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

EECS 598-007 Paper presentation

Presenter: Jiayao Yang

10/10/23

Contents

Background: problem

how can we use the generative models in some applications ?

Background: problem

how can we use the generative models in some applications ?

digital artists

Background: problem

how can we use the generative models in some applications ?

digital artists

$$\text{3D points cloud} \xrightarrow{generate} \text{3D shape}$$

Background: more technical

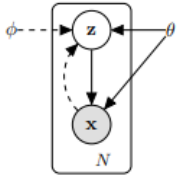
Background: more technical

how can we learn data distribution ?

$$p_\theta(\mathbf{x}) \quad \rightarrow \text{likelihood}$$

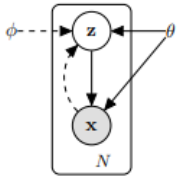
Background 1: VAE

Variational auto-encoder (VAE)



Background 1: VAE

Variational auto-encoder (VAE)



$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \quad \rightarrow \text{intractable}$$

Background 1: VAE

Variational auto-encoder (VAE)[1]

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \quad \rightarrow \text{intractable}$$

introduce model $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate posterior $p_\theta(\mathbf{z}|\mathbf{x})$
consider the log likelihood of the data,

[1] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes.", 2013

Background 1: VAE

Variational auto-encoder (VAE)[1]

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \quad \rightarrow \text{intractable}$$

introduce model $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate posterior $p_\theta(\mathbf{z}|\mathbf{x})$
consider the log likelihood of the data,

$$\log p_\theta(\mathbf{x})$$

[1] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes.", 2013

Background 1: VAE

Variational auto-encoder (VAE)[1]

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \quad \rightarrow \text{intractable}$$

introduce model $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate posterior $p_\theta(\mathbf{z}|\mathbf{x})$
consider the log likelihood of the data,

$$\log p_\theta(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})}]$$

[1] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes.", 2013

Background 1: VAE

Variational auto-encoder (VAE)[1]

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \quad \rightarrow \text{intractable}$$

introduce model $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate posterior $p_\theta(\mathbf{z}|\mathbf{x})$
consider the log likelihood of the data,

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log \frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})}] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log q_\phi(\mathbf{z}|\mathbf{x}) - \log p_\theta(\mathbf{z}|\mathbf{x}) - \log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x},\mathbf{z})] \\
&= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x},\mathbf{z})] \\
&= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}|\mathbf{z}) + \log p_\theta(\mathbf{z})]
\end{aligned}$$

[1] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes.", 2013

Background 1: VAE

Variational auto-encoder (VAE)[1]

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) d\mathbf{z} \quad \rightarrow \text{intractable}$$

introduce model $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate posterior $p_\theta(\mathbf{z}|\mathbf{x})$
consider the log likelihood of the data,

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log \frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x})}] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log q_\phi(\mathbf{z}|\mathbf{x}) - \log p_\theta(\mathbf{z}|\mathbf{x}) - \log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x},\mathbf{z})] \\
&= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x},\mathbf{z})] \\
&= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x}|\mathbf{z}) + \log p_\theta(\mathbf{z})] \\
&= D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) \underbrace{-D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{variational lower bound}=\mathcal{L}}
\end{aligned}$$

[1] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes.", 2013

Background 1: VAE

Variational auto-encoder (VAE)
data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ consisting i.i.d. samples, for each data sample $\mathbf{x}$, the log likelihood is

$$\log p_\theta(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathcal{L}$$

Background 1: VAE

Variational auto-encoder (VAE)
data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ consisting i.i.d. samples, for each data sample $\mathbf{x}$, the log likelihood is

$$\log p_\theta(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathcal{L}$$

optimize the variational bound

$$\mathcal{L} = -D_{KL}(\underbrace{q_\phi(\mathbf{z}|\mathbf{x})}_{\textit{encoder}} || \underbrace{p_\theta(\mathbf{z})}_{\textit{prior}}) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[\log \underbrace{p_\theta(\mathbf{x}|\mathbf{z})}_{\textit{decoder}}]$$

Background 1: VAE

Variational auto-encoder (VAE)
data $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ consisting i.i.d. samples, for each data sample $\mathbf{x}$, the log likelihood is

$$\log p_\theta(\mathbf{x}) = D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathcal{L}$$

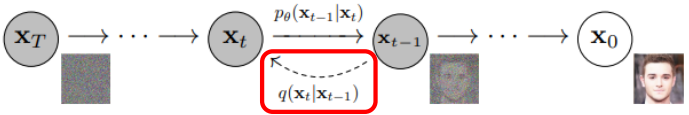
optimize the variational bound

$$\mathcal{L} = -D_{KL}(\underbrace{q_\phi(\mathbf{z}|\mathbf{x})}_{\textit{encoder}} || \underbrace{p_\theta(\mathbf{z})}_{\textit{prior}}) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[\log \underbrace{p_\theta(\mathbf{x}|\mathbf{z})}_{\textit{decoder}}]$$

for example, assume follows Gaussian distribution, and select

$$p_\theta(\mathbf{z}) \sim \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$$

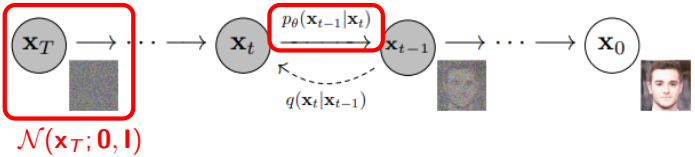
Background 2: Diffusion model



[2]Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.", 2020

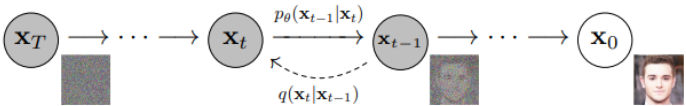
Background 2: Diffusion model

reverse process



[2]Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.", 2020

Background 2: Diffusion model

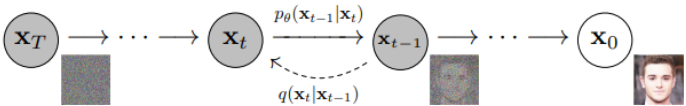


optimize the variational bound on the negative log likelihood[2]

$$\mathbb{E}[-\log p(\mathbf{x})] \leq L = \cdots$$
$$= \sum_t \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2 \right] + C$$

[2]Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.", 2020

Background 2: Diffusion model



optimize the variational bound on the negative log likelihood[2]

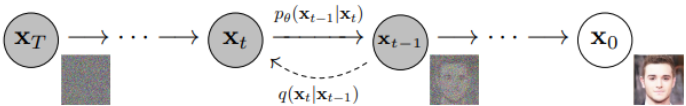
$$\mathbb{E}[-\log p(\mathbf{x})] \leq L = \cdots$$

$$= \sum_t \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2 \right] + C$$

$$= \sum_t \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \underbrace{\boldsymbol{\epsilon}_\theta}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right] + C$$

[2] Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.", 2020

Background 2: Diffusion model



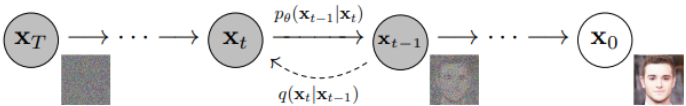
optimize the variational bound on the negative log likelihood[2]

$$\begin{aligned}\mathbb{E}[-\log p(\mathbf{x})] &\leq L = \cdots \\ &= \sum_t \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2 \right] + C \\ &= \sum_t \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \underbrace{\boldsymbol{\epsilon}_\theta}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right] + C\end{aligned}$$

simple surrogate objective: $L_{simple}(\theta) = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right]$

[2] Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.", 2020

Background 2: Diffusion model



optimize the variational bound on the negative log likelihood[2]

$$\begin{aligned}\mathbb{E}[-\log p(\mathbf{x})] &\leq L = \cdots \\ &= \sum_t \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2 \right] + C \\ &= \sum_t \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}} \left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \underbrace{\boldsymbol{\epsilon}_\theta}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right] + C\end{aligned}$$

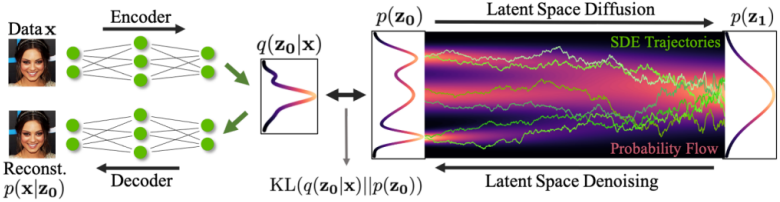
simple surrogate objective: $L_{simple}(\theta) = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right]$

- can sample/generate high quality data

[2] Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models.", 2020

Background 3: Diffusion model in latent space

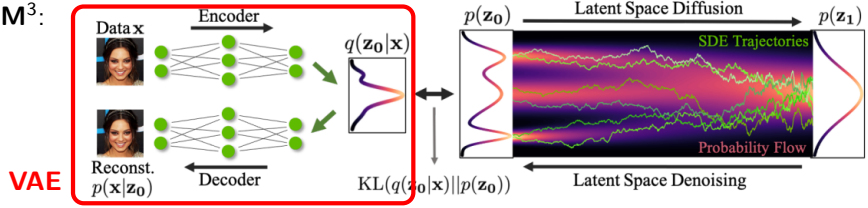
LSGM[3]:



[3]Vahdat, Arash, Karsten Kreis, and Jan Kautz. "Score-based generative modeling in latent space.", 2021

Background 3: Diffusion model in latent space

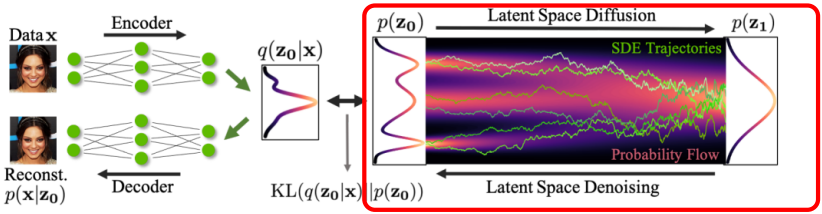
LSGM[3]:



[3]Vahdat, Arash, Karsten Kreis, and Jan Kautz. "Score-based generative modeling in latent space.", 2021

Background 3: Diffusion model in latent space

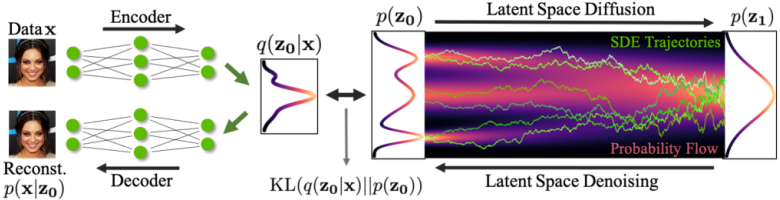
LSGM[3]:



[3]Vahdat, Arash, Karsten Kreis, and Jan Kautz. "Score-based generative modeling in latent space.", 2021

Background 3: Diffusion model in latent space

LSGM[3]:

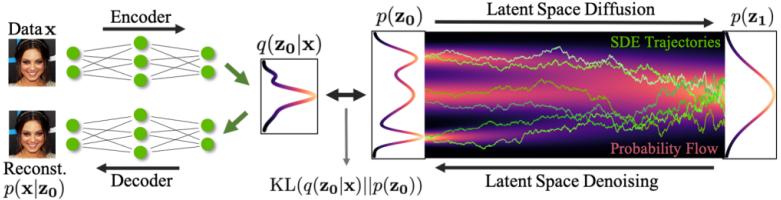


- Advantages
  - synthesis speed;
  - expressivity;
  - tailored encoders and decoders;

[3]Vahdat, Arash, Karsten Kreis, and Jan Kautz. "Score-based generative modeling in latent space.", 2021

Background 3: Diffusion model in latent space

LSGM[3]:



- Advantages
  - synthesis speed;
  - expressivity;
  - tailored encoders and decoders;
- Mixed score parameterization
  - "a novel parameterization of the score function that allows SGM to focus on the mismatch of the target distribution with respect to a simple Normal one"

    example in 1-dimensional:

    $\nabla_{z_t} \log p(z_t) = -(1-\alpha)z_t + \nabla_{z_t} \log p'_\theta(z_t)$ if think a mixture of $p(z_t) \propto \mathcal{N}(z_t; 0, 1)^{1-\alpha} p'_\theta(z_t)^\alpha$

[3] Vahdat, Arash, Karsten Kreis, and Jan Kautz. "Score-based generative modeling in latent space.", 2021

LION: Latent Point Diffusion Models for 3D Shape Generation

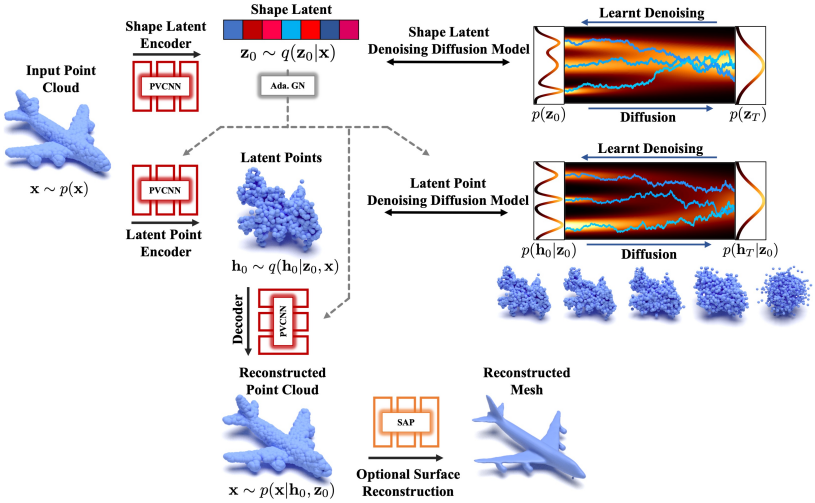
Xiaohui Zeng[1,2,3,*] **Arash Vahdat**[1] **Francis Williams**[1]

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

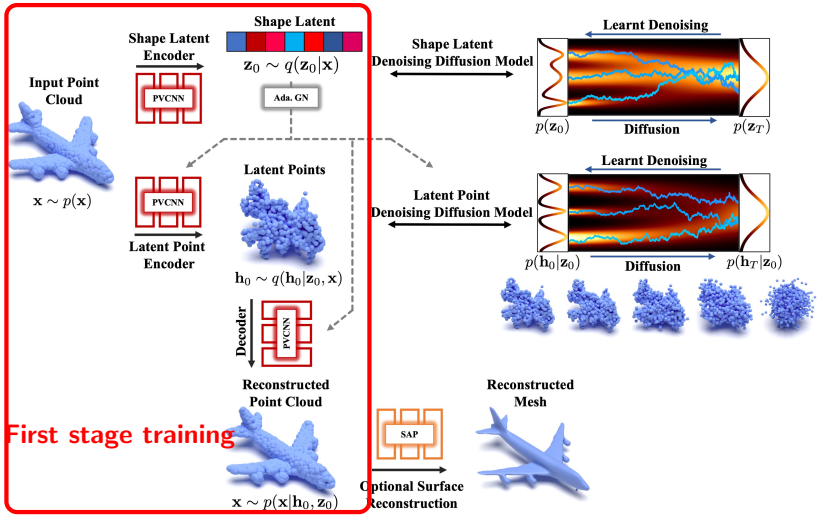
LION


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
$p(\mathbf{z}_0)$
Diffusion
$p(\mathbf{z}_T)$
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
$p(\mathbf{h}_0|\mathbf{z}_0)$
Diffusion
$p(\mathbf{h}_T|\mathbf{z}_0)$
Decoder
PVCNN
Reconstructed Point Cloud
$\mathbf{x} \sim p(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0)$
SAP
Optional Surface Reconstruction
Reconstructed Mesh

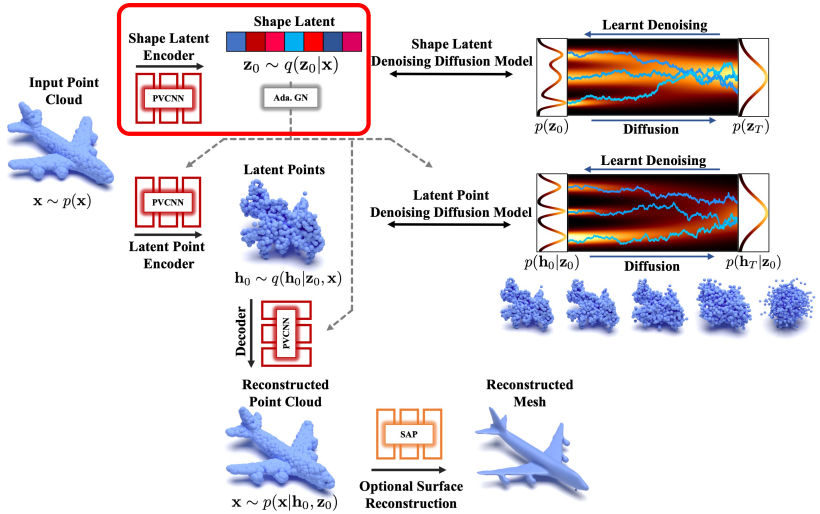
LION

- First stage training
- Second stage training
- Surface reconstruction (optional)

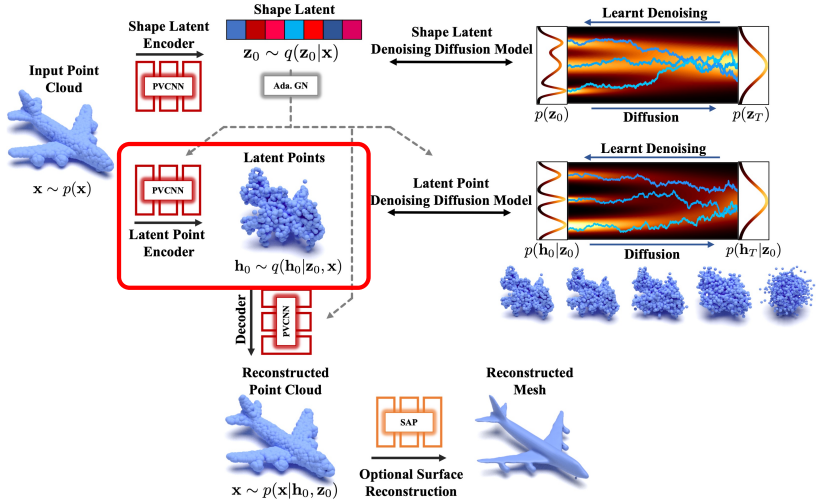
LION: First Stage Training



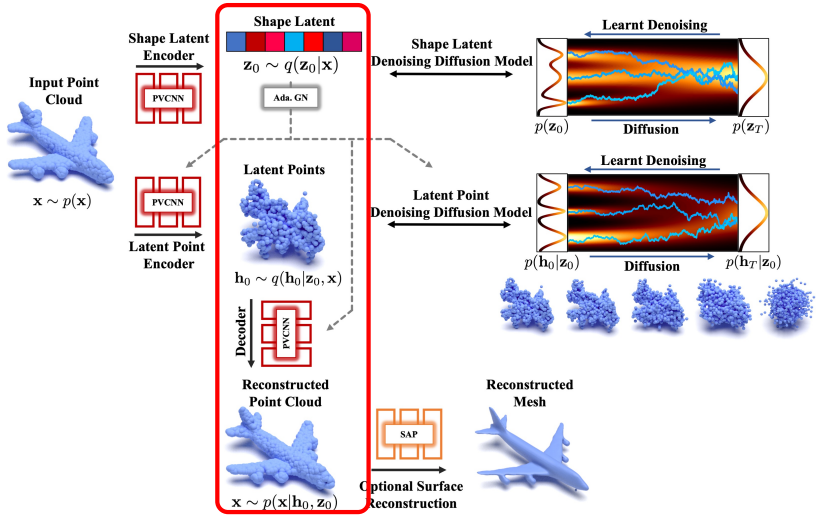
LION: First Stage Training



LION: First Stage Training



LION: First Stage Training



LION: First Stage Training

First Stage Training: follow the VAE framework

LION: First Stage Training

First Stage Training: follow the VAE framework

- 3D point cloud $\mathbf{x} \in \mathbb{R}^{3 \times N}$
- global shape latent $\mathbf{z}_0 \in \mathbb{R}^{D_{\mathbf{z}}}$ and point cloud-structured latent $\mathbf{h}_0 \in \mathbb{R}^{(3+D_{\mathbf{h}}) \times N}$

LION: First Stage Training

First Stage Training: follow the VAE framework

- 3D point cloud $\mathbf{x} \in \mathbb{R}^{3\times N}$
- global shape latent $\mathbf{z}_0 \in \mathbb{R}^{D_\mathbf{z}}$ and point cloud-structured latent $\mathbf{h}_0 \in \mathbb{R}^{(3+D_\mathbf{h})\times N}$
- optimize modified variational lower bound

$$\mathcal{L}_{ELBO}(\phi,\xi) = \mathbb{E}_{p(\mathbf{x}),q_\phi(\mathbf{z}_0|\mathbf{x}),q_\phi(\mathbf{h}_0|\mathbf{x},\mathbf{z}_0)}\Big[\underbrace{\log p_\xi(\mathbf{x}|\mathbf{h}_0,\mathbf{z}_0)}_{decoder} - \lambda_\mathbf{z} D_{KL}(\underbrace{q_\phi(\mathbf{z}_0|\mathbf{x})}_{encoder\ 1} || \underbrace{p(\mathbf{z}_0)}_{prior}) - \lambda_\mathbf{h} D_{KL}(\underbrace{q_\phi(\mathbf{h}_0|\mathbf{x},\mathbf{z}_0)}_{encoder\ 2} || \underbrace{p(\mathbf{h}_0)}_{prior})\Big]$$

$\lambda_\mathbf{z}, \lambda_\mathbf{h}$: hyperparameters

LION: First Stage Training

First Stage Training: follow the VAE framework

- 3D point cloud $\mathbf{x} \in \mathbb{R}^{3\times N}$
- global shape latent $\mathbf{z}_0 \in \mathbb{R}^{D_\mathbf{z}}$ and point cloud-structured latent $\mathbf{h}_0 \in \mathbb{R}^{(3+D_\mathbf{h})\times N}$
- optimize modified variational lower bound

$$\mathcal{L}_{ELBO}(\phi,\xi) = \mathbb{E}_{p(\mathbf{x}),q_\phi(\mathbf{z}_0|\mathbf{x}),q_\phi(\mathbf{h}_0|\mathbf{x},\mathbf{z}_0)}\Big[\log \underbrace{p_\xi(\mathbf{x}|\mathbf{h}_0,\mathbf{z}_0)}_{decoder} - \lambda_\mathbf{z} D_{KL}(\underbrace{q_\phi(\mathbf{z}_0|\mathbf{x})}_{encoder\ 1} || \underbrace{p(\mathbf{z}_0)}_{prior}) - \lambda_\mathbf{h} D_{KL}(\underbrace{q_\phi(\mathbf{h}_0|\mathbf{x},\mathbf{z}_0)}_{encoder\ 2} || \underbrace{p(\mathbf{h}_0)}_{prior})\Big]$$

$\lambda_\mathbf{z}, \lambda_\mathbf{h}$: hyperparameters

- select priors

$$p(\mathbf{z}_0) \sim \mathcal{N}(\mathbf{0},\mathbf{I}), \quad p(\mathbf{h}_0) \quad \sim \mathcal{N}(\mathbf{0},\mathbf{I})$$

LION: First Stage Training

First Stage Training: follow the VAE framework

- Implementation

Input: point clouds (2048×3)	
Output: shape latent (128×1)	
	SA 1 | SA 2
# PVC layers	2 | 1
# PVC hidden dimension	32 | 32
# PVC voxel grid size	32 | 16
# Grouper center	1024 | 256
# Grouper radius	0.1 | 0.2
# Grouper neighbors	32 | 32
# MLP layers	2 | 2
# MLP output dimension	32,32 | 32,64
Use attention	False | True
# Attention dimension	- | 128
Linear: (64, 128)	

Table 5: Shape Latent Encoder Architecture Hyperparameters.

Input: point clouds (2048×3), shape latent (1×128)				
Output: latent points ($2048 \times 2 \times (3 + D_h)$)				
	SA 1	SA 2	SA 3	SA 4
# PVC layers	2	1	1	-
# PVC hidden dimension	32	64	128	-
# PVC voxel grid size	32	16	8	-
# Grouper center	1024	256	64	16
# Grouper radius	0.1	0.2	0.4	0.8
# Grouper neighbors	32	32	32	32
# MLP layers	2	2	2	3
# MLP output dimension	32,32	64,128	128,256	128,128,128
Use attention	False	True	False	False
# Attention dimension	-	128	-	-
Global attention layer, hidden dimension: 256				
	FP 1	FP 2	FP 3	FP 4
# MLP layers	2	2	2	3
# MLP output dimension	128,128	128,128	128,128	128,128,64
# PVC layers	3	3	2	2
# PVC hidden dimension	128	128	128	64
# PVC voxel grid size	8	8	16	32
Use attention	False	True	False	False
# Attention dimension	-	128	-	-
MLP: (64, 128)				
Dropout				
Linear: (128, $2 \times (3 + D_h)$)				

Table 6: Latent Point Encoder Architecture Hyperparameters.

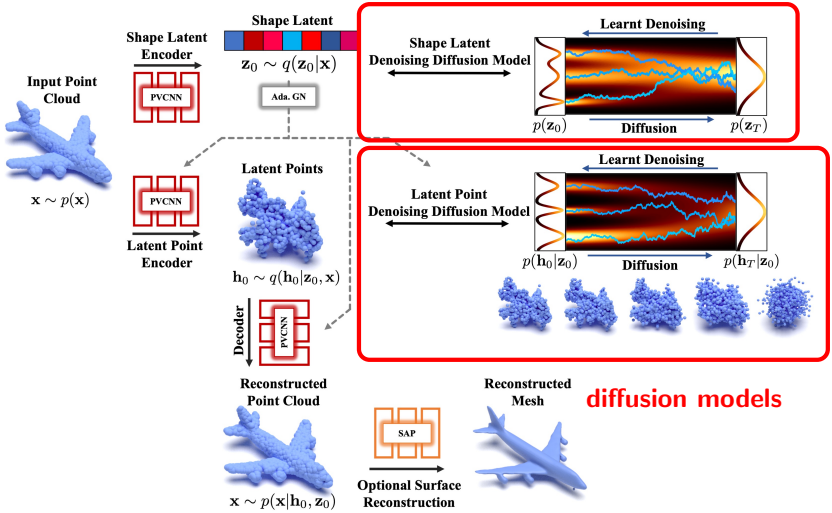
Input feature size: latent points ($2048 \times (3 + D_h)$), shape latent (1×128)				
Output feature size: point clouds (2048×3)				
	SA 1	SA 2	SA 3	SA 4
# PVC layers	2	1	1	-
# PVC hidden dimension	32	64	128	-
# PVC voxel grid size	32	16	8	-
# Grouper center	1024	256	64	16
# Grouper radius	0.1	0.2	0.4	0.8
# Grouper neighbors	32	32	32	32
# MLP layers	2	2	2	3
# MLP output dimension	32,64	64,128	128,256	128,128,128
Use attention	False	True	False	False
# Attention dimension	-	128	-	-
Global attention layer, hidden dimension: 256				
	FP 1	FP 2	FP 3	FP 4
# MLP layers	2	2	2	3
# MLP output dimension	128,128	128,128	128,128	128,128,64
# PVC layers	3	3	2	2
# PVC hidden dimension	128	128	128	64
# PVC voxel grid size	8	8	16	32
Use attention	False	False	False	False
MLP: (64, 128)				
Dropout				
Linear: (128, 3)				

Table 7: Decoder Architecture Hyperparameters.

network structures based on PVCNN[4]

[4] Liu, Zhijian, et al. "Point-voxel cnn for efficient 3d deep learning.", 2019

LION: Second Stage Training



LION: Second Stage Training

Second Stage Training: latent DDMs

LION: Second Stage Training

Second Stage Training: latent DDMs

- fix encoder and decoder,

LION: Second Stage Training

Second Stage Training: latent DDMs

- fix encoder and decoder,
- train diffusion models on encodings $\mathbf{z}_0$ and $\mathbf{h}_0$ sampled from $q_\phi(\mathbf{z}_0|\mathbf{x})$ and $q_\phi(\mathbf{h}_0|\mathbf{x}, \mathbf{z}_0)$

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

LION: Second Stage Training

Second Stage Training: latent DDMs

- fix encoder and decoder,
- train diffusion models on encodings $\mathbf{z}_0$ and $\mathbf{h}_0$ sampled from $q_\phi(\mathbf{z}_0|\mathbf{x})$ and $q_\phi(\mathbf{h}_0|\mathbf{x}, \mathbf{z}_0)$

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

- Implementation
  - $T = 1000$ time steps
  - *mixed score parameterization*, similar to LSGM [Vahdat, Arash, et al., 2021]

LION: Second Stage Training

Second Stage Training: latent DDMs

- Implementation

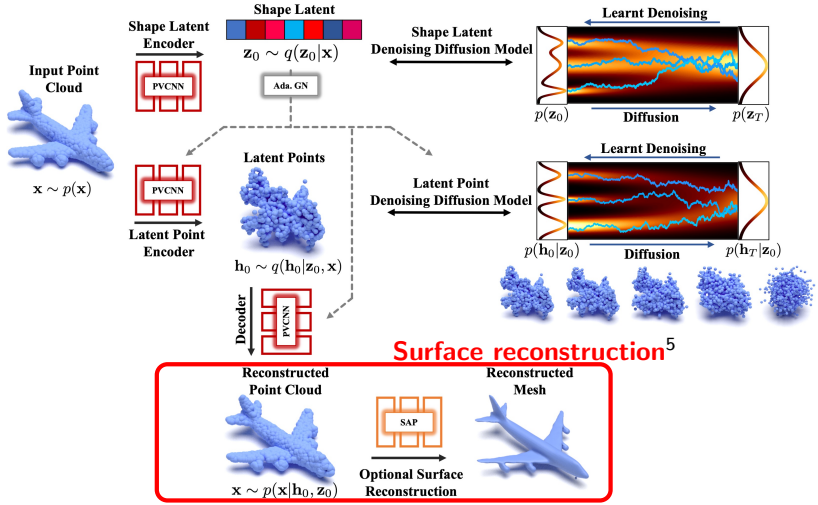
Input: latent points ($2048 \times (3 + D_h)$) at t shape latent (1×128)
Output: $2048 \times (3 + D_h)$
Time embedding layer:
Sinusoidal embedding dimension = 128 Linear (128, 512) LeakyReLU (0.1) Linear (2048)
Linear (128, 2048) Addition (linear output, time embedding) ResSE (2048, 2048) x 8 Linear (2048, 128)

Table 8: Shape Latent DDM Architecture Hyperparameters.

Input: latent points ($2048 \times (3 + D_h)$) at t, shape latent (1×128)				
Output: $2048 \times (3 + D_h)$				
Time embedding Layer:				
Sinusoidal embedding dimension = 64 Linear (64, 64) LeakyReLU(0.1) Linear (64, 64)				
	SA 1	SA 2	SA 3	SA 4
# PVC layers	2	1	1	-
# PVC hidden dimension	32	64	128	-
# PVC voxel grid size	32	16	8	-
# Grouper center	1024	256	64	16
# Grouper radius	0.1	0.2	0.4	0.8
# Grouper neighbors	32	32	32	32
# MLP layers	2	2	2	3
# MLP output dimension	32,64	64,128	128,128	128,128,128
Use attention	False	True	False	False
# Attention dimension	-	128	-	-
Global Attention Layer, hidden dimension: 256				
	FP 1	FP 2	FP 3	FP 4
# MLP layers	2	2	2	3
# MLP output dimension	128,128	128,128	128,128	128,128,64
# PVC layers	3	3	2	2
# PVC hidden dimension	128	128	128	64
# PVC voxel grid size	8	8	16	32
Use attention	False	False	False	False
MLP: (64, 128) Dropout Linear: (128, 3)				

Table 9: Latent Point DDM Architecture Hyperparameters.

LION: Surface reconstruction (optional)



[5]Songyou Peng, et al. "Shape as points: A differentiable poisson solver.", 2021.

Illustration of LION

Hierarchical generative model

$$p_{\xi,\phi,\theta}(\mathbf{x}, \mathbf{h}_0, \mathbf{z}_0) = p_\xi(\mathbf{x}|\mathbf{h}_0, \mathbf{z}_0) p_\phi(\mathbf{h}_0|\mathbf{z}_0) p_\theta(\mathbf{z}_0)$$

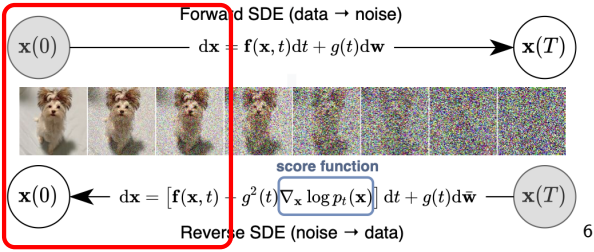
an illustration of reconstructing points
https://research.nvidia.com/labs/toronto-ai/LION/

Extension

Multimodal generation: how to synthesize variations of a given shape?

Extension

Multimodal generation: how to synthesize variations of a given shape?
for example,

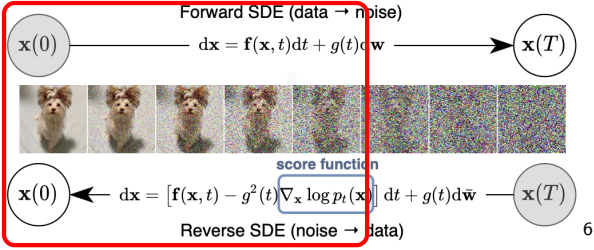


diffuse for $\tau = 3$ steps, then denoise

[6]picture from: Song, Yang, et al., 2020.

Extension

Multimodal generation: how to synthesize variations of a given shape?
for example,

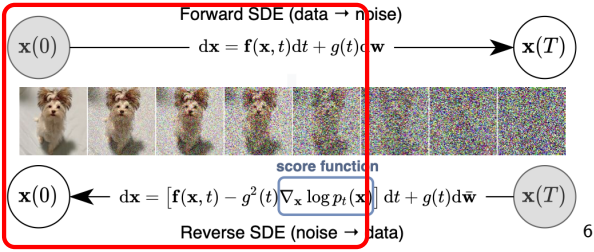


diffuse for $\tau = 5$ steps, then denoise

[6]picture from: Song, Yang, et al., 2020.

Extension

Multimodal generation: how to synthesize variations of a given shape?
for example,



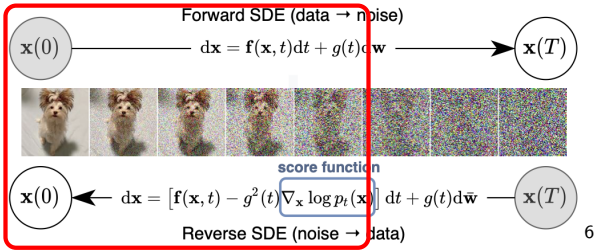
diffuse for $\tau = 5$ steps, then denoise

possibly get a different data

[6]picture from: Song, Yang, et al., 2020.

Extension

Multimodal generation: how to synthesize variations of a given shape?
for example,



[6]

diffuse-denoise procedure

in LION, do this for latent variables $\mathbf{z}_0$ and $\mathbf{h}_0$

example of some τ used in the paper, $\tau = 50, 200, 250$

[6] picture from: Song, Yang, et al., 2020.

Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...

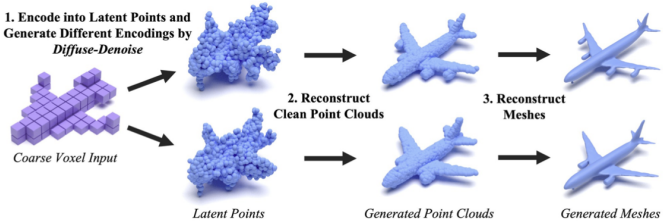
Multimodal generation of airplanes.

Multimodal generation of chairs.

Multimodal generation of cars.

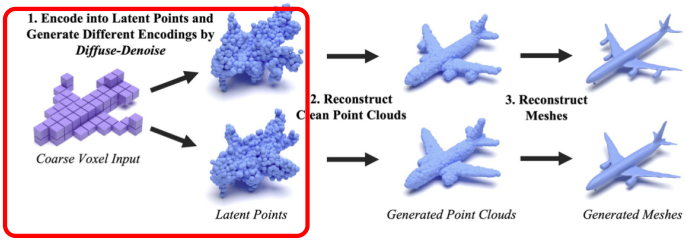
Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...



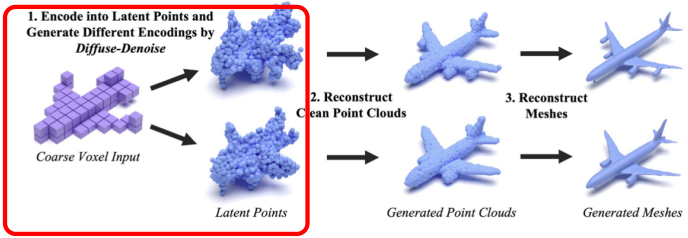
Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...

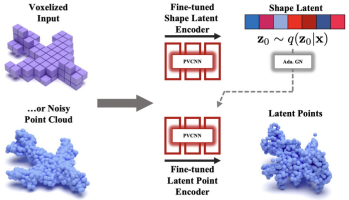


Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...

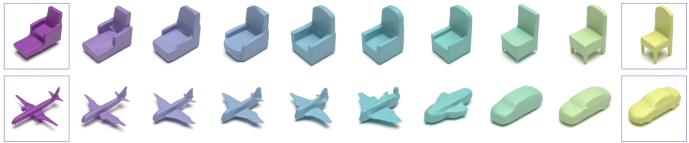


(fine-tune the encoder)



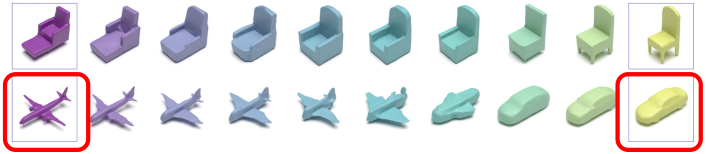
Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...



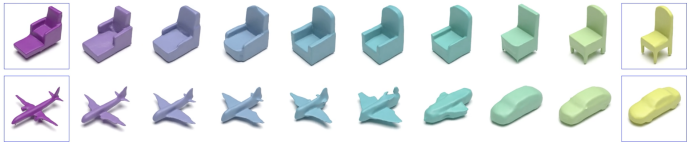
Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...



Applications and extensions

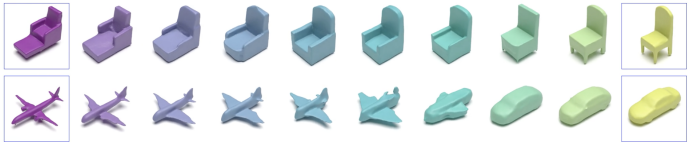
- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...



interpolate in the prior spaces of latent DDMs

Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...



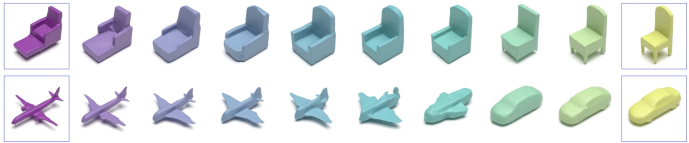
interpolate in the prior spaces of latent DDMs

shape A: $\mathbf{x}_0^A \quad \rightarrow \quad \mathbf{z}_0^A, \mathbf{h}_0^A \quad \rightarrow \quad \mathbf{z}_1^A, \mathbf{h}_1^A,$

shape B: $\mathbf{x}_0^B \quad \rightarrow \quad \mathbf{z}_0^B, \mathbf{h}_0^B \quad \rightarrow \quad \mathbf{z}_1^B, \mathbf{h}_1^B,$

Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...



interpolate in the prior spaces of latent DDMs

shape A: $\mathbf{x}_0^A \rightarrow \mathbf{z}_0^A, \mathbf{h}_0^A \rightarrow \mathbf{z}_1^A, \mathbf{h}_1^A,$

shape B: $\mathbf{x}_0^B \rightarrow \mathbf{z}_0^B, \mathbf{h}_0^B \rightarrow \mathbf{z}_1^B, \mathbf{h}_1^B,$

- **spherical interpolation**

$$\mathbf{z}_1^s = \sqrt{s}\mathbf{z}_1^A + \sqrt{1-s}\mathbf{z}_1^B,$$
$$\mathbf{h}_1^s = \sqrt{s}\mathbf{h}_1^A + \sqrt{1-s}\mathbf{h}_1^B,$$

Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...

incorporate **SAP method** into LION
[Songyou Peng, et al., 2021]

Applications and extensions

- Multimodal generation
- Voxel-conditioned synthesis
- Shape interpolation
- Surface reconstruction
- Text-driven generation
- others ...

Experiments

- Single-class 3D shape generation
  - ShapeNet dataset (trained on category airplane, chair, car)
  - evaluated by 1-NNA with Chanfer distance (CD) and earth mover distance (EMD)
  - LION outperforms all baselines and achieves state-of-the-art performance
- Many-class unconditional 3D shape generation
- Training on small datasets

Experiments

- Single-class 3D shape generation

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**

- Many-class unconditional 3D shape generation
- Training on small datasets

Experiments

- Single-class 3D shape generation
- Many-class unconditional 3D shape generation
  - trained 13-class LION model, and 55-class LION model
  - the generation performance of LION was better than the others
  - a surprise that it can be trained on diverse 3D data without relying on conditioning info

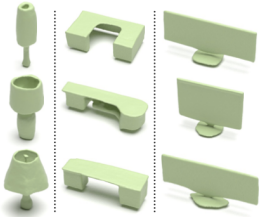


Figure 8: Samples from our unconditional 13-class model: In each column, we use the same global shape latent $\mathbf{z}_0$.

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**

- Training on small datasets

Experiments

- Single-class 3D shape generation
- Many-class unconditional 3D shape generation
- Training on small datasets
  - with 149 Mug data samples / 340 Bottle data samples
  - authors concluded that also performed well

Summary

- advantages: (i) expressivity; (ii) flexibility; (iii) mesh reconstruction;
- can potentially improve 3D content creation and assist the workflow of digital artists;

Summary

- advantages: (i) expressivity; (ii) flexibility; (iii) mesh reconstruction;
- can potentially improve 3D content creation and assist the workflow of digital artists;

- however not able to generate the textures

Quiz

1. in LION, the diffusion models were trained for ____ ?
   A) recover the point cloud $\mathbf{x}$
   B) sample better latent variable $\mathbf{z}_0, \mathbf{h}_0$

Quiz

1. in LION, the diffusion models were trained for ____ ?
   A) recover the point cloud $\mathbf{x}$
   B) sample better latent variable $\mathbf{z}_0, \mathbf{h}_0$
2. given the same input points, what technique was used to generate two possible shapes ?

Q&A