row is the text prompt used as input.

ally see a trade-off: More diffuse-denoise steps result in lower 1-NNA (better quality), but also lower IOU. LION strikes the best balance by a large gap: Its additional encoder network directly generates plausible latent encodings from the perturbed inputs that are both high quality and also correspond well to the input. This trade-off is visualized in Fig. 11 for LION, DPM, and PVD, where we show generated point clouds and voxelizations (note that performing no diffuse-denoise at all for PVD and DPM corresponds to simply keeping the input, as these models' DDMs operate directly on point clouds). We see that running 50 *diffuse-denoise* steps to generate diverse outputs for DPM and especially PVD results in a significant violation of the input voxelization. In contrast, LION generates realistic outputs that also obey the driving voxels. Overall, LION wins out both in this task and also in unconditional generation with large gaps over these previous DDM-based point cloud generative models. We conclude that LION does not only offer state-of-the-art 3D shape generation quality, but is also very versatile. Note that guided synthesis can also be combined with mesh reconstruction, as shown in Fig. 4.

### 5.5 Sampling Time

While our main experiments use 1,000-step DDPM-based synthesis, which takes $\approx 27.12$ seconds, we can significantly accelerate generation without significant loss in quality. Using DDIM-based sampling [106], we can generate high quality shapes in under one second (Fig. 15), which would enable real-time interactive applications. More analyses in App. F.9.

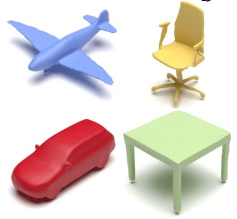


Figure 15: 25-step DDIM [106] samples (0.89 seconds per shape).

### 5.6 Overview of Additional Experiments in Appendix

**(i)** In App. F.1, we perform various ablation studies. The experiments quantitatively validate LION's architecture choices and the advantage of our hierarchical VAE setup with conditional latent DDMs. **(ii)** In App. F.8, we measure LION's autoencoding performance. **(iii)** To demonstrate the value of directly outputting meshes, in App. F.10 we use Text2Mesh [49] to generate textures based on text prompts for synthesized LION samples (Fig. 14). This would not be possible, if we only generated point clouds. **(iv)** To qualitatively show that LION can be adapted easily to other relevant tasks, in App. F.11 we condition LION on CLIP embeddings of the shapes' rendered images, following CLIP-Forge [34] (Fig. 16). This enables text-driven 3D shape generation and single view 3D reconstruction (Fig. 17). **(v)** We also show many more samples (Apps. F.2-F.6) and shape interpolations (App. F.12) from our models, more examples of voxel-guided and noise-guided synthesis (App. F.7), and we further analyze our 13-class LION model (App. F.3.2).

## 6 Conclusions

We introduced LION, a novel generative model of 3D shapes. LION uses a VAE framework with hierarchical DDMs in latent space and can be combined with SAP for mesh generation. LION achieves state-of-the-art shape generation performance and enables applications such as voxel-conditioned synthesis, multimodal shape denoising, and shape interpolation. LION is currently trained on 3D point clouds only and can not directly generate textured shapes. A promising extension would be to include image-based training by incorporating neural or differentiable rendering [17, 107–111] and to also synthesize textures [16, 112–114]. Furthermore, LION currently focuses on single object generation only. It would be interesting to extend it to full 3D scene synthesis. Moreover, synthesis could be further accelerated by building on works on accelerated sampling from DDMs [61, 62, 67, 106, 115–121].

**Broader Impact.** We believe that LION can potentially improve 3D content creation and assist the workflow of digital artists. We designed LION with such applications in mind and hope that it can grow into a practical tool enhancing artists' creativity. Although we do not see any immediate negative use-cases for LION, it is important that practitioners apply an abundance of caution to mitigate impacts given generative modeling more generally can also be used for malicious purposes, discussed for instance in Vaccari and Chadwick [122], Nguyen et al. [123], Mirsky and Lee [124].

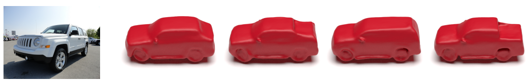


Figure 17: Single view 3D reconstructions of a car from an RGB image. LION can generate multiple plausible outputs using our *diffuse-denoise* technique.

## References

[1] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, 2016.

[2] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICML*, 2018.

[3] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018.

[4] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *International Conference on Learning Representations (ICLR) 2019*, 2019.

[5] Wenlong Huang, Brian Lai, Weijian Xu, and Zhuowen Tu. 3d volumetric modeling with introspective neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33(01), pages 8481–8488, 2019.

[6] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3859–3868, 2019.

[7] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[9] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[10] Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proc. CVPR*, 2021.

[12] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021.

[13] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. *arXiv preprint arXiv:2112.11427*, 2021.

[14] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *International Conference on Learning Representations*, 2022.

[15] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. CIPS-3D: A 3D-Aware Generator of GANs Based on Conditionally-Independent Pixel Synthesis. *arXiv preprint arXiv:2110.09788*, 2021.

[16] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurelien Lucchi. Learning generative models of textured 3d meshes from real-world images. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[17] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image {gan}s meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *International Conference on Learning Representations*, 2021.

[18] Moritz Ibing, Isaak Lim, and Leif P. Kobbelt. 3d shape generation with grid-based implicit functions. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[19] Ruihui Li, Xianzhi Li, Ke-Hei Hui, and Chi-Wing Fu. SP-GAN:sphere-guided 3d shape generation and manipulation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 40(4), 2021.

[20] A. Luo, T. Li, W. Zhang, and T. Lee. Surfgen: Adversarial 3d shape synthesis with explicit surface discriminators. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[21] Zhiqin Chen, Vladimir G. Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decor-gan: 3d shape detailization by conditional refinement. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[22] Cheng Wen, Baosheng Yu, and Dacheng Tao. Learning progressive point embeddings for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10266–10275, 2021.

[23] Zekun Hao, Arun Mallya, Serge Belongie, and Ming-Yu Liu. GANcraft: Unsupervised 3D Neural Rendering of Minecraft Worlds. In *ICCV*, 2021.

[24] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. *CVPR*, 2018.

[25] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Variational autoencoders for deforming 3d mesh models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5841–5850, 2018.

[26] Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas J Guibas. Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575*, 2019.

[27] Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao(Richard) Zhang. SDM-NET: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2019)*, 38(6):243:1–243:15, 2019.

[28] Lin Gao, Tong Wu, Yu-Jie Yuan, Ming-Xian Lin, Yu-Kun Lai, and Hao Zhang. Tm-net: Deep generative networks for textured meshes. *ACM Transactions on Graphics (TOG)*, 40(6):263:1–263:15, 2021.

[29] Jinwoo Kim, Jaehoon Yoo, Juho Lee, and Seunghoon Hong. Setvae: Learning hierarchical composition for generative modeling of set-structured data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15059–15068, June 2021.

[30] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. AutoSDF: Shape priors for 3d completion, reconstruction and generation. In *CVPR*, 2022.

[31] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. PointFlow: 3D point cloud generation with continuous normalizing flows. In *ICCV*, 2019.

[32] Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. SoftFlow: Probabilistic framework for normalizing flow on manifolds. In *NeurIPS*, 2020.

[33] Roman Klokov, Edmond Boyer, and Jakob Verbeek. Discrete point flow networks for efficient point cloud generation. In *ECCV*, 2020.

[34] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, and Marco Fumero. Clip-forge: Towards zero-shot text-to-shape generation. *arXiv preprint arXiv:2110.02624*, 2021.

[35] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua E Siegel, and Sanjay E Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *Winter Conference on Applications of Computer Vision*, 2020.

[36] Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. Polygen: An autoregressive generative model of 3d meshes. *ICML*, 2020.

[37] Wei-Jan Ko, Hui-Yu Huang, Yu-Liang Kuo, Chen-Yi Chiu, Li-Heng Wang, and Wei-Chen Chiu. Rpg: Learning recursive point cloud generation. *arXiv preprint arXiv:2105.14322*, 2021.

[38] Moritz Ibing, Gregor Kobsik, and Leif Kobbelt. Octree transformer: Autoregressive 3d shape generation on hierarchically structured sequences. *arXiv preprint arXiv:2111.12480*, 2021.

[39] Jianwen Xie, Zilong Zheng, Ruiqi Gao, Wenguan Wang, Zhu Song-Chun, and Ying Nian Wu. Learning descriptor networks for 3d shape synthesis and analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[40] Jianwen Xie, Yifei Xu, Zilong Zheng, Ruiqi Gao, Wenguan Wang, Zhu Song-Chun, and Ying Nian Wu. Generative pointnet: Deep energy-based learning on unordered point sets for 3d generation, reconstruction and classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[41] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[42] Kangxue Yin, Jun Gao, Maria Shugrina, Sameh Khamis, and Sanja Fidler. 3dstylenet: Creating 3d shapes with geometric and texture style variations. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2021.

[43] Dongsu Zhang, Changwoon Choi, Jeonghwan Kim, and Young Min Kim. Learning to generate 3d shapes with generative cellular automata. In *International Conference on Learning Representations*, 2021.

[44] Chen Chao, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Unsupervised learning of fine structure generation for 3d point clouds by 2d projection matching. *arXiv preprint arXiv:2108.03746*, 2021.

[45] 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)*, 2020.

[46] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[47] 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)*, 2021.

[48] Yu Deng, Jiaolong Yang, and Xin Tong. Deformed implicit field: Modeling 3d shapes with learned dense correspondence. In *IEEE Computer Vision and Pattern Recognition*, 2021.

[49] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. *arXiv preprint arXiv:2112.03221*, 2021.

[50] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[51] Nasir Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Text to mesh without 3d supervision using limit subdivision. *arXiv preprint arXiv:2203.13333*, 2022.

[52] Le Hui, Rui Xu, Jin Xie, Jianjun Qian, and Jian Yang. Progressive point cloud deconvolution generation network. In *ECCV*, 2020.

[53] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

[54] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *arXiv preprint arXiv:2106.15282*, 2021.

[55] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 2021.

[56] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.

[57] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2021.

[58] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *Advances in Neural Information Processing Systems*, 2021.

[59] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[60] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[61] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. In *International Conference on Learning Representations (ICLR)*, 2022.

[62] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion GANs. In *International Conference on Learning Representations (ICLR)*, 2022.

[63] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[64] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

[65] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015.

[66] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, 2019.

[67] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[68] Songyou Peng, Chiyu "Max" Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[69] Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In *Advances in Neural Information Processing Systems*, 2021.

[70] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *The International Conference on Learning Representations*, 2014.

[71] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

[72] Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223, 2018.

[73] Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, and Satoshi Yagi. Variational autoencoder with implicit optimal priors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5066–5073, Jul. 2019.

[74] Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 66–75. PMLR, 16–18 Apr 2019.

[75] Arash Vahdat and Jan Kautz. NVAE: A deep hierarchical variational autoencoder. In *Advances in Neural Information Processing Systems*, 2020.

[76] Jyoti Aneja, Alexander Schwing, Jan Kautz, and Arash Vahdat. NCP-VAE: Variational autoencoders with noise contrastive priors. In *Advances in Neural Information Processing Systems*, 2021.

[77] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. D2c: Diffusion-denoising models for few-shot conditional generation. In *Advances in Neural Information Processing Systems*, 2021.

[78] Mihaela Rosca, Balaji Lakshminarayanan, and Shakir Mohamed. Distribution matching in variational inference. *arXiv preprint arXiv:1802.06847*, 2018.

[79] Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NeurIPS*, 2016.

[80] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*, 2019.

[81] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[82] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, 2017.

[83] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[84] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[85] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

[86] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations (ICLR)*, 2021.

[87] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. DiffWave: A Versatile Diffusion Model for Audio Synthesis. In *International Conference on Learning Representations*, 2021.

[88] Myeonghun Jeong, Hyeongju Kim, Sung Jun Cheon, Byoung Jin Choi, and Nam Soo Kim. Diff-tts: A denoising diffusion model for text-to-speech. *arXiv preprint arXiv:2104.01409*, 2021.

[89] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, Najim Dehak, and William Chan. Wavegrad 2: Iterative refinement for text-to-speech synthesis. *arXiv preprint arXiv:2106.09660*, 2021.

[90] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, 2021.

[91] Songxiang Liu, Dan Su, and Dong Yu. Diffgan-tts: High-fidelity and efficient text-to-speech with denoising diffusion gans. *arXiv preprint arXiv:2201.11972*, 2022.

[92] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, 2021.

[93] Kushagra Pandey, Avideep Mukherjee, Piyush Rai, and Abhishek Kumar. Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents. *arXiv preprint arXiv:2201.00308*, 2022.

[94] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, 2015.

[95] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *International Conference on Learning Representations ICLR*, 2017.

[96] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018.

[97] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2019.

[98] Zhaoyang Lyu, Zhifeng Kong, Xudong XU, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. In *International Conference on Learning Representations (ICLR)*, 2022.

[99] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[100] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[101] Songyou Peng, Michael Niemeyer, Lars Mescheder, and Andreas Geiger Marc Pollefeys. Convolutional occupancy networks. In *European Conference on Computer Vision (ECCV)*, 2020.

[102] Francis Williams, Matthew Trager, Joan Bruna, and Denis Zorin. Neural splines: Fitting 3d surfaces with infinitely-wide neural networks. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[103] Francis Williams, Zan Gojcic, Sameh Khamis, Denis Zorin, Joan Bruna, Sanja Fidler, and Or Litany. Neural fields as learnable kernels for 3d reconstruction. *arXiv preprint arXiv:2111.13674*, 2021.

[104] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing 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. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[105] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning, ICML*, 2021.

[106] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[107] Wenzheng Chen, Huan Ling, Jun Gao, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. In *Advances in Neural Information Processing Systems*, 2019.

[108] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 2020.

[109] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[110] Wenzheng Chen, Joey Litalien, Jun Gao, Zian Wang, Clement Fuji Tsang, Sameh Khamis, Or Litany, and Sanja Fidler. DIB-R++: Learning to predict lighting and material with a hybrid differentiable renderer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[111] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Niessner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhoefer, and Vladislav Golyanik. Advances in neural rendering. *arXiv preprint arXiv:2111.05849*, 2021.

[112] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proceedings IEEE International Conf. on Computer Vision (ICCV)*, 2019.

[113] Zhiqin Chen, Kangxue Yin, and Sanja Fidler. Auv-net: Learning aligned uv maps for texture transfer and synthesis. In *The Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[114] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Texturify: Generating textures on 3d shape surfaces. *arXiv preprint arXiv:2204.02411*, 2022.

[115] Daniel Watson, Jonathan Ho, Mohammad Norouzi, and William Chan. Learning to efficiently sample from diffusion probabilistic models. *arXiv preprint arXiv:2106.03802*, 2021.

[116] Zhifeng Kong and Wei Ping. On fast sampling of diffusion probabilistic models. *arXiv preprint arXiv:2106.00132*, 2021.

[117] Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta Go Fast When Generating Data with Score-Based Models. *arXiv preprint arXiv:2105.14080*, 2021.

[118] Daniel Watson, William Chan, Jonathan Ho, and Mohammad Norouzi. Learning Fast Samplers for Diffusion Models by Differentiating Through Sample Quality. In *International Conference on Learning Representations*, 2022.

[119] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations (ICLR)*, 2022.

[120] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022.

[121] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-DPM: an Analytic Estimate of the Optimal Reverse Variance in Diffusion Probabilistic Models. In *International Conference on Learning Representations*, 2022.

[122] Cristian Vaccari and Andrew Chadwick. Deepfakes and disinformation: Exploring the impact of synthetic political video on deception, uncertainty, and trust in news. *Social Media+ Society*, 6(1): 2056305120903408, 2020.

[123] Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Cuong M. Nguyen, Dung Nguyen, Duc Thanh Nguyen, and Saeid Nahavandi. Deep learning for deepfakes creation and detection: A survey. *arXiv preprint arXiv:1909.11573*, 2021.

[124] Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *ACM Comput. Surv.*, 54(1), 2021.

[125] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: a retargetable forward and inverse renderer. *ACM Transactions on Graphics*, 38:1–17, 11 2019. doi: 10.1145/3355089.3356498.

[126] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[127] Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016.

[128] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[129] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[130] Ulrich G Haussmann and Etienne Pardoux. Time reversal of diffusions. *The Annals of Probability*, pages 1188–1205, 1986.

[131] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

[132] Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.

[133] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019.

[134] J. R. Dormand and P. J. Prince. A family of embedded runge–kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980.

[135] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

[136] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery.

[137] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[138] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[139] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[140] 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)*, 2019.

[141] 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*, pages 16259–16268, 2021.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Sec. 6.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Please see Sec. 6.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] We did not derive novel theoretical results. We rather propose a novel generative model of 3D shapes.
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We will release code and instructions to reproduce all experiments upon acceptance of the manuscript. The internal guidelines of our institution prevent us from releasing code at this stage.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We provide all model training and evaluation details in the App. D, including all hyperparameters.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Following standard conventions in the related 3D generative modeling literature, we do not report error bars. Furthermore, we avoid running similar setups repeatedly to save computational resources (our main models are quite large and require substantial GPU resources for training, see App. E.9).
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please see App. E.9.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] For baseline comparisons we run publicly available code from previous, publicly available papers, which we cite. We also use various datasets. Here, we present a summary.

      To compare to baselines, we use the following codes:

      - r-GAN, l-GAN [2]: `https://github.com/optas/latent_3d_points` (MIT License)
      - PointFlow [31]: `https://github.com/stevenygd/PointFlow` (MIT License)
      - SoftFlow [32]: `https://github.com/ANLGBOY/SoftFlow`
      - Set-VAE [29]: `https://github.com/jw9730/setvae` (MIT License)
      - DPF-NET [33]: `https://github.com/Regenerator/dpf-nets`
      - DPM [47]: `https://github.com/luost26/diffusion-point-cloud` (MIT License)
      - PVD [46]: `https://github.com/alexzhou907/PVD` (MIT License)
      - ShapeGF [45]: `https://github.com/RuojinCai/ShapeGF` (MIT License)
      - SP-GAN [19]: `https://github.com/liruihui/sp-gan` (MIT License)
      - PDGN [52]: `https://github.com/fpthink/PDGN` (MIT License)
      - IM-GAN [7]: `https://github.com/czq142857/implicit-decoder` (MIT license) and `https://github.com/czq142857/IM-NET-pytorch` (MIT license)
      - GCA [43]: `https://github.com/96lives/gca` (MIT license)

      We use further codebases in other places:

      - We use the MitSuba renderer for visualizations [125]: `https://github.com/mitsuba-renderer/mitsuba2` (License: `https://github.com/mitsuba-renderer/mitsuba2/blob/master/LICENSE`), and the code to generate the scene discription files for MitSuba [31]: https://github.com/zekunhao1995/PointFlowRenderer.

- We rely on SAP [68] for mesh generation with the code at https://github.com/autonomousvision/shape_as_points (MIT License).
- For calculating the evaluation metrics, we use the implementation for CD at https://github.com/ThibaultGROUEIX/ChamferDistancePytorch (MIT License) and for EMD at https://github.com/daerduoCarey/PyTorchEMD.
- We use Text2Mesh [49] for per-sample text-driven texture synthesis: https://github.com/threedle/text2mesh (MIT License)

We also rely on the following datasets:

- ShapeNet [104]. Its terms of use can be found at https://shapenet.org/terms.
- The Cars dataset [126] from http://ai.stanford.edu/~jkrause/cars/car_dataset.html with ImageNet License: https://image-net.org/download.php.
- The TurboSquid data repository, https://www.turbosquid.com. We obtained a custom license from TurboSquid.
- Redwood 3DScan Dataset [127]: https://github.com/isl-org/redwood-3dscan (Public Domain)
- Pix3D [128]: https://github.com/xingyuansun/pix3d. (Creative Commons Attribution 4.0 International License).

(b) Did you mention the license of the assets? [Yes] In App. E.8, we mention the licenses of the codes and other assets we are using.

(c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We are primarily using the publicly available ShapeNet [104] dataset, which has been widely used in the generative modeling literature as standard benchmark. It only consists of simple 3D models of shapes such as airplanes, chairs, cars, etc.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]