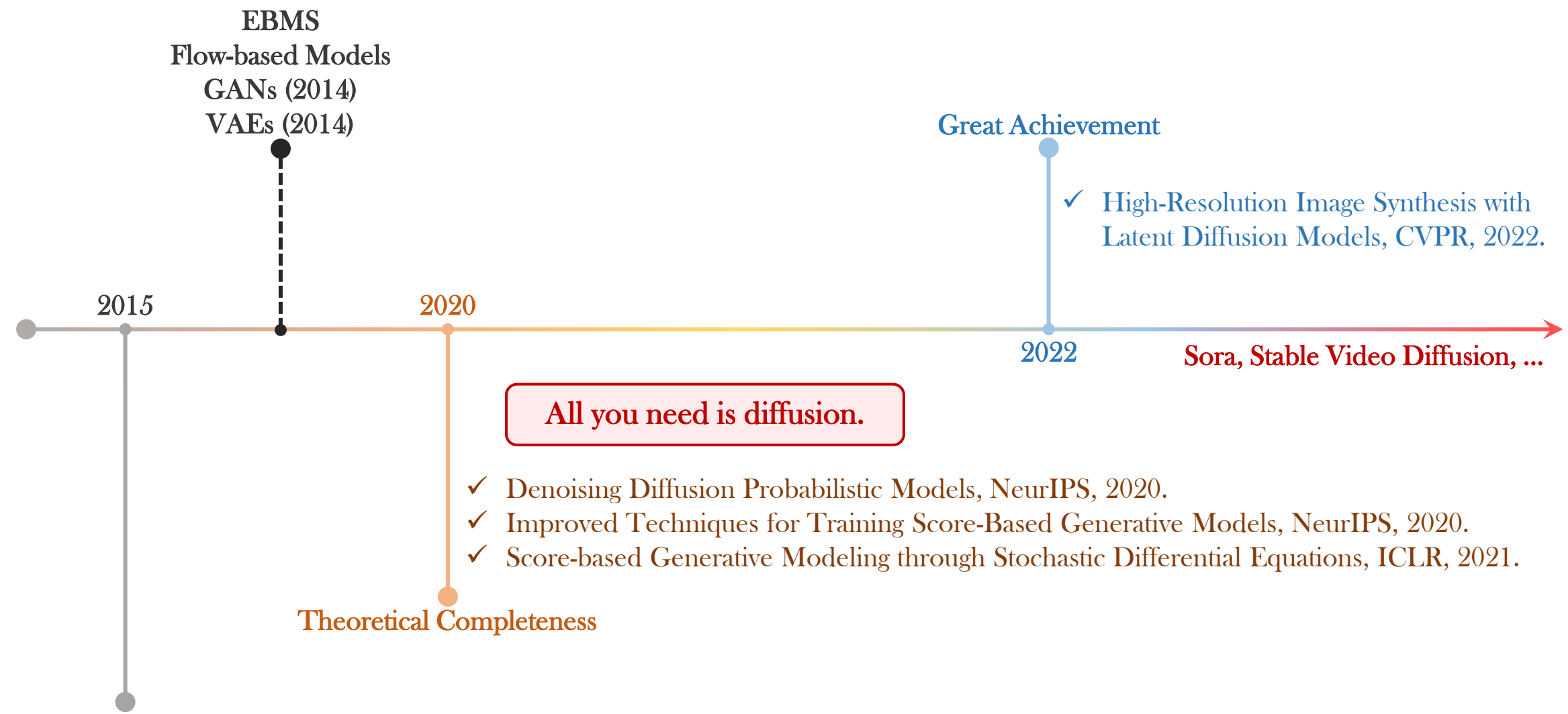
Diffusion Generative Model

Sixu Yan

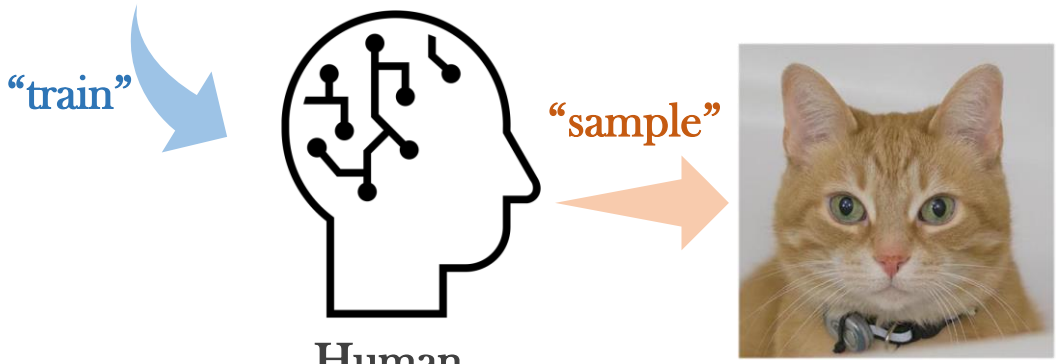
Preliminaries for Diffusion Model

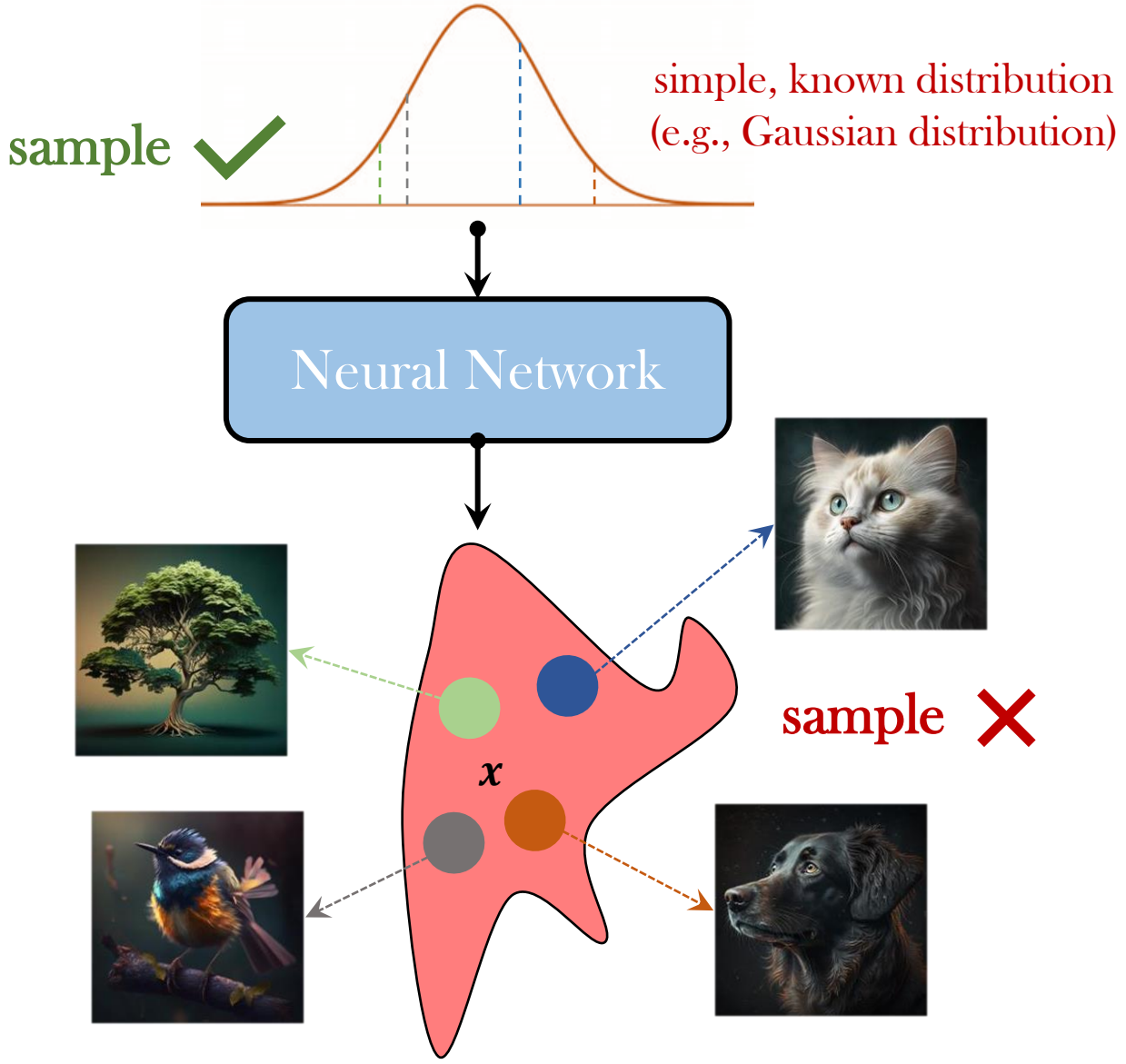
Preliminaries for Diffusion Model



Preliminaries for Diffusion Model

- What do the diffusion models learn ?
  - Learning to generate data → sample from data distribution



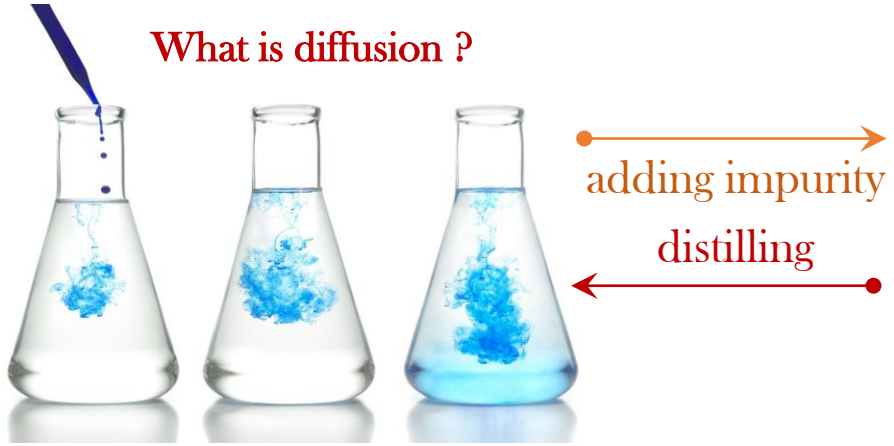


Preliminaries for Diffusion Model

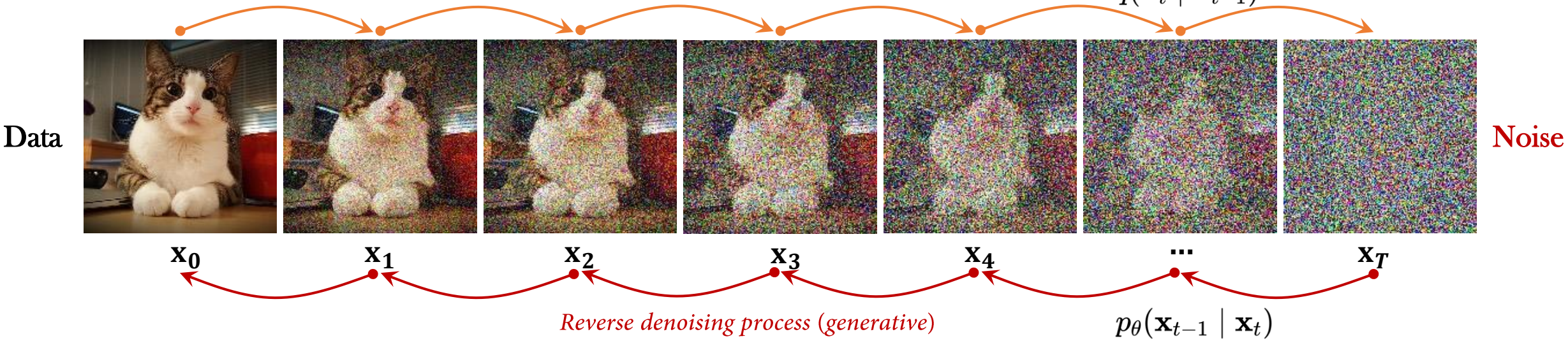
How do Diffusion Models Model Generative Process?

The idea of the diffusion model is not that old. In the 2015 paper called "Deep Unsupervised Learning using Nonequilibrium Thermodynamics", the Authors described it like this:

> *"The essential idea, inspired by non-equilibrium statistical physics, is to systematically and* ***slowly destroy structure in a (clear) data distribution*** *through an* ***iterative forward diffusion process****. We then* ***learn*** *a* ***reverse diffusion process*** *that* ***restores structure in data****, yielding a highly flexible and tractable generative model of the data."*



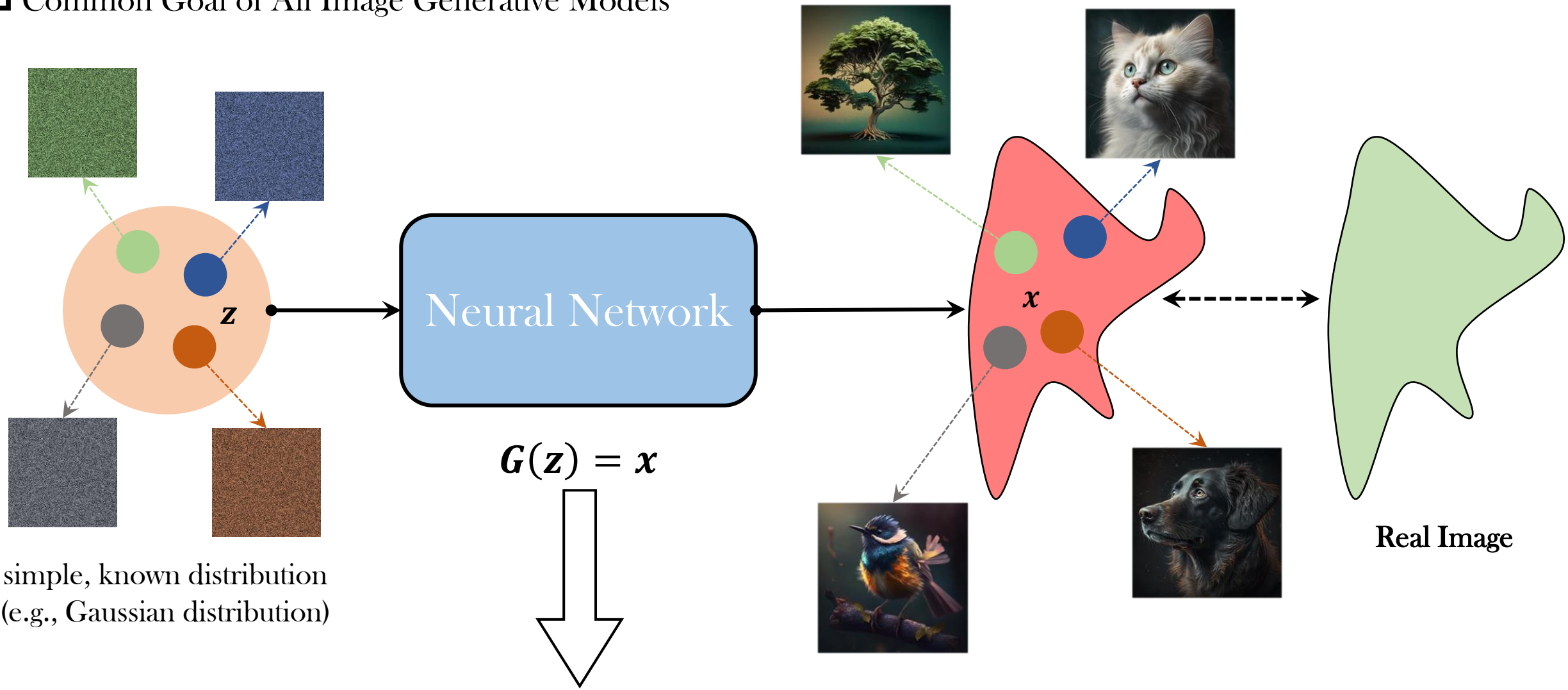
Diffusion rocess of Diffusion Model



What is Generative Model ?

Generative Models

- Common Goal of All Image Generative Models



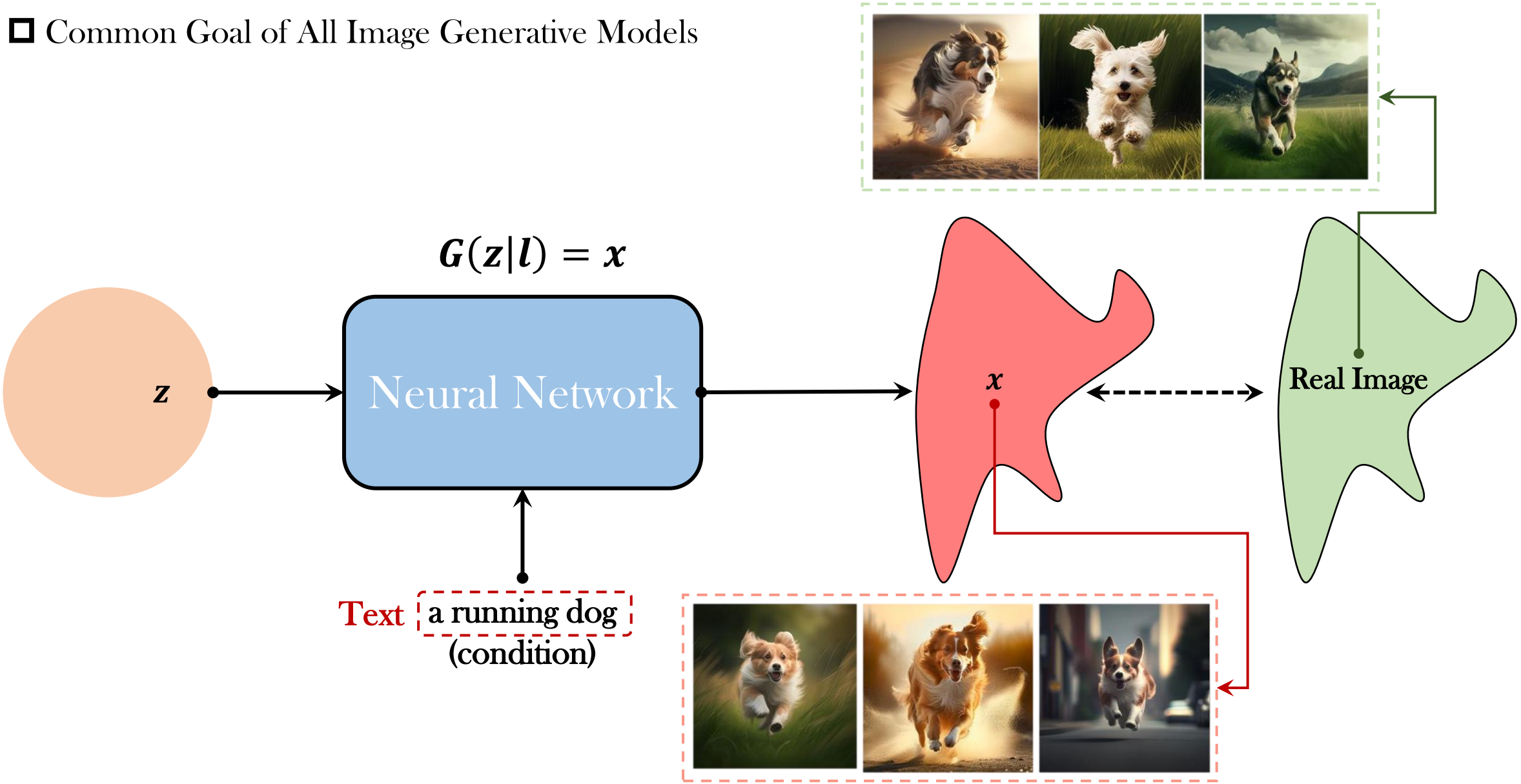
$G(z) = x$

simple, known distribution
(e.g., Gaussian distribution)

mapping from a simple known distribution to an unknown complex distribution (e.g., real image distribution)

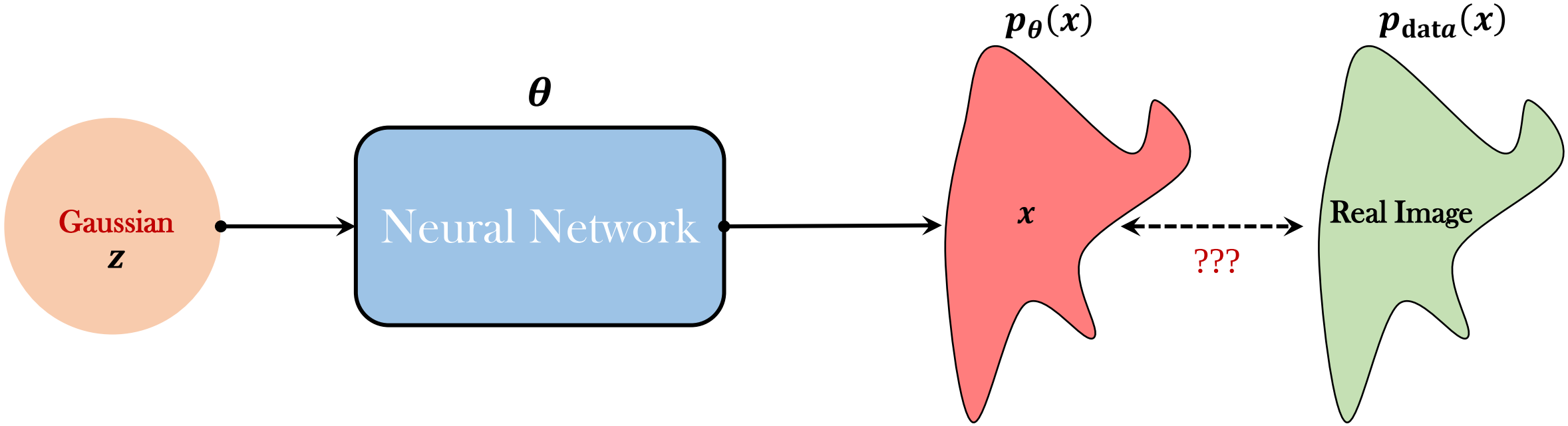
Generative Models

- Common Goal of All Image Generative Models



Generative Models

☐ Common Goal of All Image Generative Models



Step 1: collect $X = \{x^1, x^2, .., x^N\}$ from $p_{\text{data}}(x)$

Step 2: compute $p_\theta(x)$

how to compute $p_\theta(x)$?

$$\theta^* = \arg\max_\theta p_\theta(X) = \arg\max_\theta \prod_{i=1}^{N} p_\theta(x^i)$$

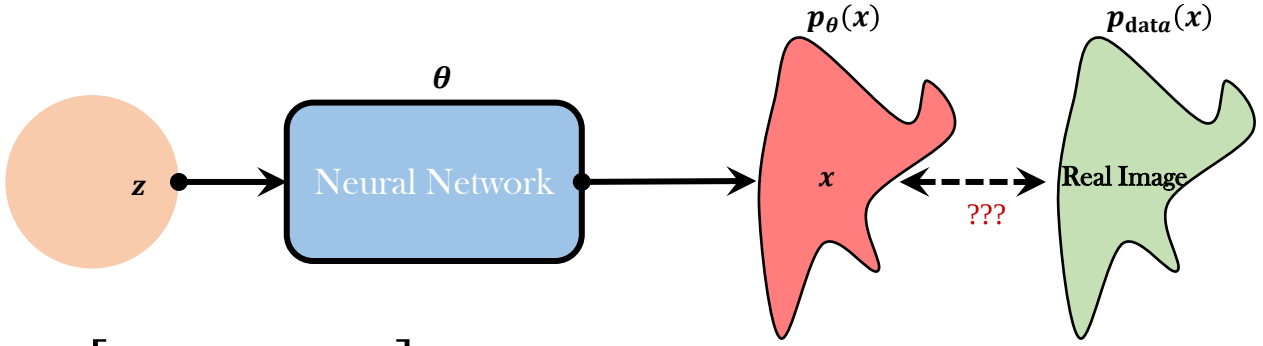
Maximum Likelihood Estimation

Generative Models

- Common Goal of All Image Generative Models

  Step 1: sample $X = \{x^1, x^2, .., x^N\}$ from $p_{\text{data}}(x)$

  Step 2: compute $p_\theta(x)$



$$\theta^* = \arg\max_\theta p_\theta(X) = \arg\max_\theta \prod_{i=1}^{N} p_\theta(x^i) = \arg\max_\theta \left[\log \prod_{i=1}^{N} p_\theta(x^i)\right]$$

$$= \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^i) \approx \arg\max_\theta \mathbb{E}_{x\sim p_{\text{data}}}[\log p_\theta(x)]$$

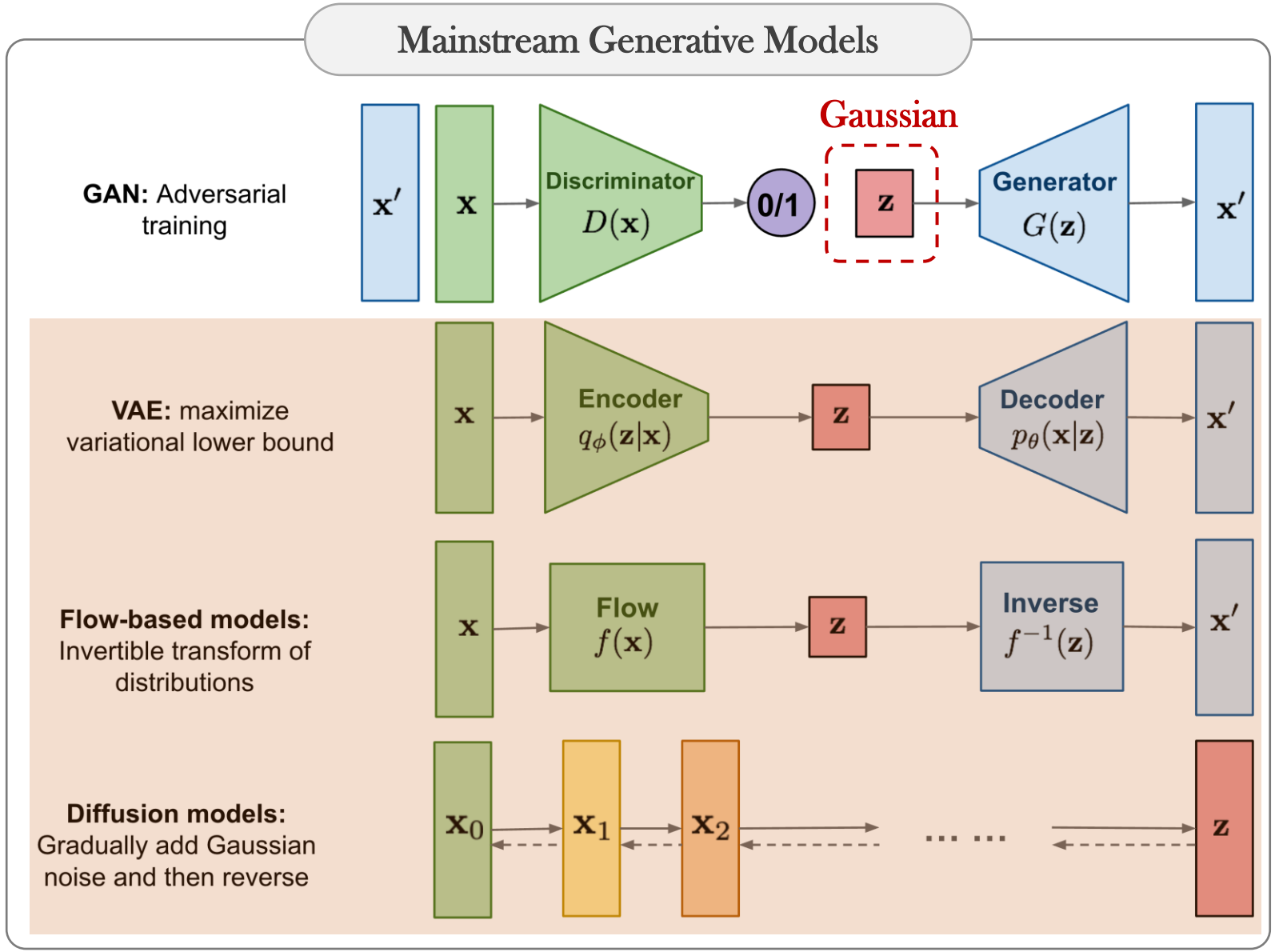
$$= \arg\max_\theta \left[\int_x p_{\text{data}}(x) \log p_\theta(x)\, dx - \int_x p_{\text{data}}(x) \log p_{\text{data}}(x)\, dx\right] \quad \text{(not related to } \theta\text{)}$$

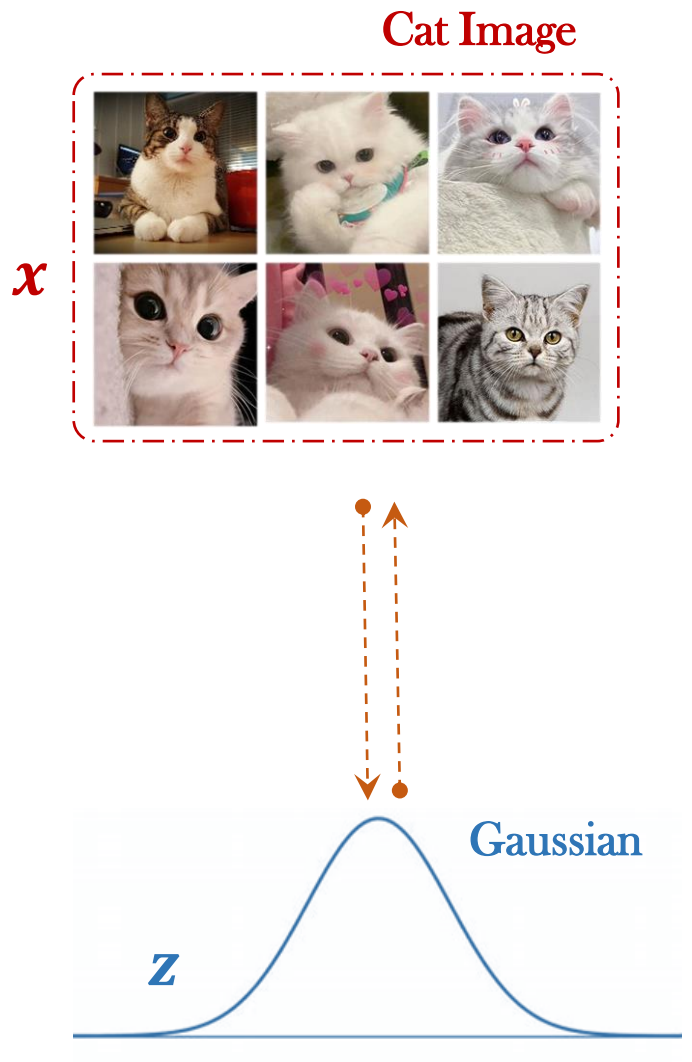
$$= \arg\max_\theta \int_x p_{\text{data}}(x) \log \frac{p_\theta(x)}{p_{\text{data}}(x)}\, dx = \arg\min_\theta D_{\text{KL}}(p_{\text{data}} || p_\theta) \quad \text{(difference between } p_{\text{data}} \text{ and } p_\theta\text{)}$$

Maximum Likelihood = Minimize KL Divergence

Generative Models

Formulation of Generative Process



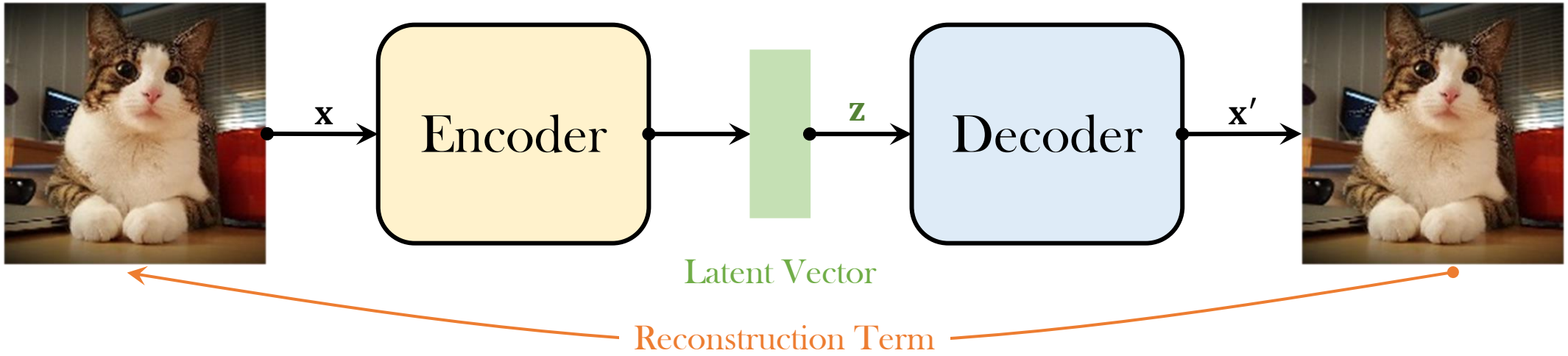


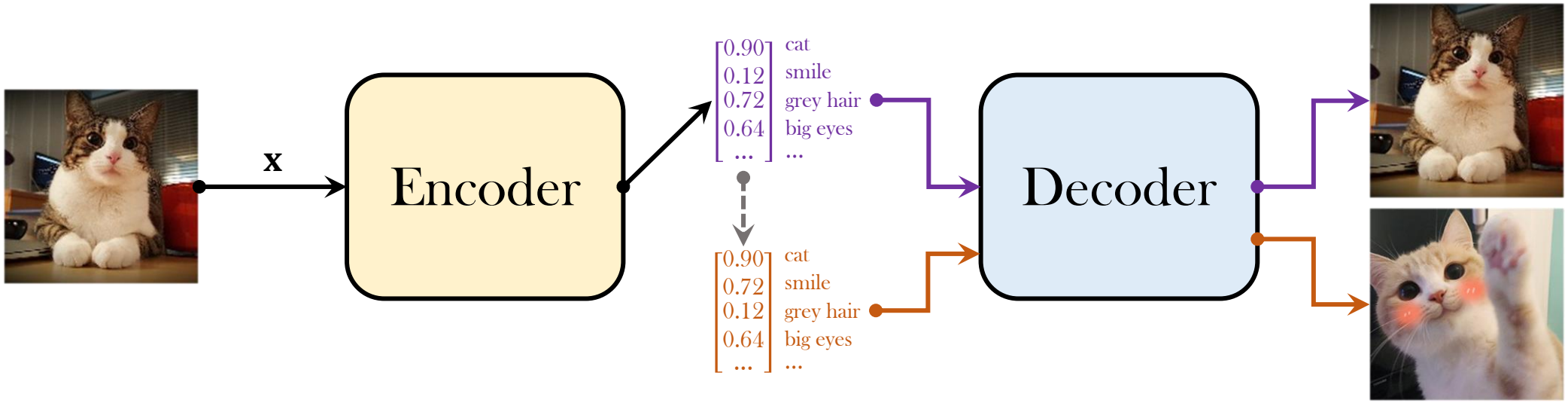
Variational Auto Encoder

Variational Auto Encoder

Variational Auto Encoder (VAE)

- Auto Encoder

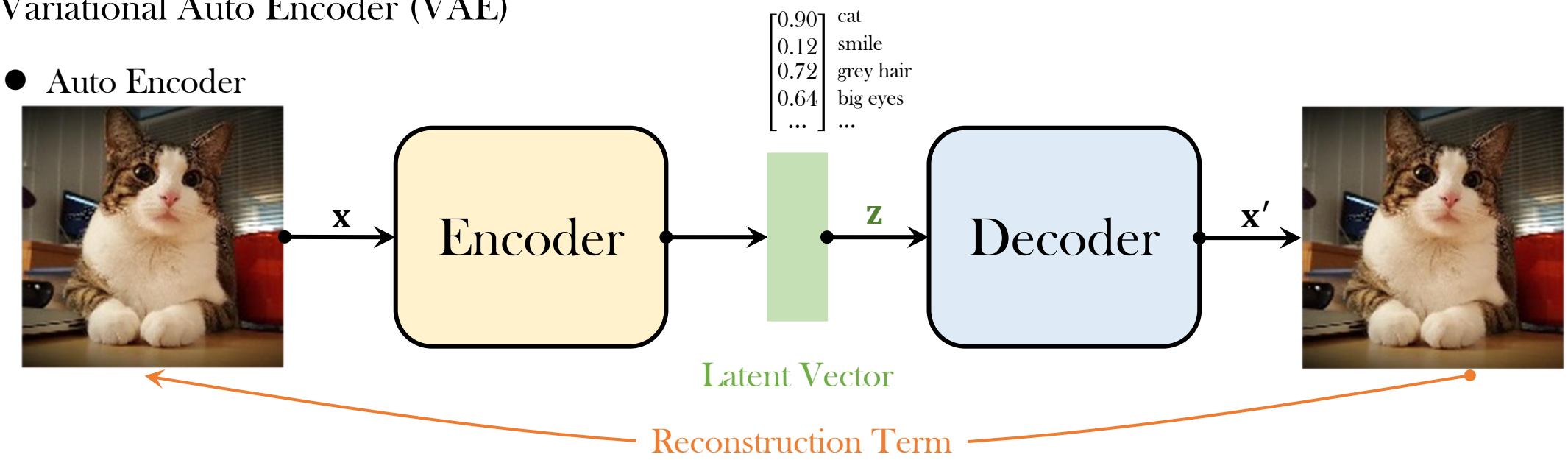




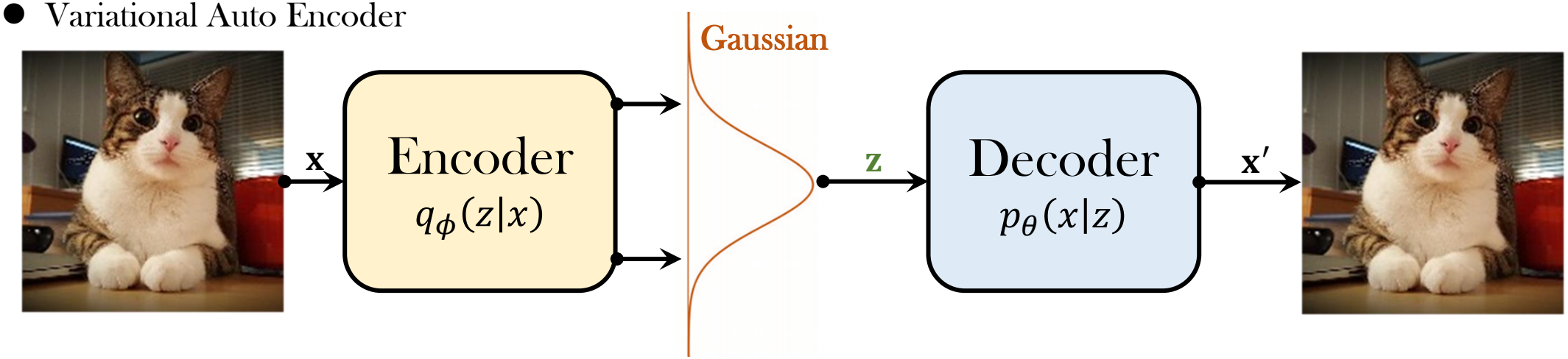
Variational Auto Encoder

Variational Auto Encoder (VAE)

- Auto Encoder



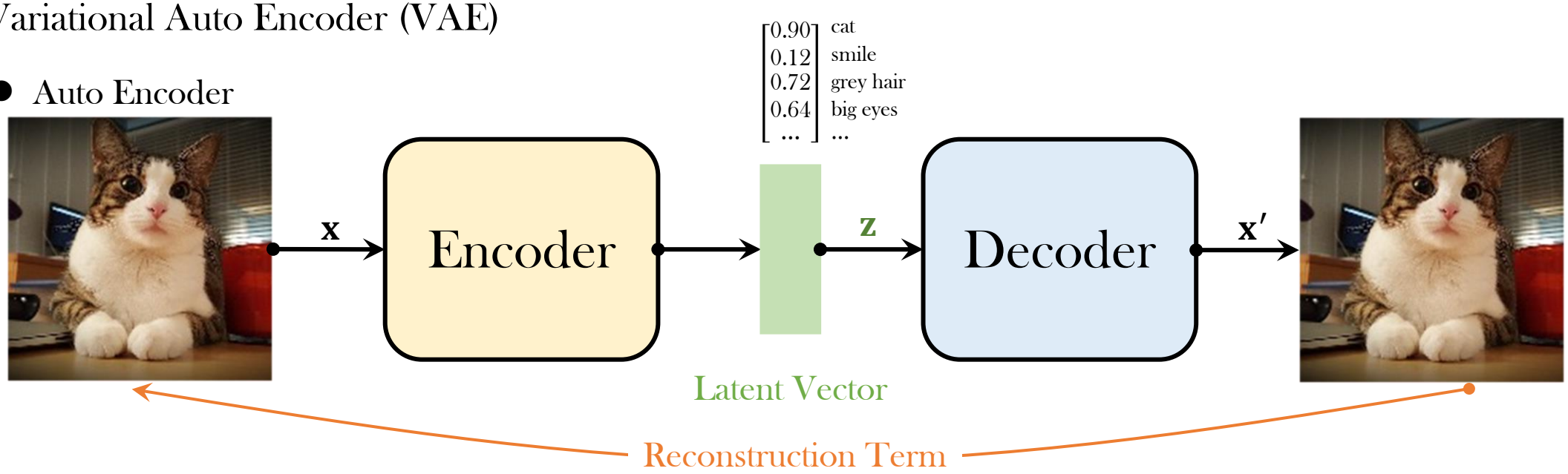
- Variational Auto Encoder



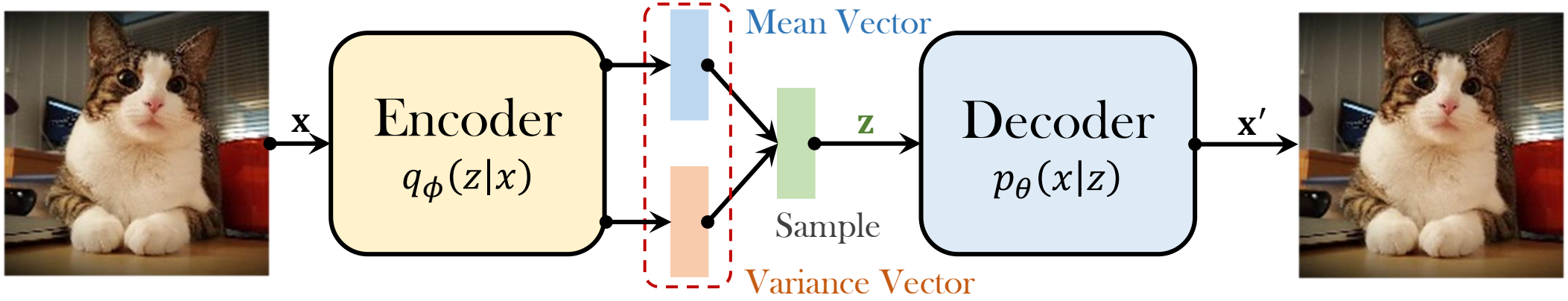
Variational Auto Encoder

Variational Auto Encoder (VAE)

- Auto Encoder



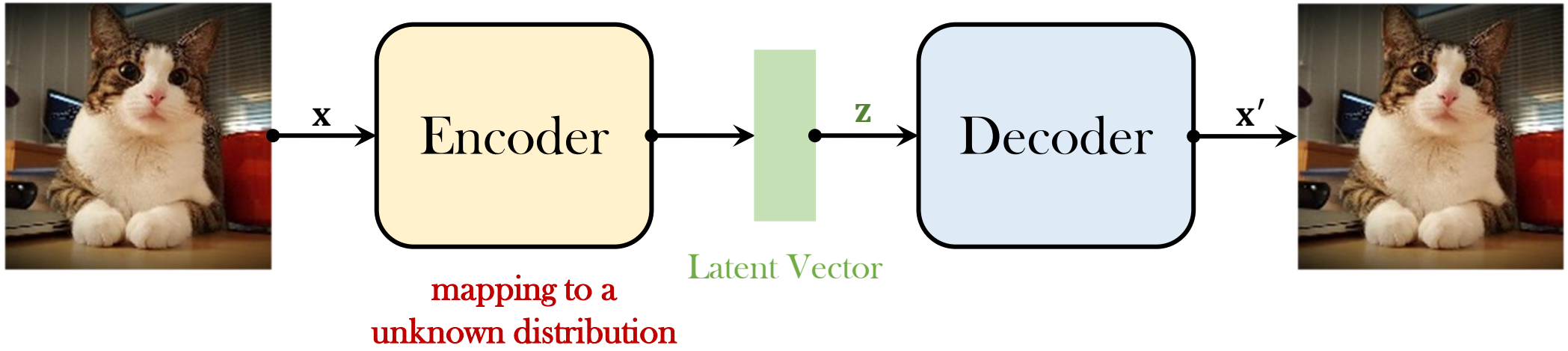
- Variational Auto Encoder



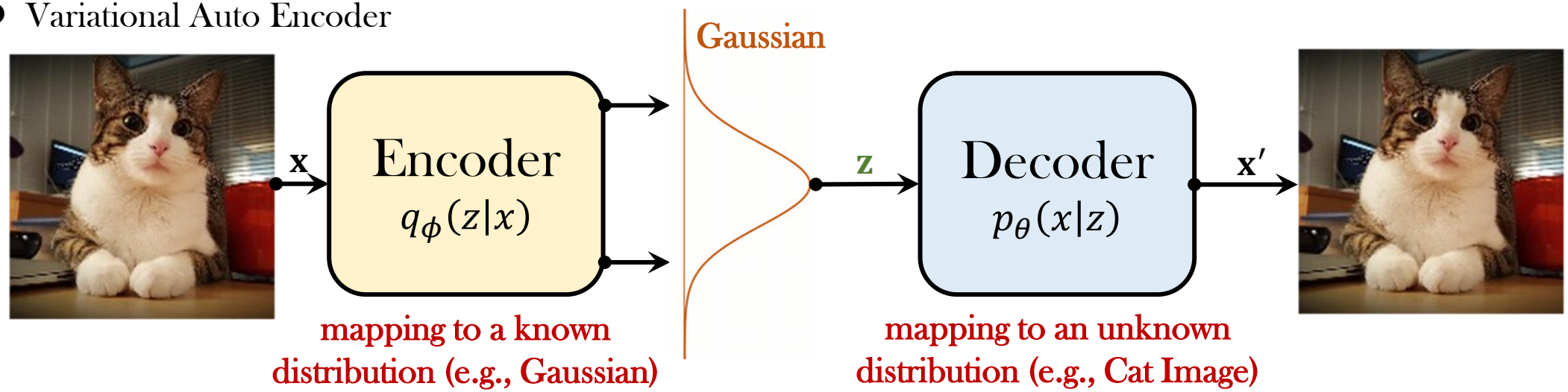
Variational Auto Encoder

Variational Auto Encoder (VAE)

- Auto Encoder

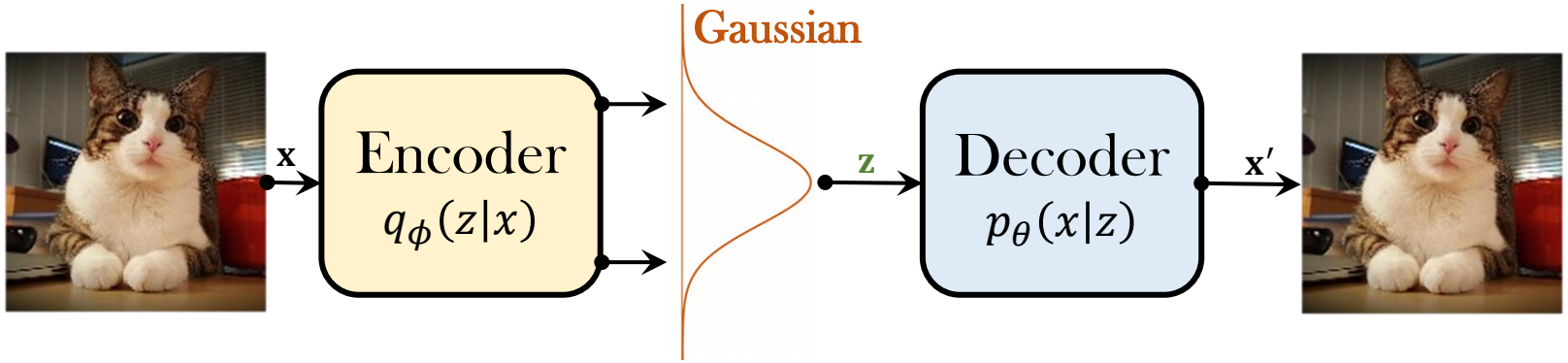


- Variational Auto Encoder



Variational Auto Encoder

Variational Auto Encoder (VAE)



$$\theta^* = \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^i)$$

$X = \{x^1, x^2, .., x^N\}$ from $p_{\text{data}}(x)$

Maximum Likelihood Estimation

$$\begin{aligned}
\log p_\theta(x) &= \log p_\theta(x) \int_z q_\phi(z|x)dz \\
&= \int_z q_\phi(z|x) \log p_\theta(x)\, dz \\
&= \int_z q_\phi(z|x) \log \left(\frac{p_\theta(x,z)}{p_\theta(z|x)}\right) dz \\
&= \int_z q_\phi(z|x) \log \left(\frac{p_\theta(x,z) q_\phi(z|x)}{q_\phi(z|x) p_\theta(z|x)}\right) dz \\
&= \int_z q_\phi(z|x) \log \left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right) dz + \int_z q_\phi(z|x) \log \left(\frac{q_\phi(z|x)}{p_\theta(x,z)}\right) dz \qquad D_{\text{KL}}(q_\phi||p_\theta) \geq 0 \\
&\geq \int_z q_\phi(z|x) \log \left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right) dz = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right]
\end{aligned}$$

lower bound

$$\begin{aligned}
\max \log p_\theta(x) &\approx \max \int_z q_\phi(z|x) \log \left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right) dz \\
&= \max \int_z q_\phi(z|x) \log \left(\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\right) dz \qquad \text{a simple, known distribution} \\
&= \max \int_z q_\phi(z|x) \left(\log \frac{p(z)}{q_\phi(z|x)} + \log p_\theta(x|z)\right) dz \\
&= \max \int_z q_\phi(z|x) \log p_\theta(x|z)\, dz - \int_z q_\phi(z|x) \log \frac{q_\phi(z|x)}{p(z)} dz \\
&= \max \left(\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{\text{KL}}\left(q_\phi(z|x)||p(z)\right)\right)
\end{aligned}$$

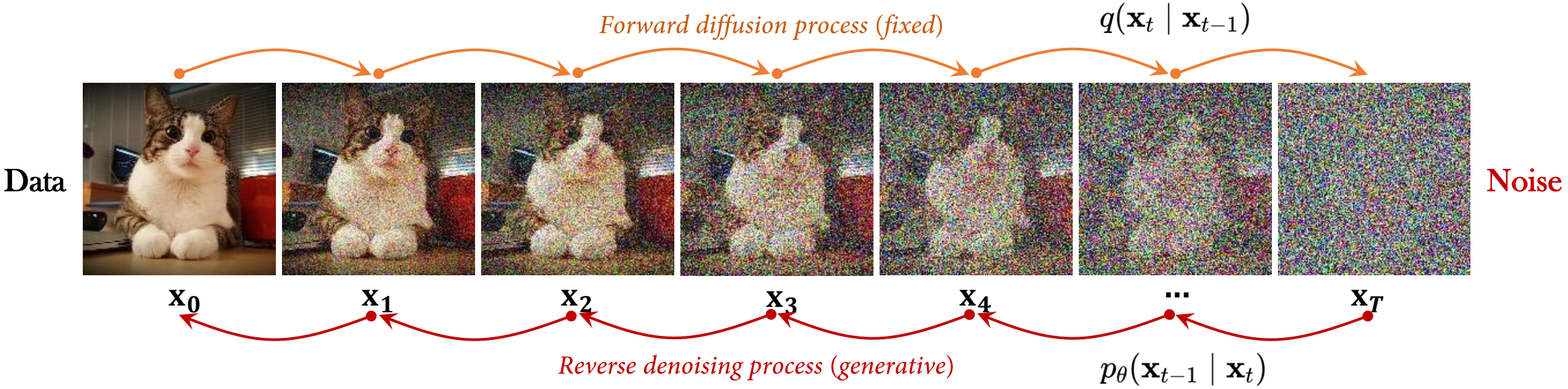
reconstruction term — prior matching term

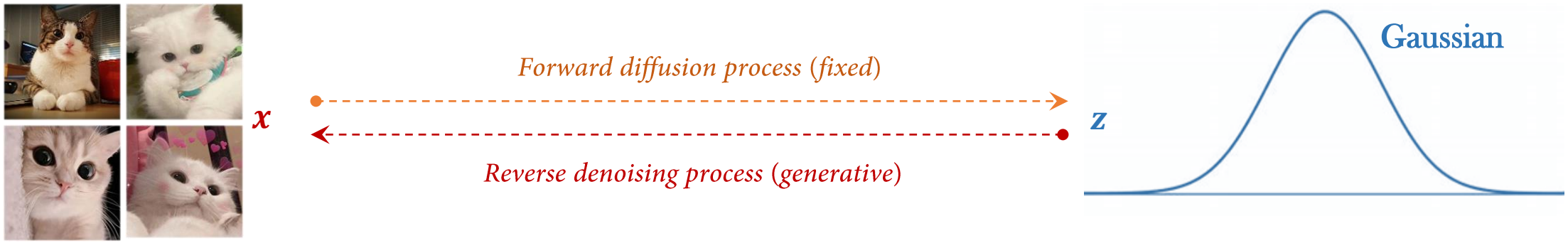
Denoising Diffusion Probabilistic Model

Denoising Diffusion Probabilistic Model

Introduction of DDPM

- Forward diffusion process that gradually adds noise to input
- Reverse denoising process that learns to generate data by denoising



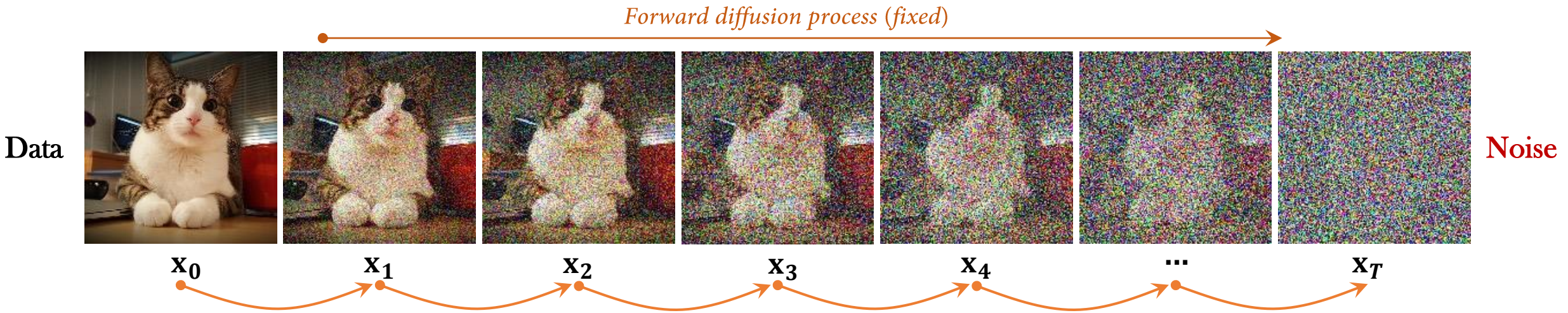


Denoising Diffusion Probabilistic Model

Introduction of DDPM

- Forward Diffusion Process (pre-defined)

The formal definition of the forward process in T steps:



$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}\right) \Rightarrow q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$$

Define $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ $\Rightarrow$ $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$ **Diffusion Kernel**

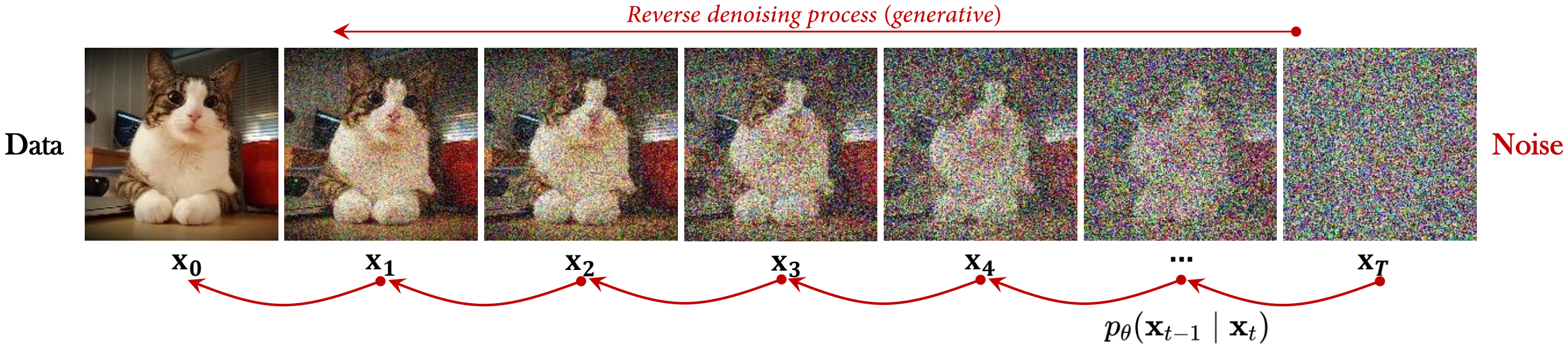
For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\epsilon$, where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

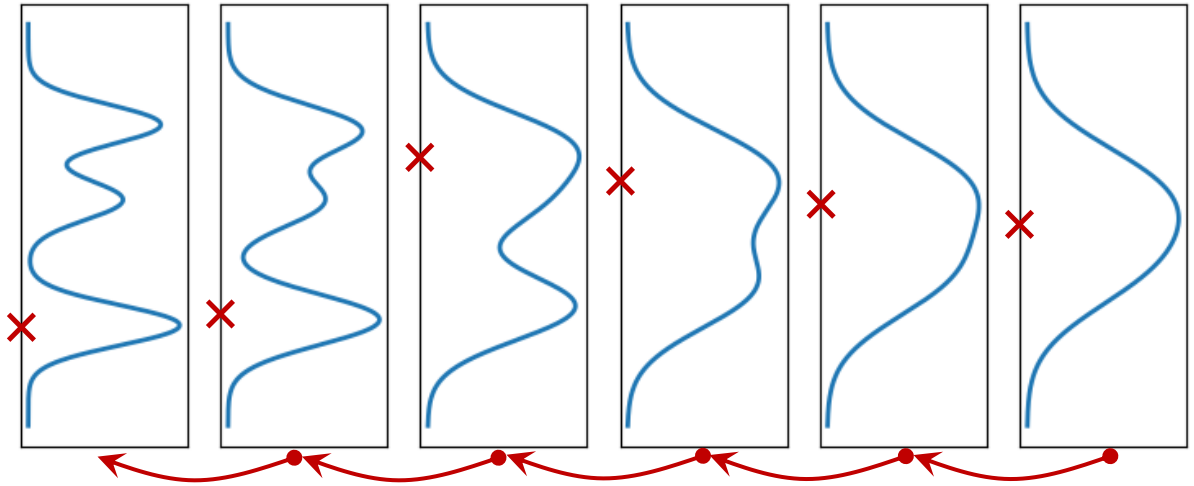
β_t values schedule (i.e., the noise schedule) is designed such that $\bar{\alpha}_T \to 0$ and $q(\mathbf{x}_T \mid \mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$

Denoising Diffusion Probabilistic Model

Introduction of DDPM

- Generative Learning by Denoising & Reverse Denoising Process





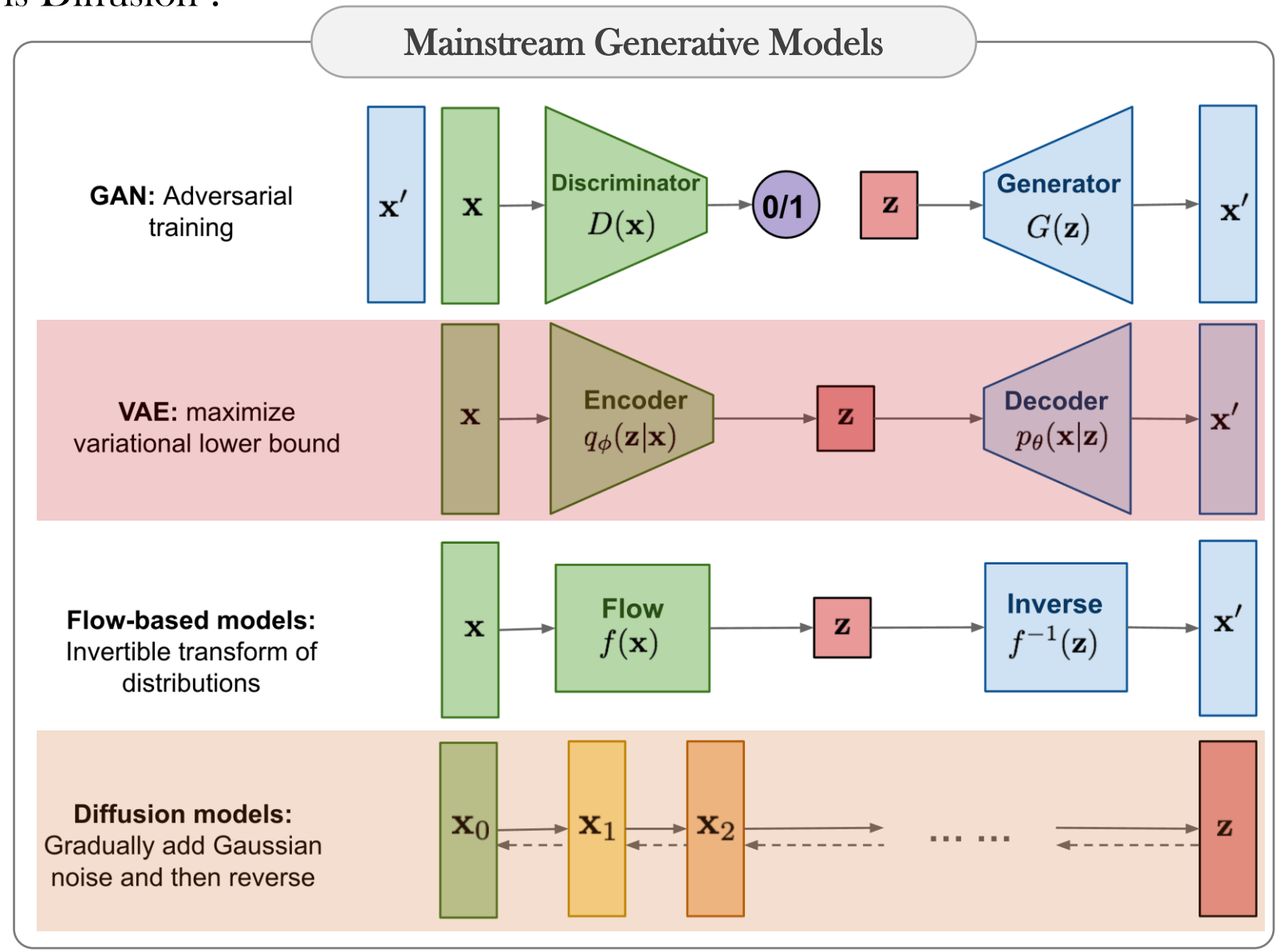
$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2 \mathbf{I}\right)$$

trainable network (U-net, transformer, etc.)

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$$

Denoising Diffusion Probabilistic Model

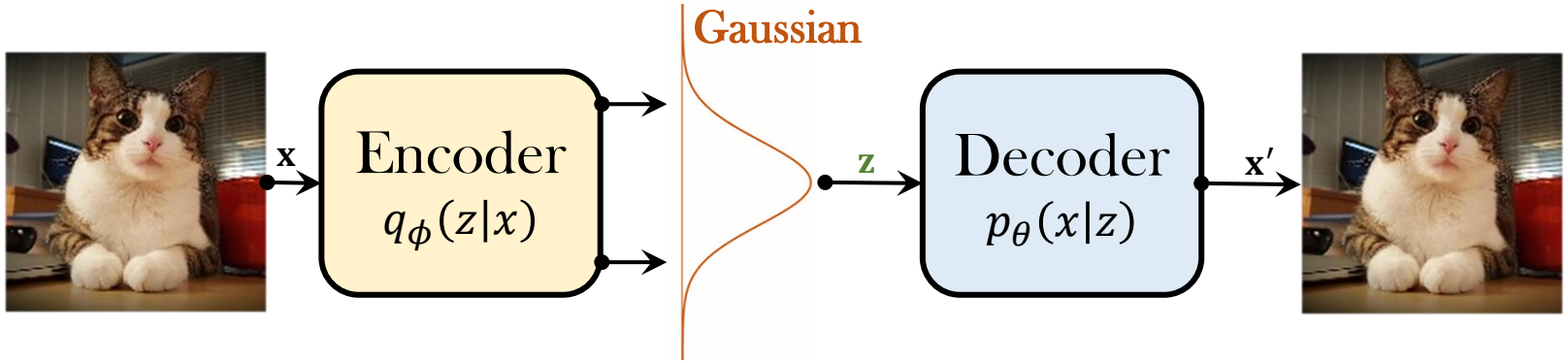
- Why is Diffusion ?



Diffusion vs. VAE

Denoising Diffusion Probabilistic Model

Diffusion vs. VAE



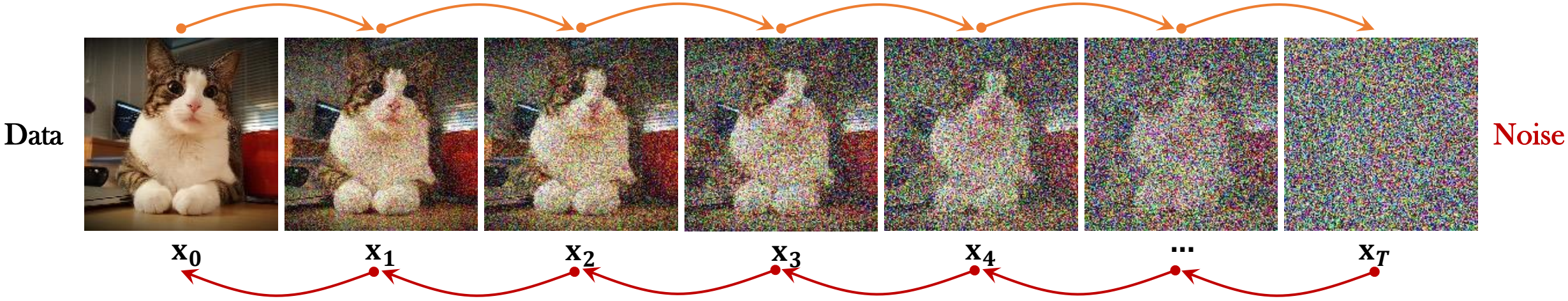
$$\theta^* = \underset{\theta}{\text{argmax}} \sum_{i=1}^{N} \log p_\theta(x^i)$$

$X = \{x^1, x^2, .., x^N\}$ from $p_{\text{data}}(x)$

Maximum Likelihood Estimation

Forward diffusion process (fixed)

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t \mathbf{I}\right)$$

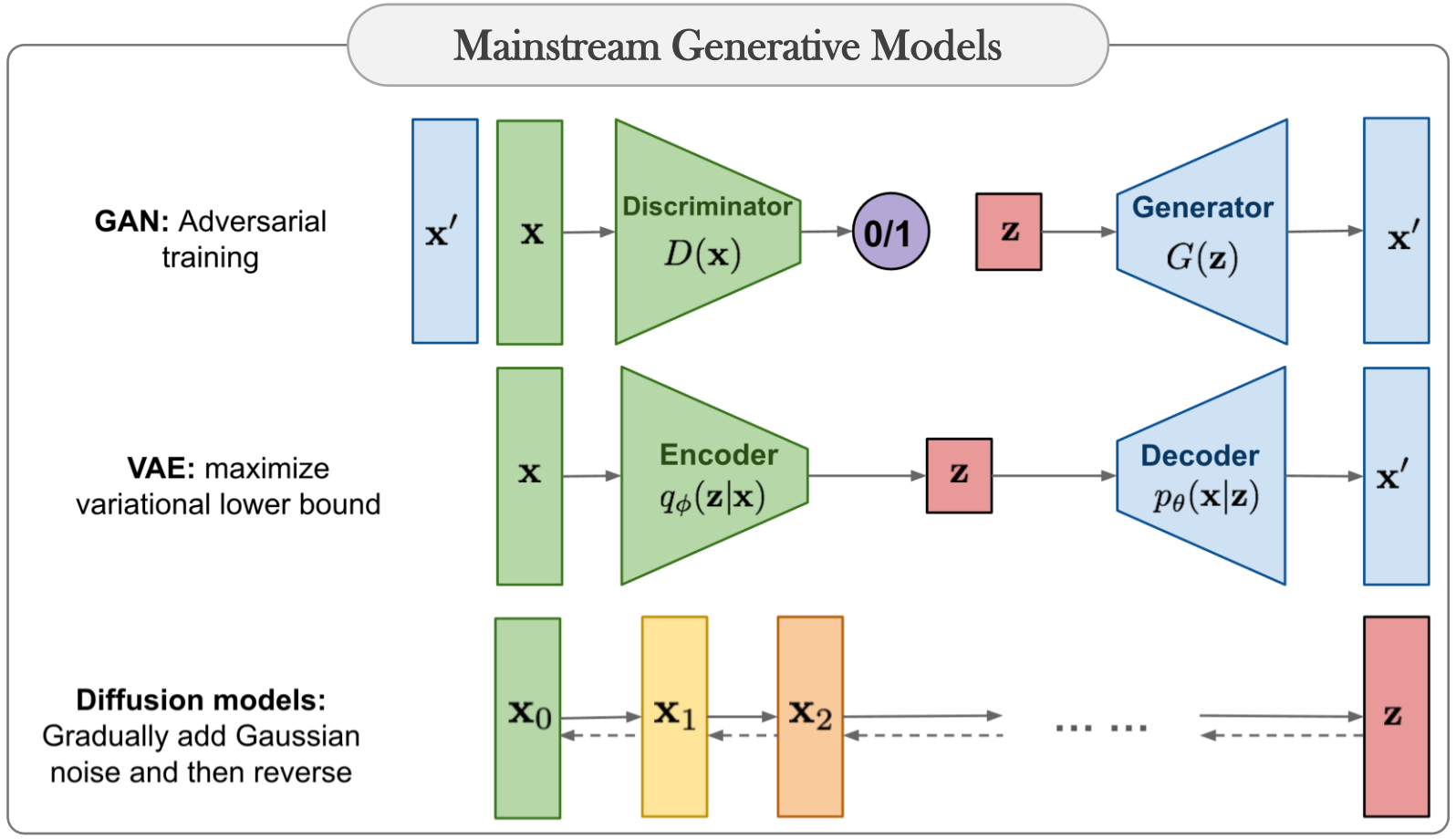


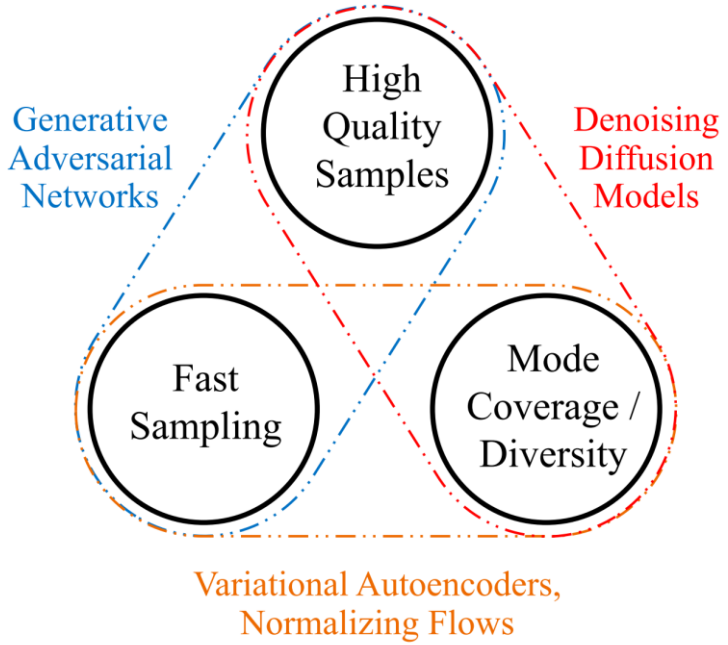
Reverse denoising process (generative)

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2 \mathbf{I}\right)$$

Denoising Diffusion Probabilistic Model

Diffusion vs. VAE and GAN

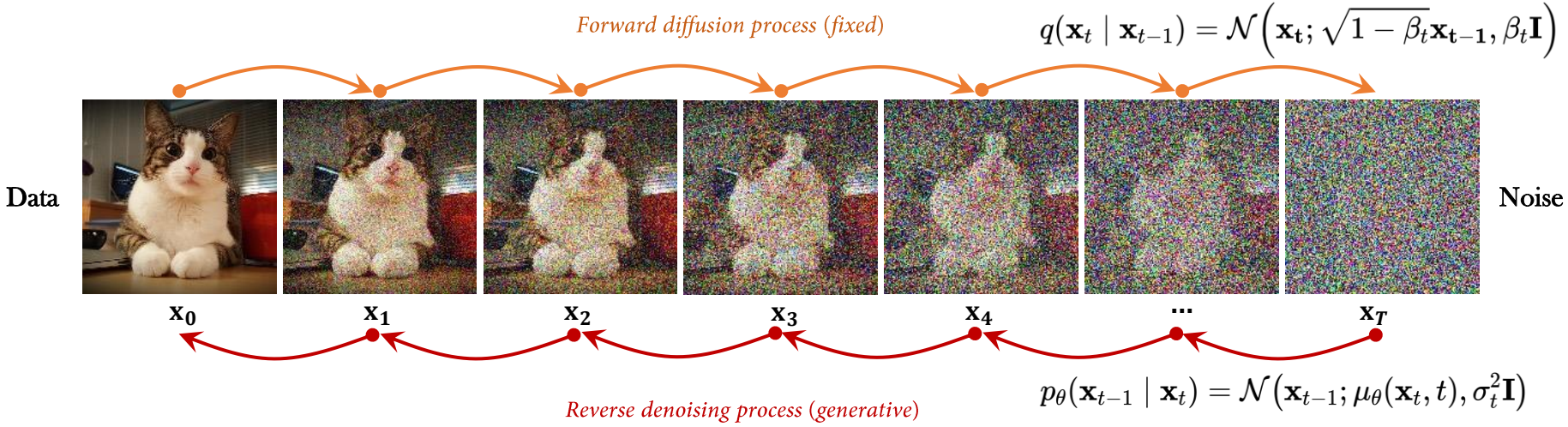




- Tackling the Generative Learning Trilemma with Denoising Diffusion GANs, ICLR, 2022.

Denoising Diffusion Probabilistic Model

Denoising Loss Function of DDPM



$$\theta^* = \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^i)$$

$X = \{x^1, x^2, .., x^N\}$ from $p_{\text{data}}(x)$

Maximum Likelihood Estimation

$$\log p_\theta(x) \geq \underbrace{\mathbb{E}_{q(x_{1:T}|x_0)}\left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]}_{\text{lower bound}}$$

$$= \underbrace{\mathbb{E}_{q(x_1|x_0)}[\log p_\theta(x_0|x_1)]}_{\text{reconstruction term}} - \underbrace{D_{\text{KL}}(q(x_T|x_0)||p(x_T))}_{\text{prior matching term}} - \underbrace{\sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)}\left[D_{\text{KL}}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))\right]}_{\text{denoising matching term}}$$

Diffusion Model

- Understanding Diffusion Models: A Unified Perspective, Arxiv, 2022.

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] = \underbrace{\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]}_{\text{reconstruction term}} - \underbrace{D_{\text{KL}}\left(q_\phi(z|x)||p(z)\right)}_{\text{prior matching term}}$$

VAE

Denoising Diffusion Probabilistic Model

Denoising Loss Function of DDPM

$$\log p_\theta(x) \geq \mathbb{E}_{q(x_{1:T}|x_0)}\left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right] = \mathbb{E}_{q(x_1|x_0)}[\log p_\theta(x_0|x_1)] - \underbrace{D_{\mathrm{KL}}(q(x_T|x_0)||p(x_T))}_{\text{no trainable parameters}} - \sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)}[D_{\mathrm{KL}}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t))]$$

Recall that the KL Divergence between two Gaussian distributions is:

$$D_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x) \parallel \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)) = \frac{1}{2}\left[\log \frac{|\boldsymbol{\Sigma}_y|}{|\boldsymbol{\Sigma}_x|} - d + \mathrm{tr}(\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_x) + (\boldsymbol{\mu}_y - \boldsymbol{\mu}_x)^T \boldsymbol{\Sigma}_y^{-1}(\boldsymbol{\mu}_y - \boldsymbol{\mu}_x)\right]$$

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0)q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \propto \mathcal{N}(\boldsymbol{x}_{t-1}; \underbrace{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}}_{\mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0)}, \underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)})$$

That is why we model p_θ as Gaussian distribution

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \qquad \sigma_q^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$$

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 \mathbf{I}\right)$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1-\bar{\alpha}_t} \qquad \sigma_q^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$$

Denoising Diffusion Probabilistic Model

Denoising Loss Function of DDPM

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0)q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \propto \mathcal{N}(\boldsymbol{x}_{t-1}; \underbrace{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}}_{\mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0)}, \underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)})$$

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \qquad \sigma_q^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$$

forward diffusion process: $q(x_t \mid x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right)$ ⇒ $\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}{\sqrt{\bar{\alpha}_t}}$

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}$$

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2\mathbf{I}\right)$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1-\bar{\alpha}_t} \qquad \sigma_q^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$$

Denoising Diffusion Probabilistic Model

Denoising Loss Function of DDPM

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)=\frac{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{x}_0)q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\propto\mathcal{N}(\boldsymbol{x}_{t-1};\underbrace{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t+\sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t}}_{\mu_q(\boldsymbol{x}_t,\boldsymbol{x}_0)},\underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)})$$

$$\boldsymbol{\mu}_q(\boldsymbol{x}_t,\boldsymbol{x}_0)=\frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}\qquad\sigma_q^2(t)=\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$$

$$p_\theta(x_{t-1}\mid x_t)=\mathcal{N}\left(x_{t-1};\mu_\theta(x_t,t),\sigma_t^2\mathbf{I}\right)$$

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)=\frac{1}{\sqrt{\alpha_t}}\boldsymbol{x}_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\sqrt{\alpha_t}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t,t)\qquad\sigma_q^2(t)=\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$$

Denoising is king.

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0\sim q(\mathbf{x}_0)$
3: $t\sim\text{Uniform}(\{1,\dots,T\})$
4: $\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t)\right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T\sim\mathcal{N}(\mathbf{0},\mathbf{I})$
2: **for** $t=T,\dots,1$ **do**
3: $\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I})$ if $t>1$, else $\mathbf{z}=\mathbf{0}$
4: $\mathbf{x}_{t-1}=\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t-\frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right)+\sigma_t\mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

- Denoising Diffusion Probabilistic Models, NeurIPS, 2020.

Denoising Diffusion Probabilistic Model

Training Process of DDPM

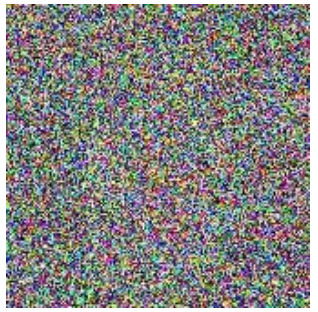
Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

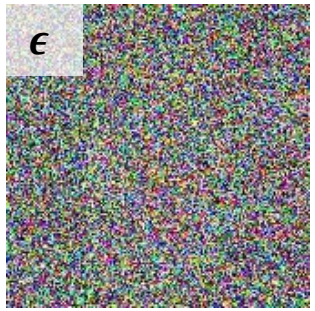
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

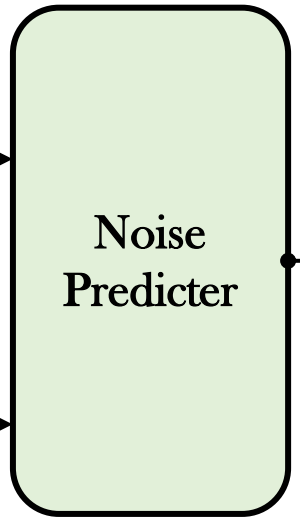
$\boldsymbol{x}_0$ clean image



$\boldsymbol{\epsilon}$ random noise

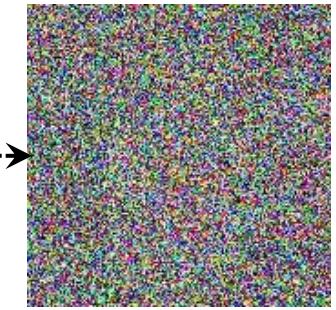
t diffusion time step

$\sqrt{\bar{\alpha}_t}$  + $\sqrt{1-\bar{\alpha}_t}$  = 



t

?



Denoising Diffusion Probabilistic Model

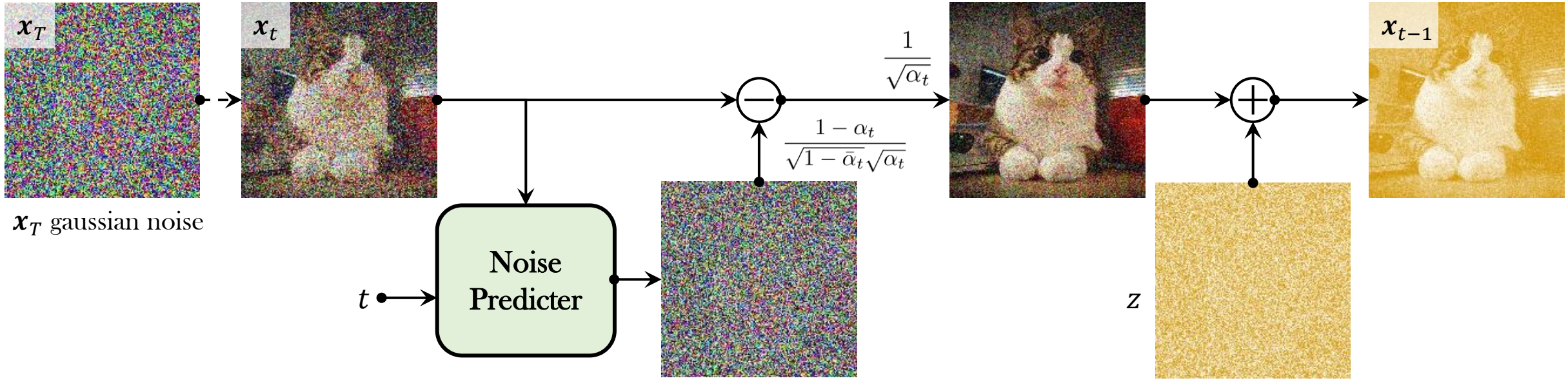
Inference Process of DDPM

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

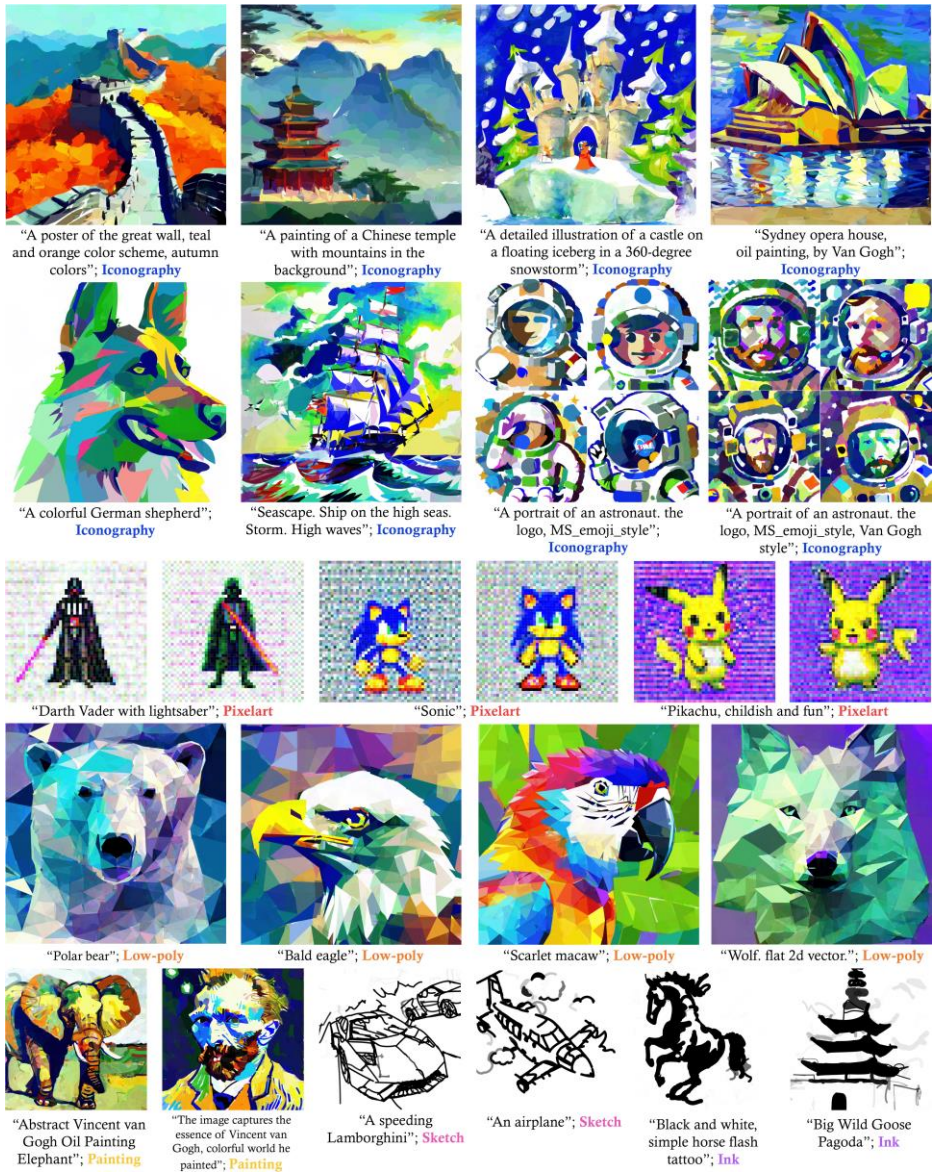
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do** — sample a noise ?
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$



Mainstream Work for Diffusion Models

Mainstream Work for Diffusion Models

Image Generation



- DEADiff: An Efficient Stylization Diffusion Model with Disentangled Representations, CVPR, 2024.
- SVGDreamer: Text guided SVG generation with diffusion model, CVPR, 2024. 2D Style Transfer

Mainstream Work for Diffusion Models

Image Generation

Imagen Family DALL Family Stable Diffusion Family

Image Caption: An image depicting in the morning. A *brown* cute teddy bear, a *purple* cute teddy bear, a *yellow* cute teddy bear, a *blue* cute teddy bear all standing side by side on a *red* brick road. The scene should be set in front of *Pink* Castle with clear *blue* sky overhead, punctuated by fluffy *white* clouds, and trees with *green* leaves. The *pink* castle should loom majestically in the background. **Instance Captions:** 1-4) A *brown*/*purple*/*yellow*/*blue* teddy bear; 5) a *red* brick road; 6) *Pink* Castle; 7-8) *green* leaves; 9-10) fluffy *white* clouds

a) Diverse Instance Attributes and Locations

Image Caption: Craft an oil painting: Picture a seaside garden drenched in radiant hues of roses, lilies, and lavender, transitioning gracefully into the expansive azure ocean and blue sky. Integrate a weathered, rustic pathway with steps that invite viewers towards the water's edge, complemented by a prominent bouquet of flowers and plants.
Instance Captions: 1-19) roses; 20) sky; 21) ocean; 22) pathway with steps; 23) bouquet of flowers; 24) plant; 25-26) plants

b) Dense Small Objects

Image Caption: An image of two little husky puppy in a wicker basket.
Instance Captions: 1) a husky puppy sitting in a wicker basket + *Mask*. 2) a black and white husky puppy in a blue towel + *Box*. 3) two husky puppies sitting in a wicker basket + *Scribble*. 4) a blue towel + *Point*

c) Various Location Conditions (box, mask, scribble, point)

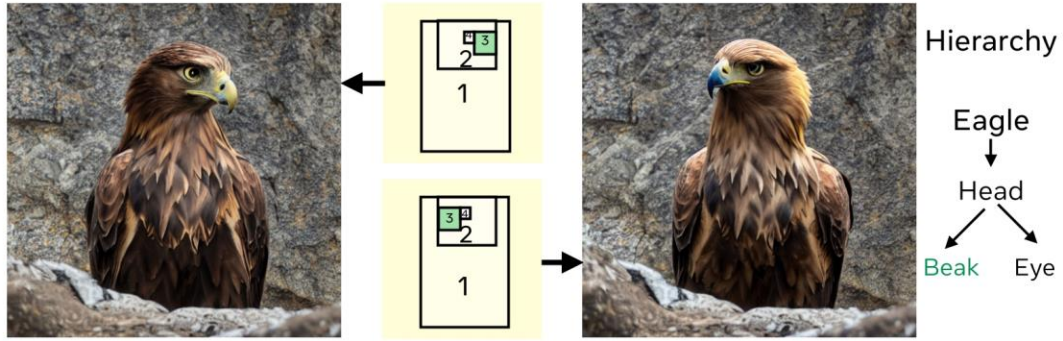


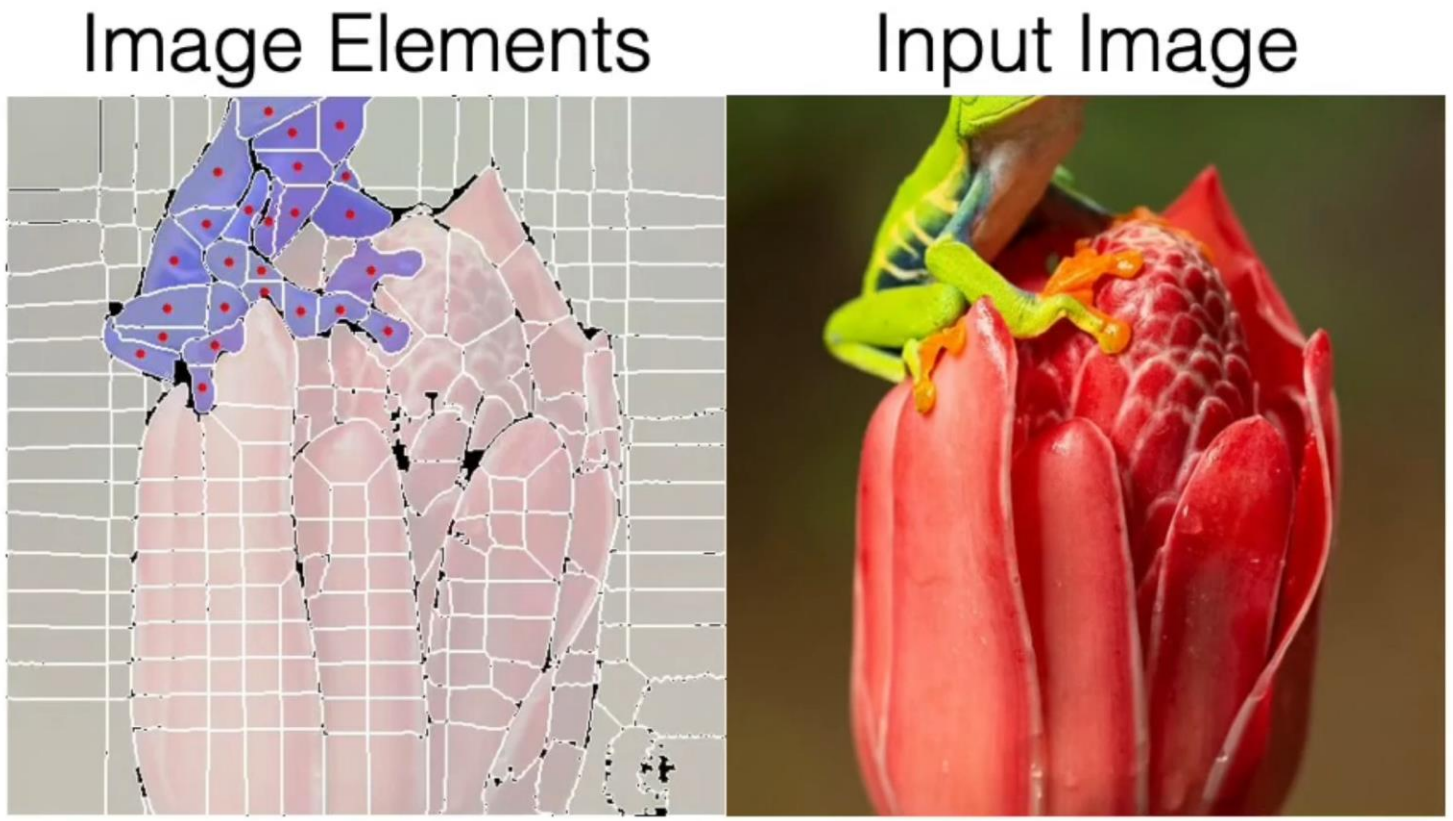
Image Caption: A golden eagle perched on a rugged rock.
Instance Caption: 1) A golden eagle; 2) Eagle's head; 3) **Eagle's beak**; 4) Eagle's eye

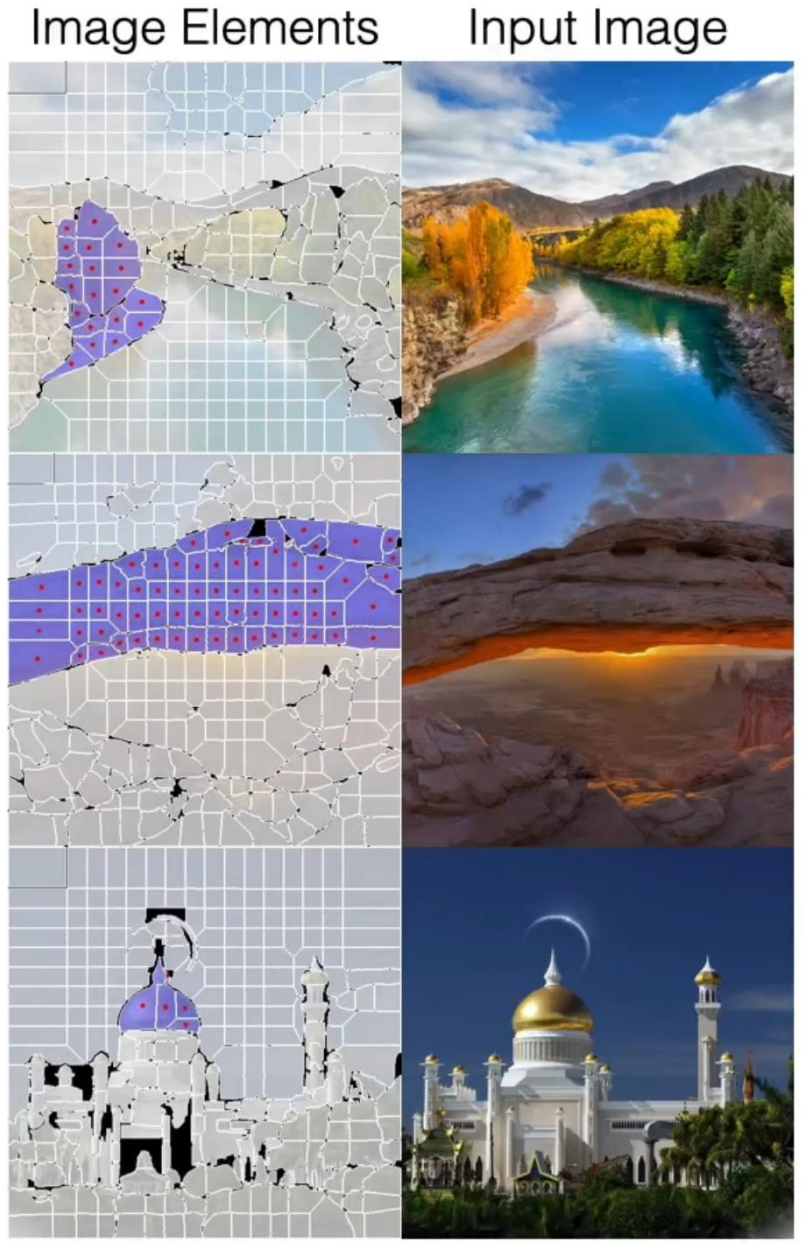
d) Image Composition with Whole Instance, Part and Subpart

- Instancediffusion: Instance-level control for image generation, CVPR, 2024. Instructed 2D Generation

Mainstream Work for Diffusion Models

Image Generation

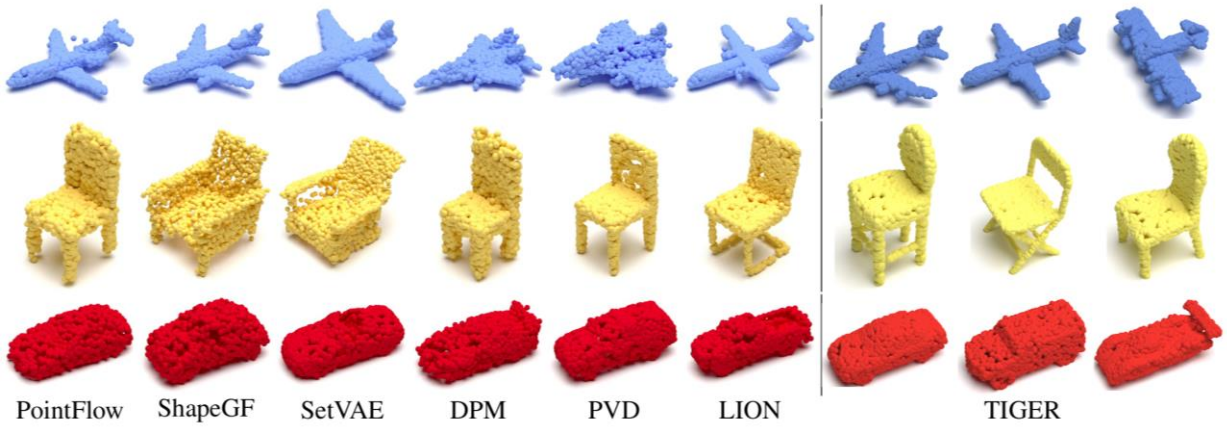




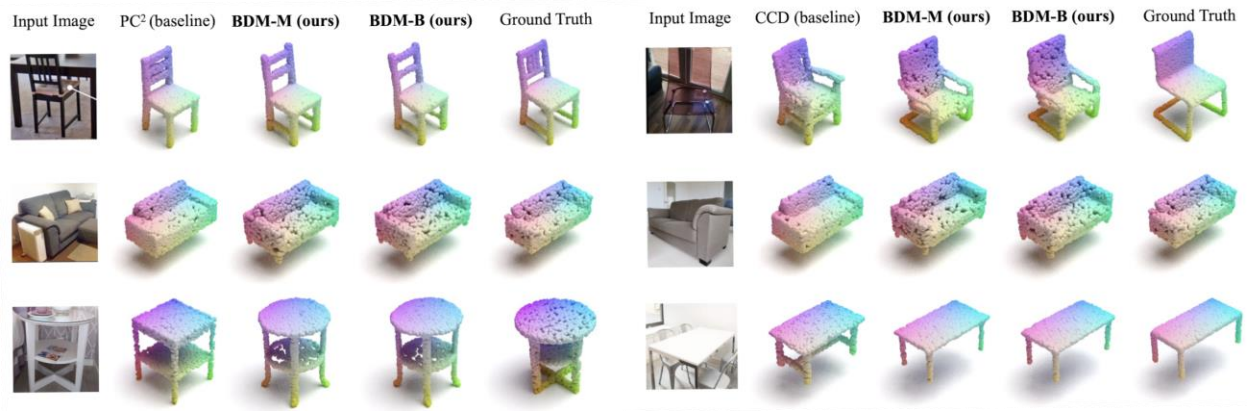
- Editable Image Elements for Controllable Synthesis, ECCV, 2024. Image Editor

Mainstream Work for Diffusion Models

3D Generation



- TIGER: Time-Varying Denoising Model for 3D Point Cloud Generation with Diffusion Process, CVPR, 2024. Unconditional 3D Generation



- Bayesian Diffusion Models for 3D Shape Reconstruction, CVPR, 2024. 2D-to-3D

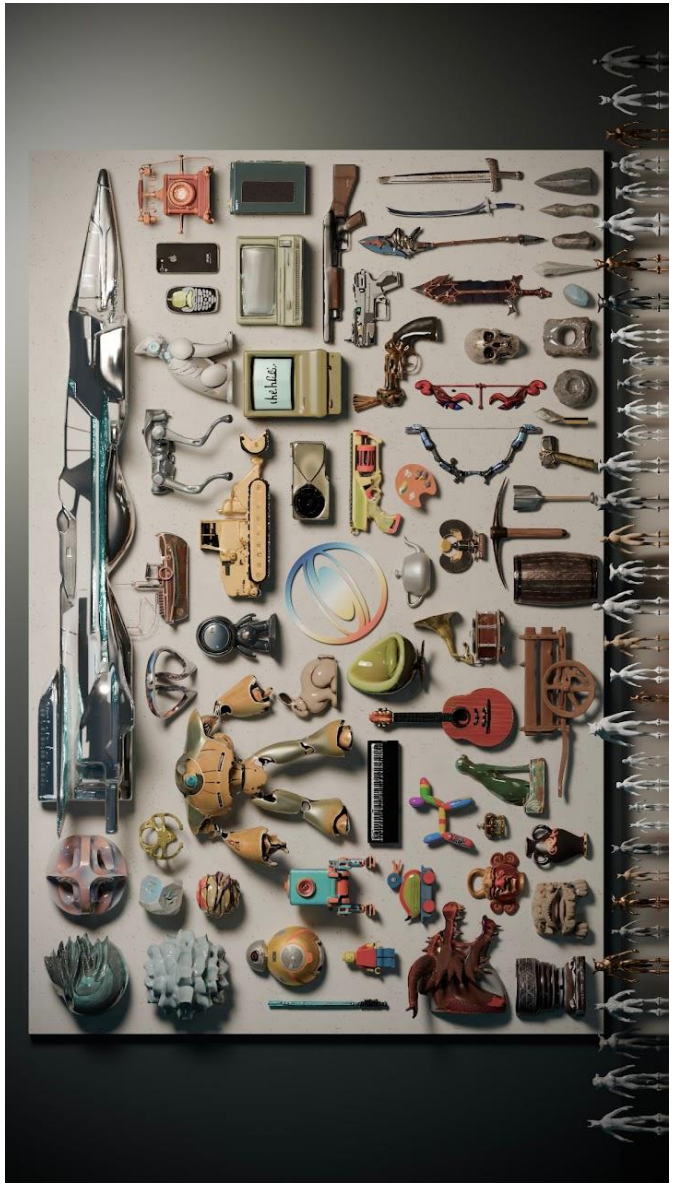
- RichDreamer: A Generalizable Normal-Depth Diffusion Model for Detail Richness in Text-to-3D, CVPR, 2024. Text-to-3D

- Paint3D: Paint Anything 3D with Lighting-Less Texture Diffusion Models, CVPR, 2024. 3D Texture Generation

Mainstream Work for Diffusion Models

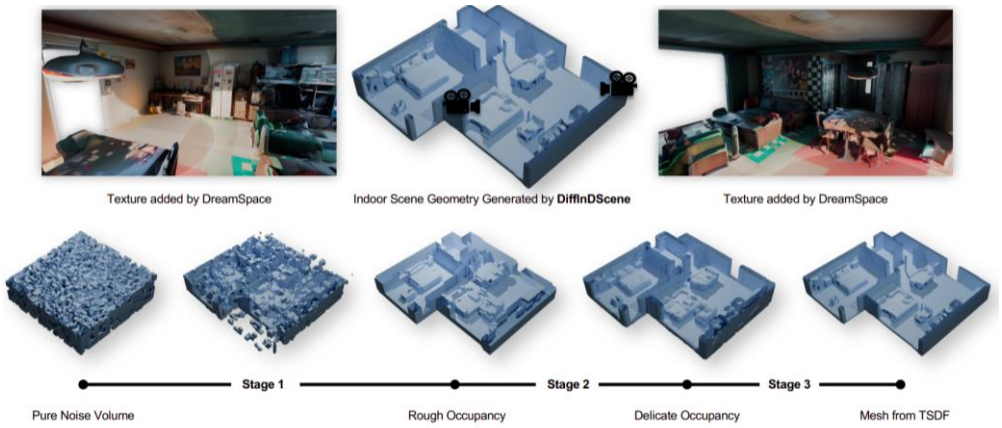
3D Generation

- CLAY: A Controllable Large-scale Generative Model for Creating High-quality 3D Assets, SIGGRAPH, 2024.

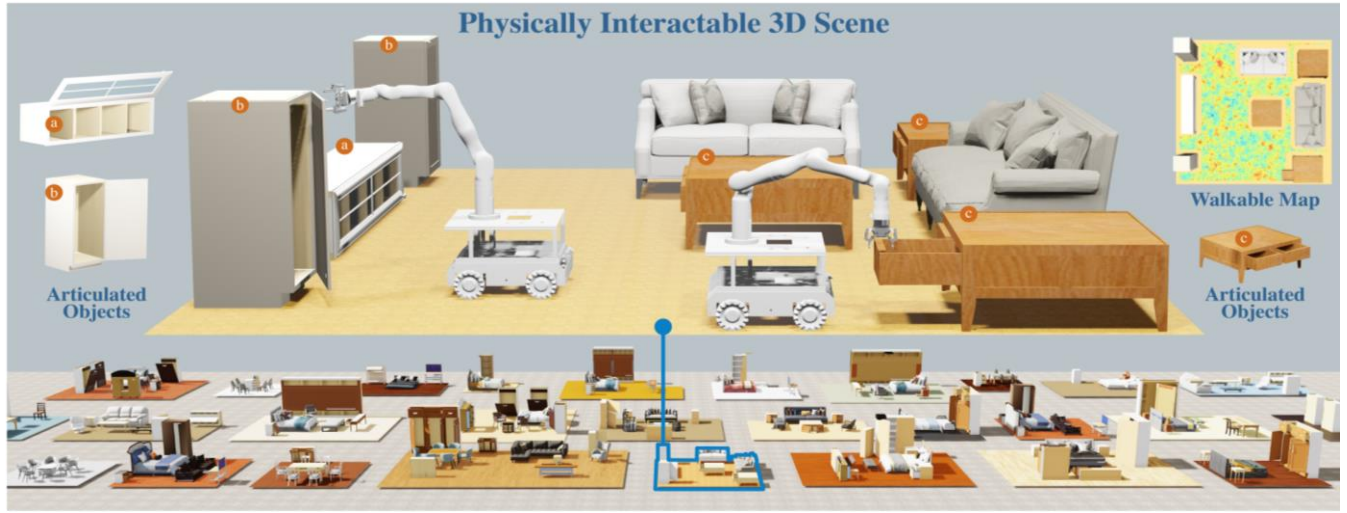


Mainstream Work for Diffusion Models

Scene Generation



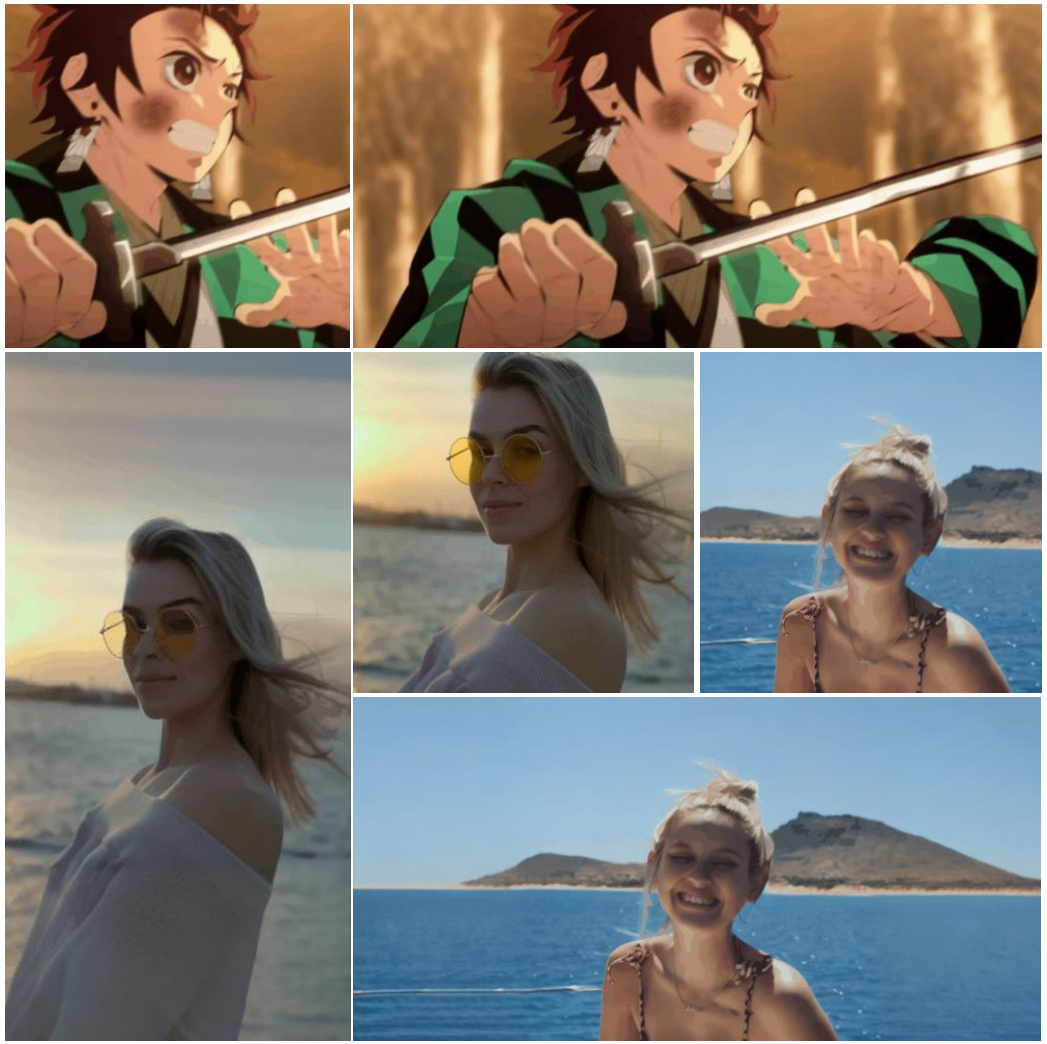
- DiffInDScene: Diffusion-based High-Quality 3D Indoor Scene Generation, CVPR, 2024.



- PhyScene: Physically Interactable 3D Scene Synthesis for Embodied AI, CVPR, 2024.

Mainstream Work for Diffusion Models

Video Generation



- Be-Your-Outpainter: Mastering Video Outpainting through Input-Specific Adaptation, ECCV, 2024. Video Outpainting

Sora – Open AI

Gen3 - Runway

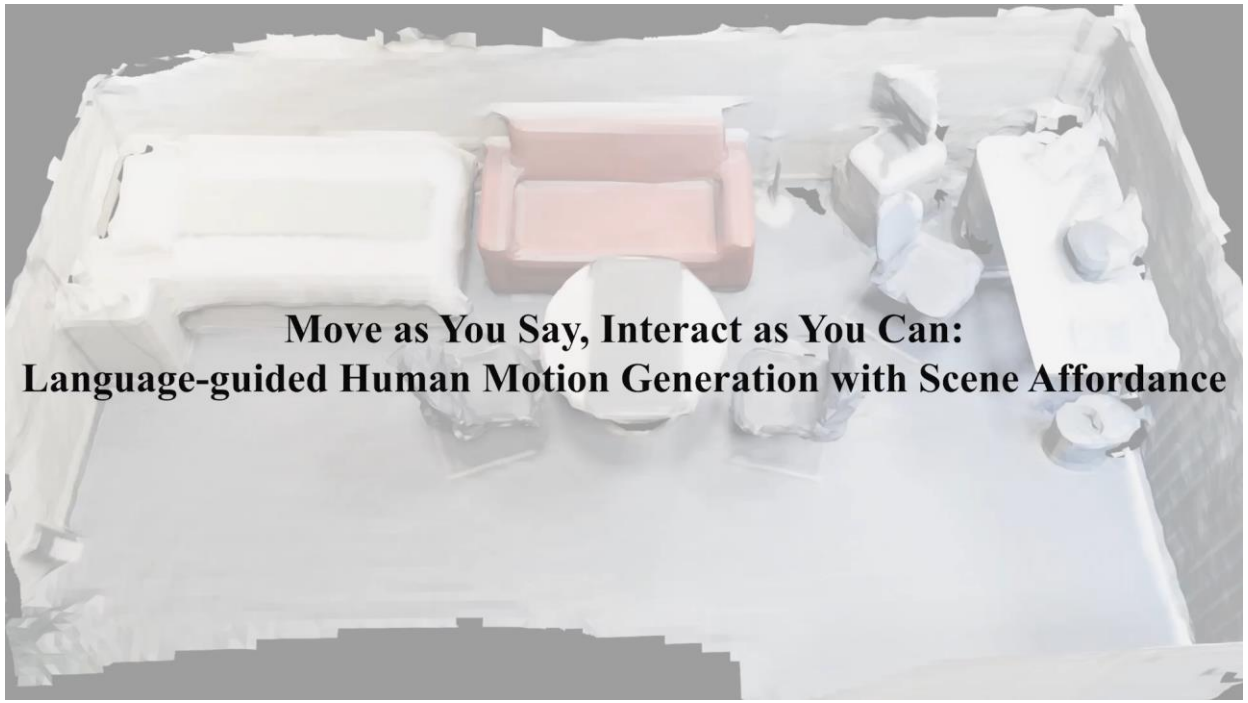
Dream Machine –Luma AI

Mainstream Work for Diffusion Models

Human Motion Generation

Human Expression Generation

- Diffusion Avatars: Deferred Diffusion for High-fidelity 3D Head Avatars, CVPR, 2024.

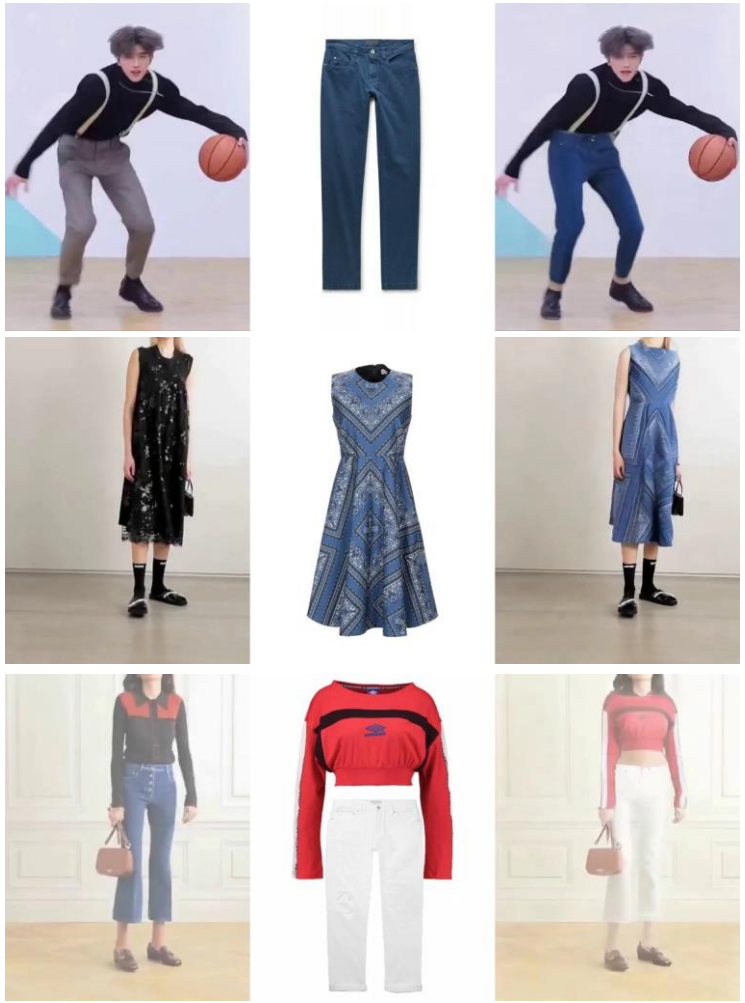


- Move as You Say, Interact as You Can: Language-guided Human Motion Generation with Scene Affordance, CVPR, 2024. Language-guided Human Motion Generation

- MotionEditor: Editing Video Motion via Content-Aware Diffusion, CVPR, 2024. Motion Editor

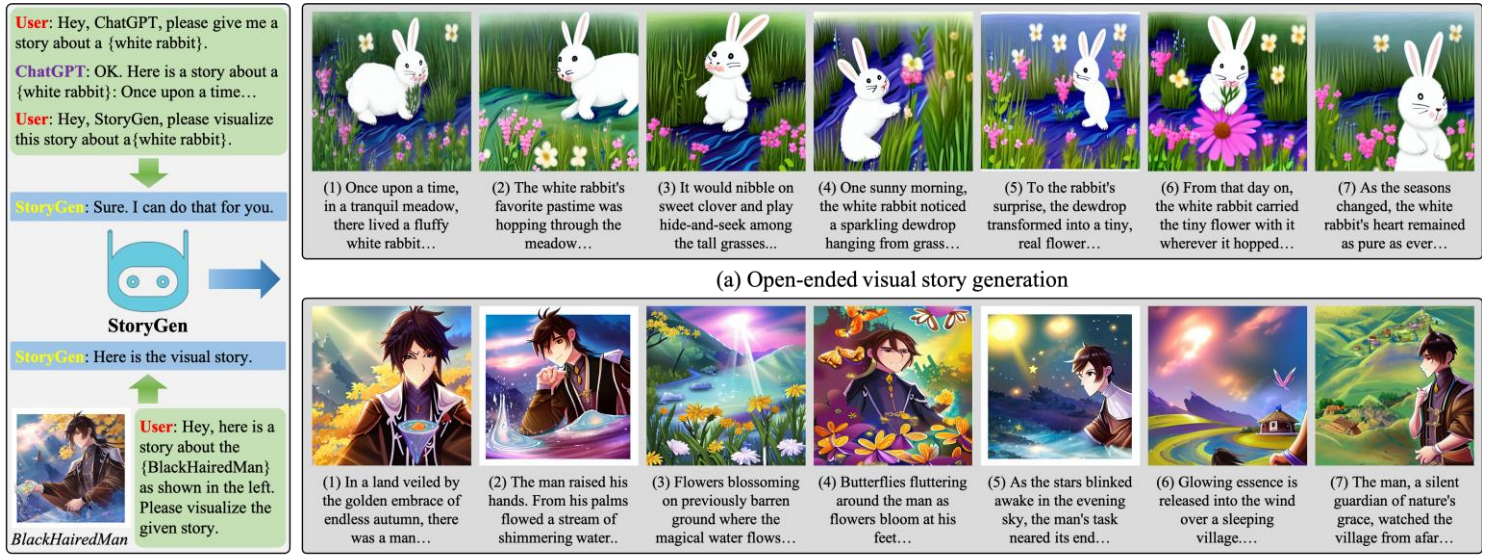
Mainstream Work for Diffusion Models

Human Motion Generation



- ViViD: Video Virtual Try-on using Diffusion Models, Arxiv, 2024. Virtual Try-on

- Diffusion4D: Fast Spatial-temporal Consistent 4D Generation via Video Diffusion Models, Arix, 2024. 4D Generation



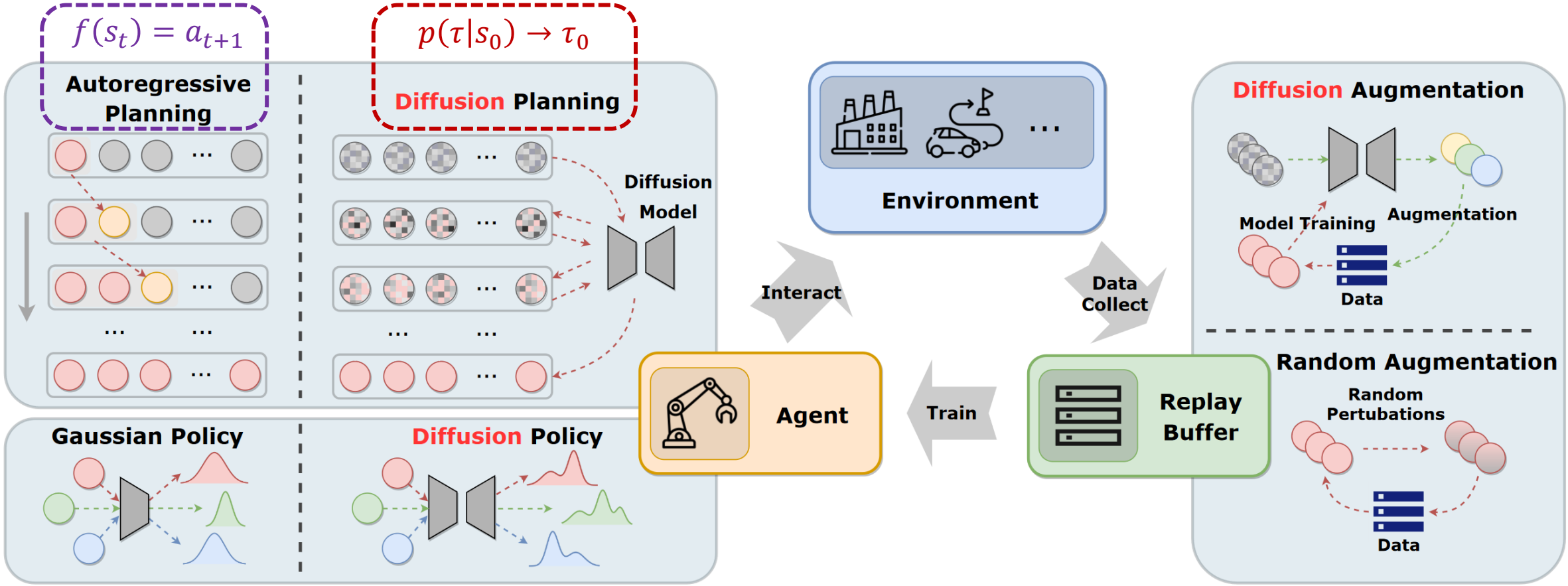
- StoryGen Intelligent Grimm - Open-ended Visual Storytelling via Latent Diffusion Models, CVPR, 2024. Story Generation

Diffusion Model in Robotics

Diffusion Model in Robotics

Policy Learning & Imitation Learning

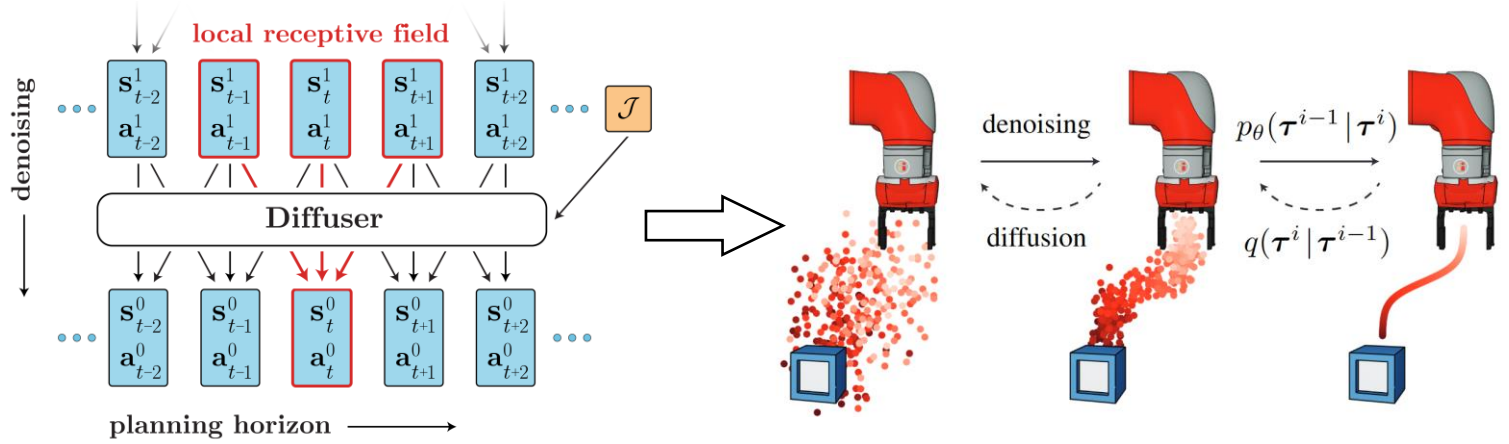
- Diffusion Models for Reinforcement Learning: A Survey, Arxiv, 2023.



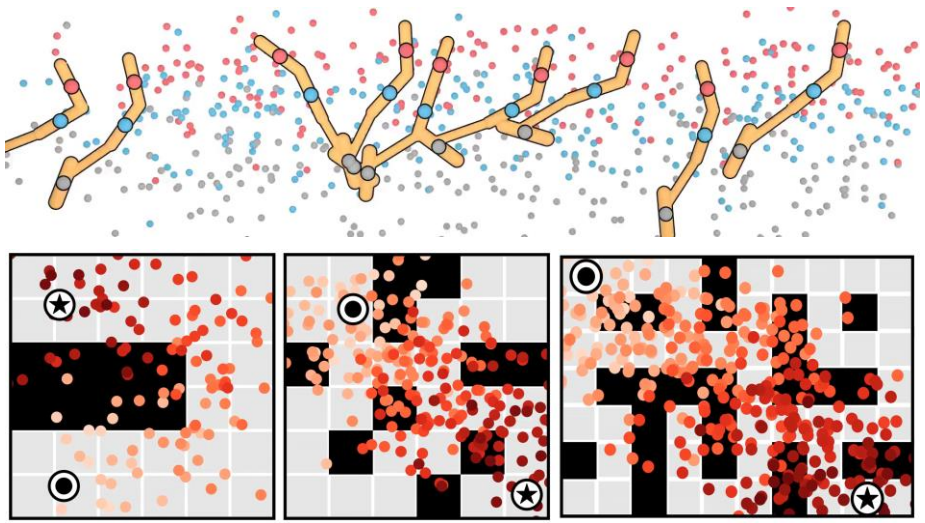
Diffusion Model in Robotics

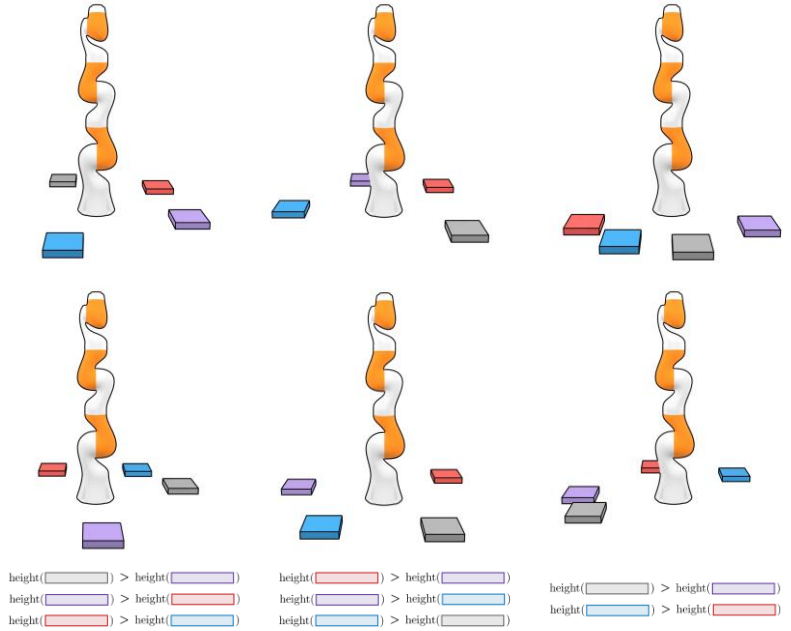
☐ Policy Learning & Imitation Learning

- Planning with Diffusion for Flexible Behavior Synthesis, ICML, 2022.



Diffuser is a denoising diffusion probabilistic model that plans by iteratively refining randomly sampled noise. The denoising process lends itself to flexible conditioning, by either using gradients of an objective function to bias plans toward high-reward regions or conditioning the plan to reach a specified goal.





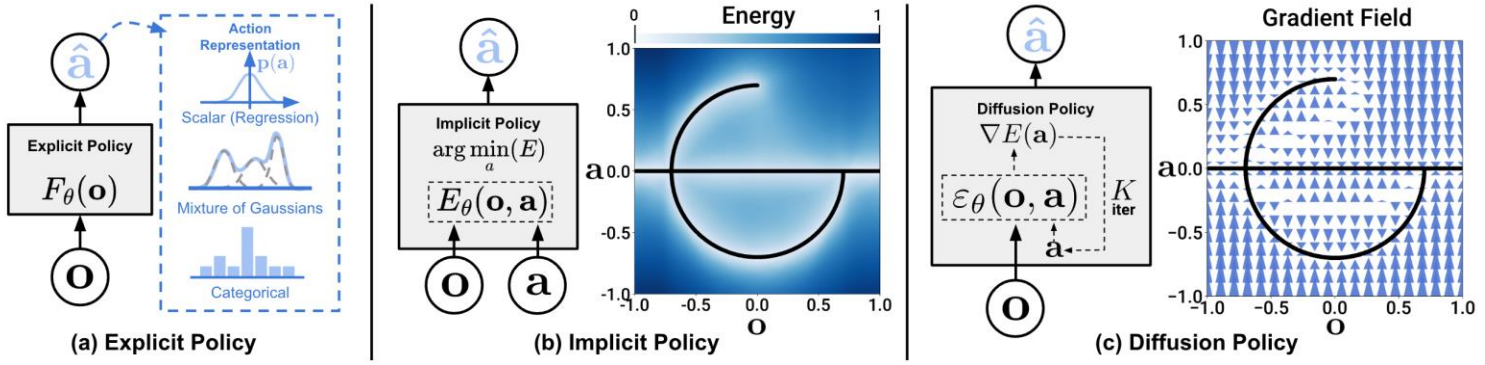
Unconditional stacking: Maximize the height of a block tower, with no further constraints.

Conditional stacking: Stack towers subject to test-time constraints.

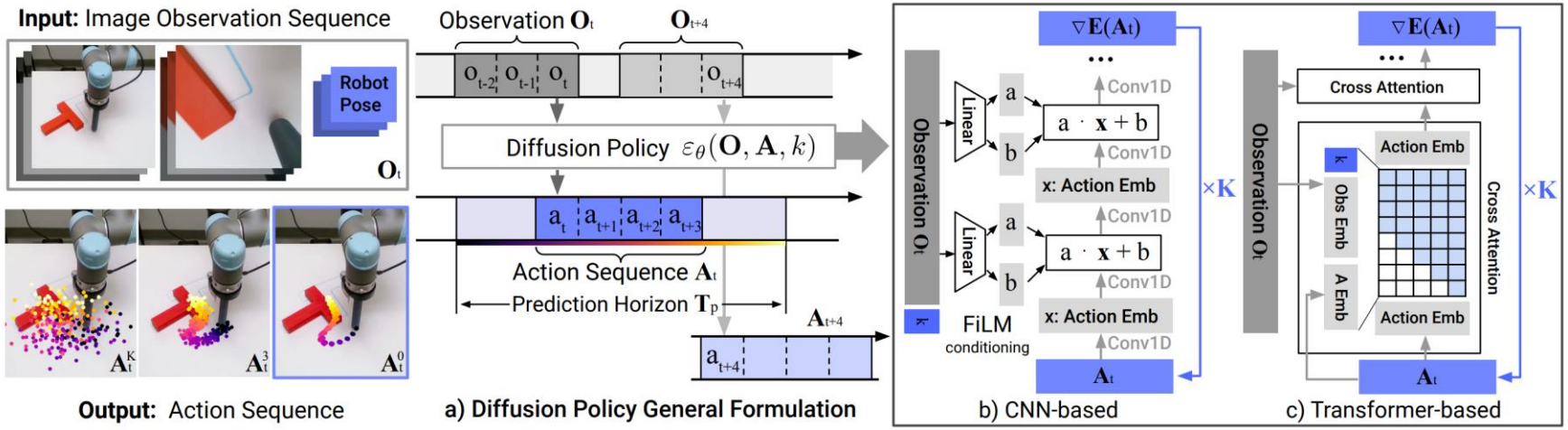
Diffusion Model in Robotics

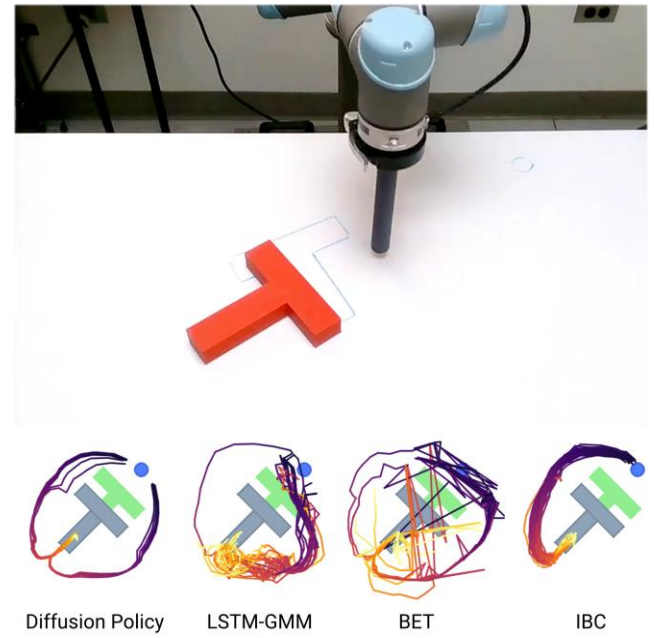
☐ Policy Learning & Imitation Learning

- Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, RSS 2023, IJRR 2024.



Policy Representations. a) Explicit policy with different types of action representations. b) Implicit policy learns an energy function conditioned on both action and observation and optimizes for actions that minimize the energy landscape c) Diffusion policy refines noise into actions via a learned gradient field.





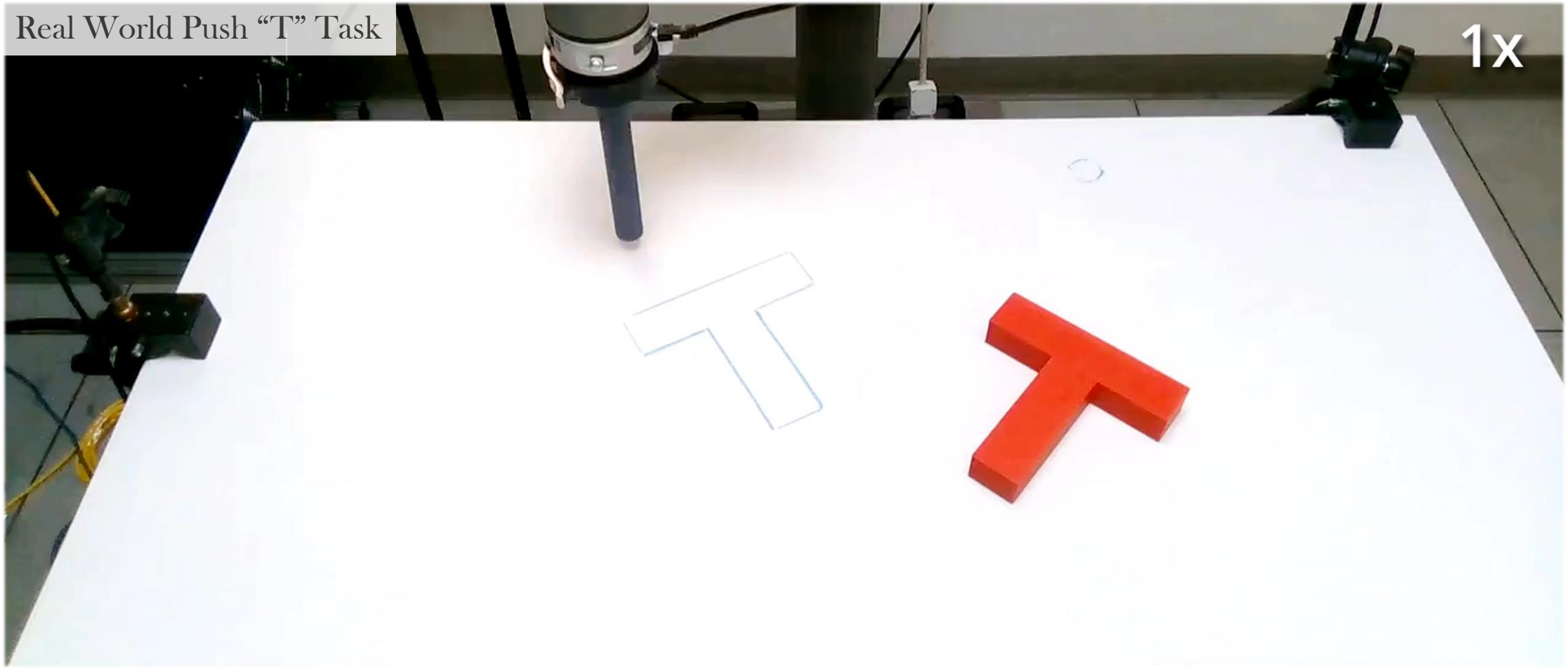
$p_\theta(\mathbf{A}|\mathbf{O})$: learning observation-conditioned action generation

Diffusion Policy (columbia.edu)

Diffusion Model in Robotics

☐ Policy Learning & Imitation Learning

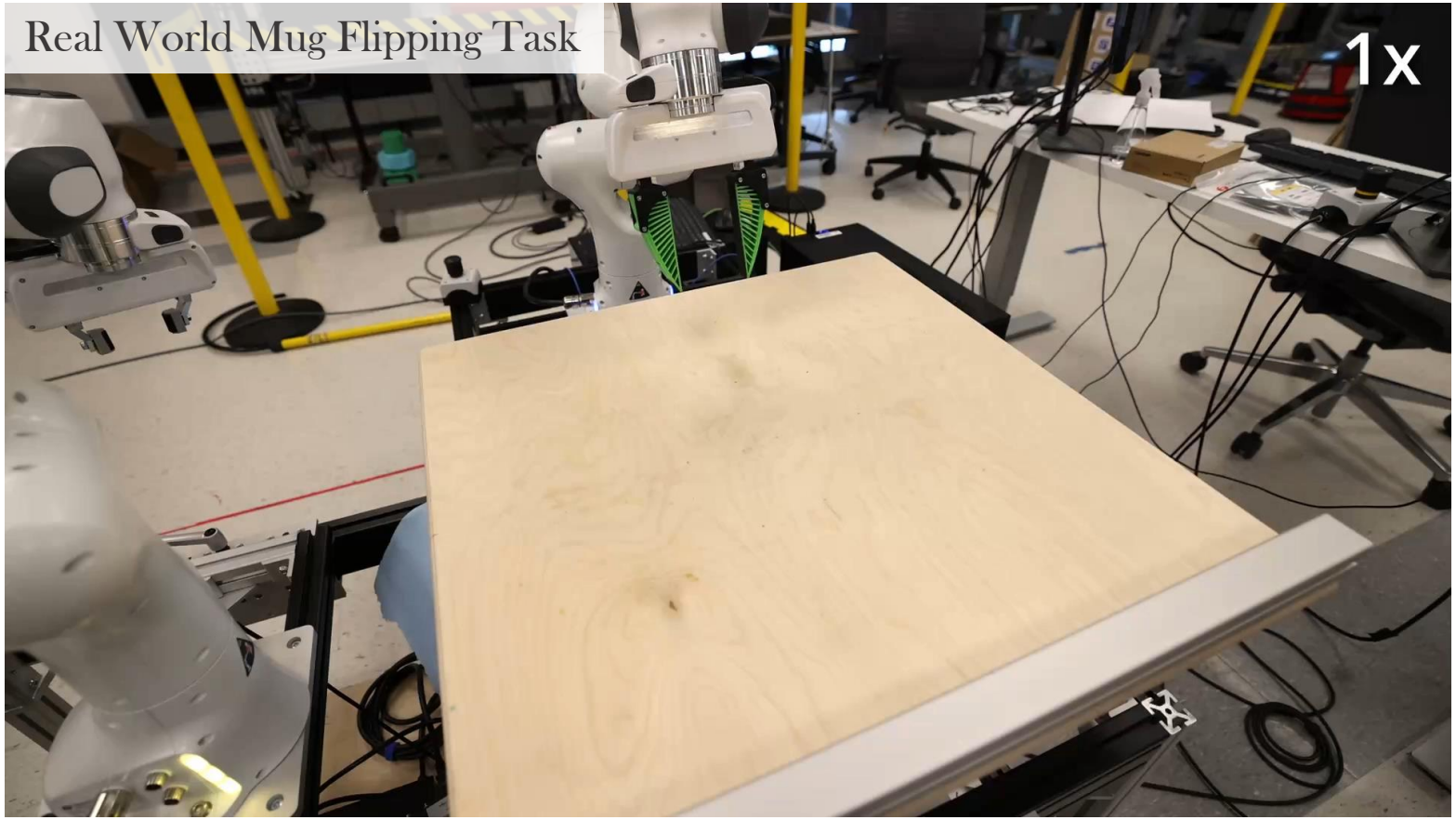
- Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, RSS 2023, IJRR 2024.
  (0:05) Occlusion caused by waiving hand in front of the camera.
  (0:11) Perturbation during pushing stage.
  (0:39) Perturbation during finishing stage.

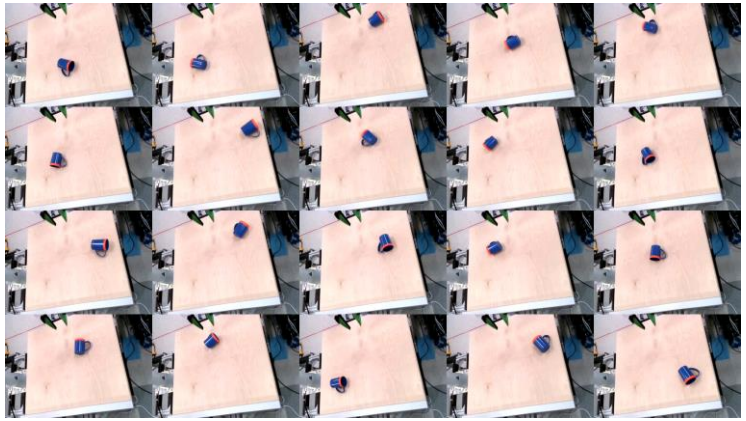


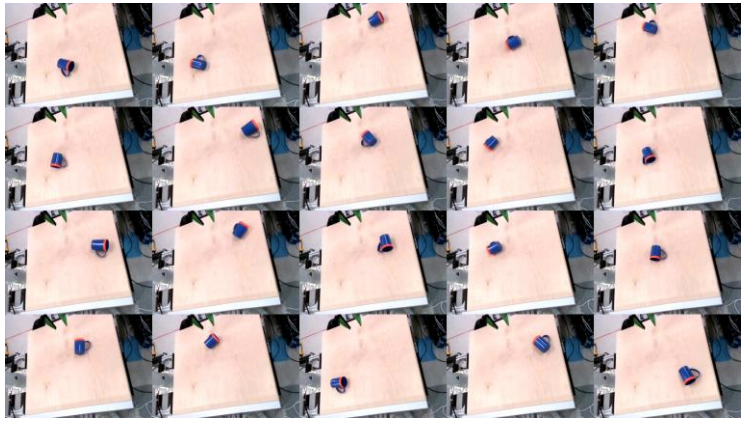
Diffusion Model in Robotics

☐ Policy Learning & Imitation Learning

- Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, RSS 2023, IJRR 2024.
  1. Pickup a randomly placed mug and place it lip down (marked orange).
  2. Rotate the mug such that its handle is pointing left.



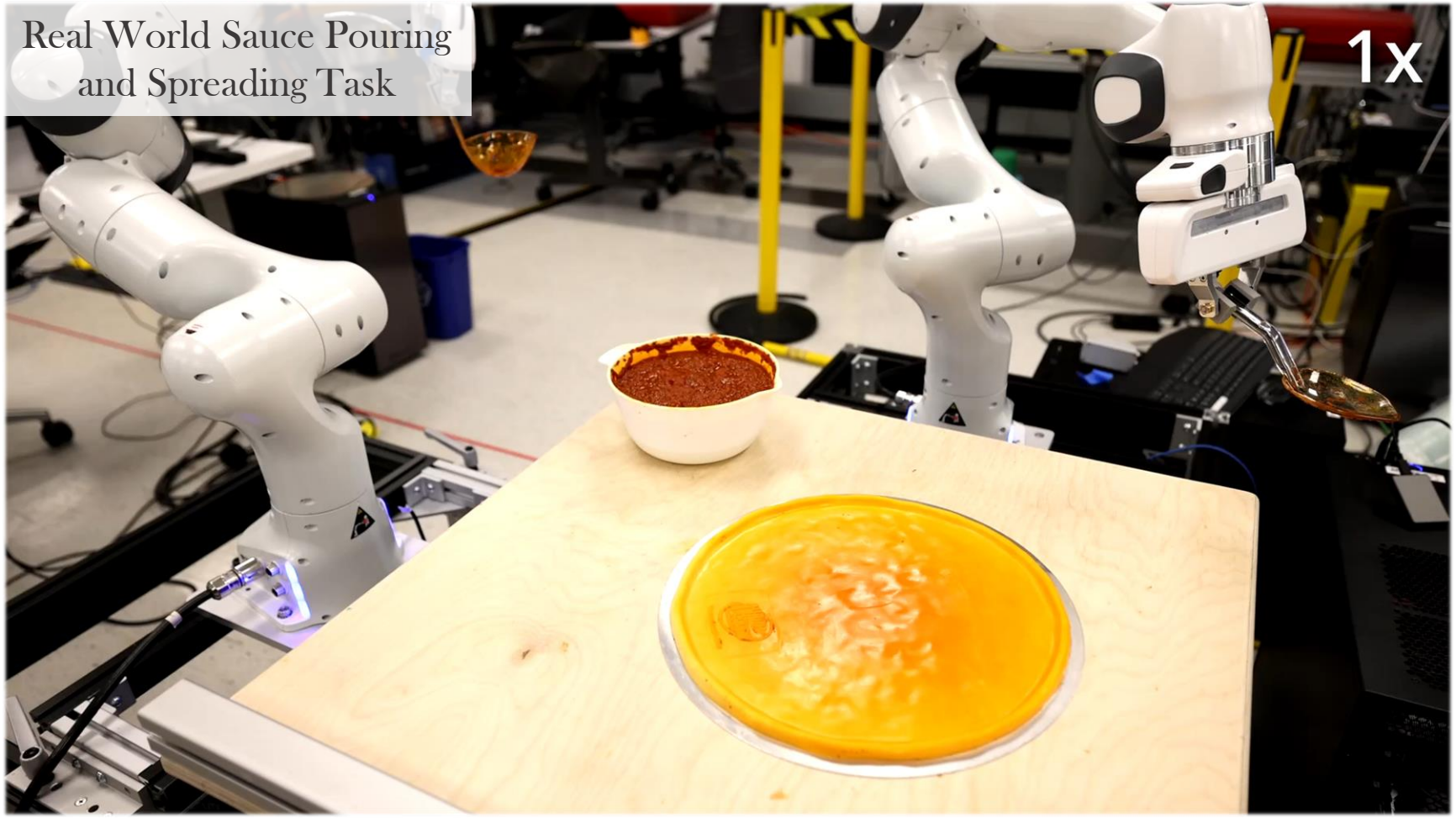




Diffusion Model in Robotics

Policy Learning & Imitation Learning

- Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, RSS 2023, IJRR 2024.



In the sauce pouring task, the robot needs to: 1) Dip the ladle to scoop sauce from the bowl, 2) approach the center of the pizza dough, 3) pour sauce, and 4) lift the ladle to finish the task.

In the sauce spreading task, the robot needs to: 1) Approach the center of the sauce with a grasped spoon, 2) spread the sauce to cover pizza in a spiral pattern, and 3) lift the spoon to finish the task.

Diffusion Model in Robotics

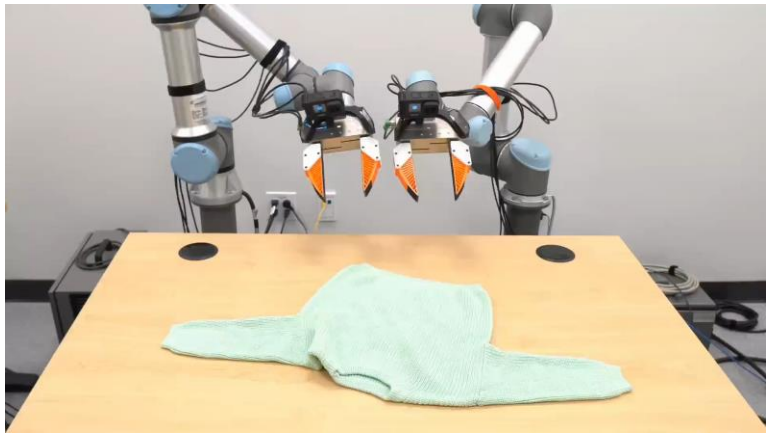
Policy Learning & Imitation Learning

- Universal Manipulation Interface: In-The-Wild Robot Teaching Without In-The-Wild Robots, RSS 2024.

Not a diffusion model, but an interesting work.

Human Demonstration with UMI

Fully Autonomous Policy Rollout

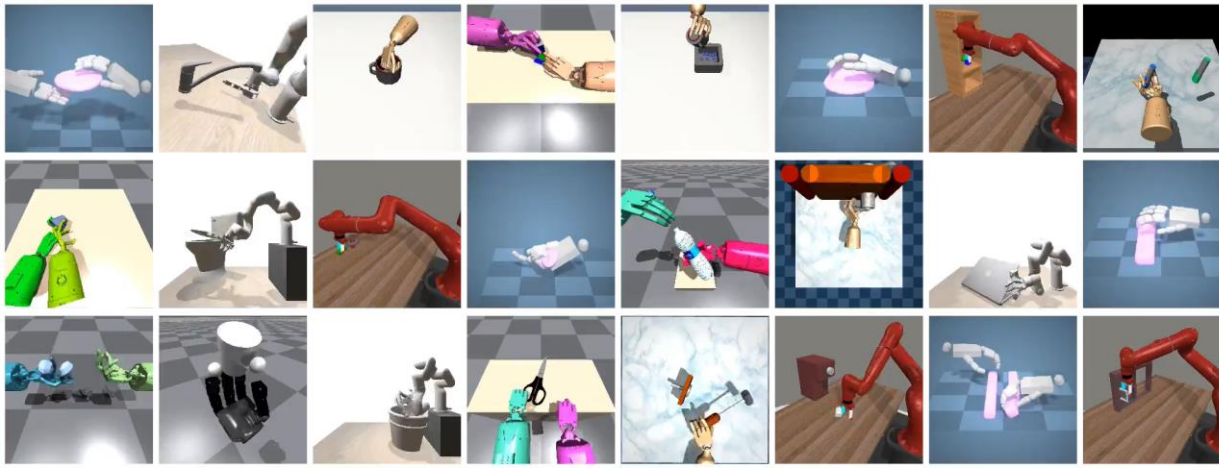


Universal Manipulation Interface (umi-gripper.github.io)

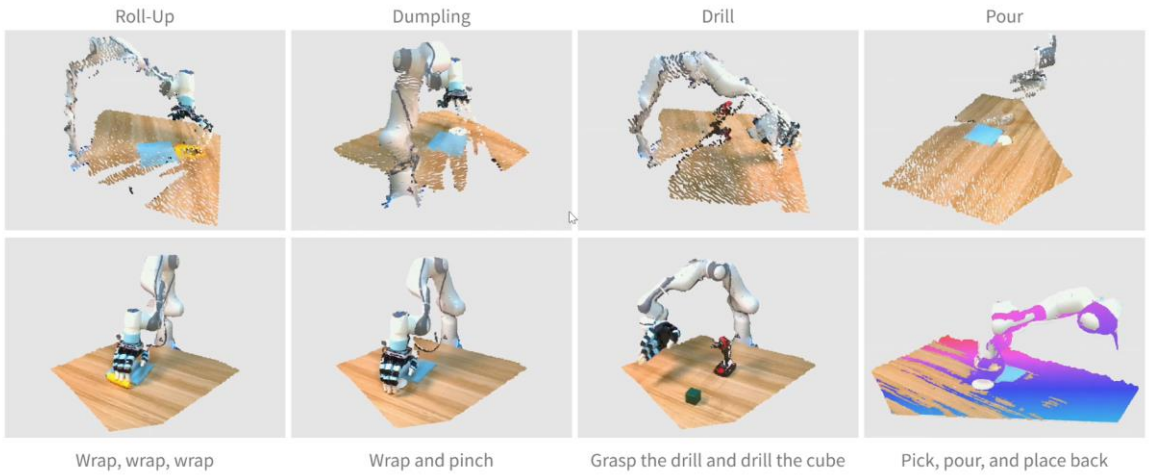
Diffusion Model in Robotics

Policy Learning & Imitation Learning

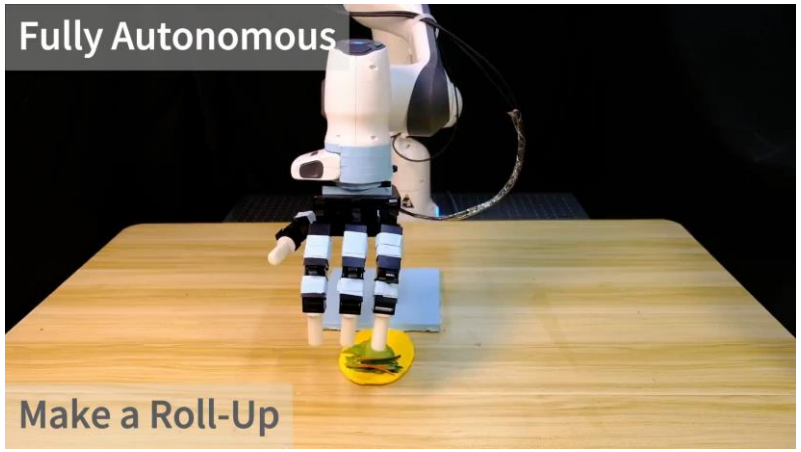
- 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations, RSS, 2024

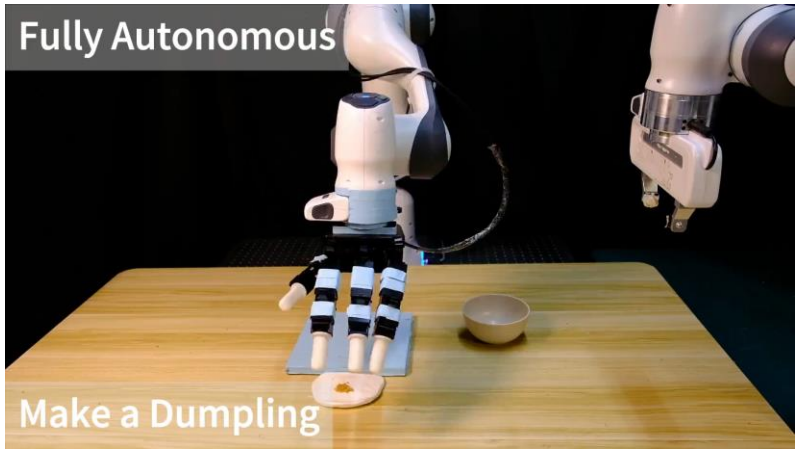


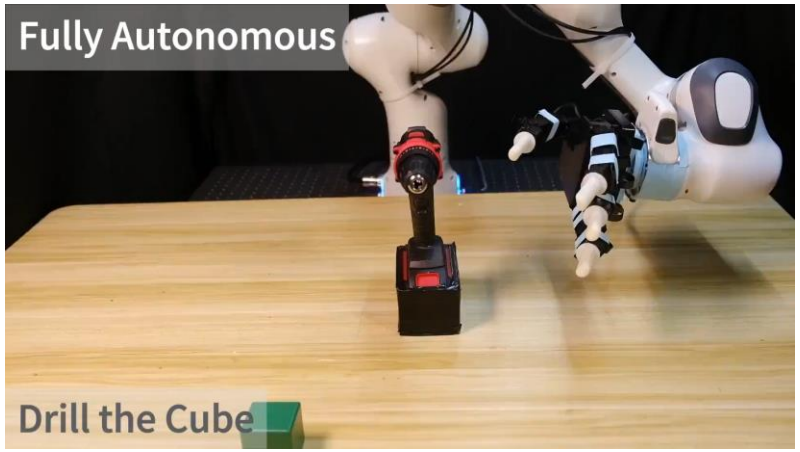
72 simulated tasks from 7 benchmarks



4 real-world tasks



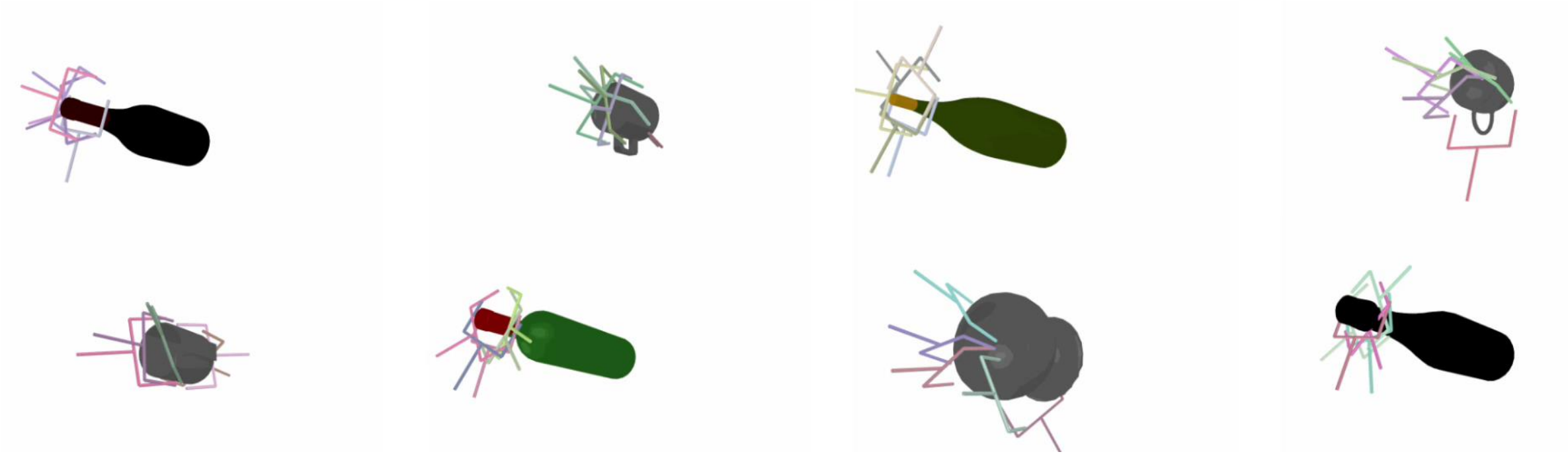




Diffusion Model in Robotics

Objective Learning

- SE(3)-DiffusionFields: Learning smooth cost functions for joint grasp and motion optimization through diffusion, ICRA, 2023.
  - ✓ Grasping Poses Generation: learning 3D-based grasping poses generators to approximate end-effector goals;
  - ✓ Grasping Energy Computation: learning smooth objective functions for robot trajectory optimization;
  - ✓ SE(3)-Diffusion Formulation: theoretical supporting for learning data distribution in SE(3) space.

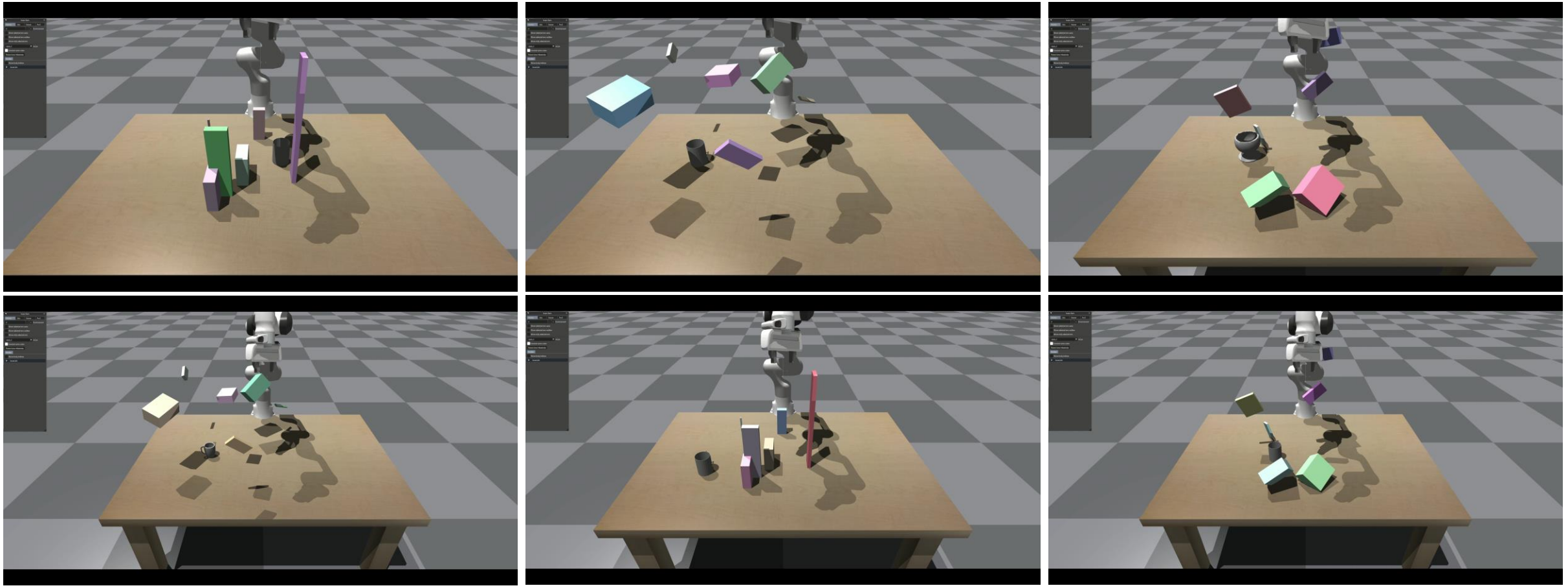


$$\mathcal{L}_{\text{dsm}} = \frac{1}{L}\sum_{k=0}^{L}\mathbb{E}_{\boldsymbol{H},\hat{\boldsymbol{H}}}\left[\left\|\boldsymbol{s}_{\boldsymbol{\theta}}(\hat{\boldsymbol{H}},k) - \frac{D\log q(\hat{\boldsymbol{H}}|\boldsymbol{H},\sigma_k\boldsymbol{I})}{D\hat{\boldsymbol{H}}}\right\|\right] \qquad \boldsymbol{H}_{k-1} = \text{Expmap}\left(\frac{\alpha_k^2}{2}\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{H}_k,k) + \alpha_k\boldsymbol{\epsilon}\right)\boldsymbol{H}_k$$

Diffusion Model in Robotics

☐ Objective Learning

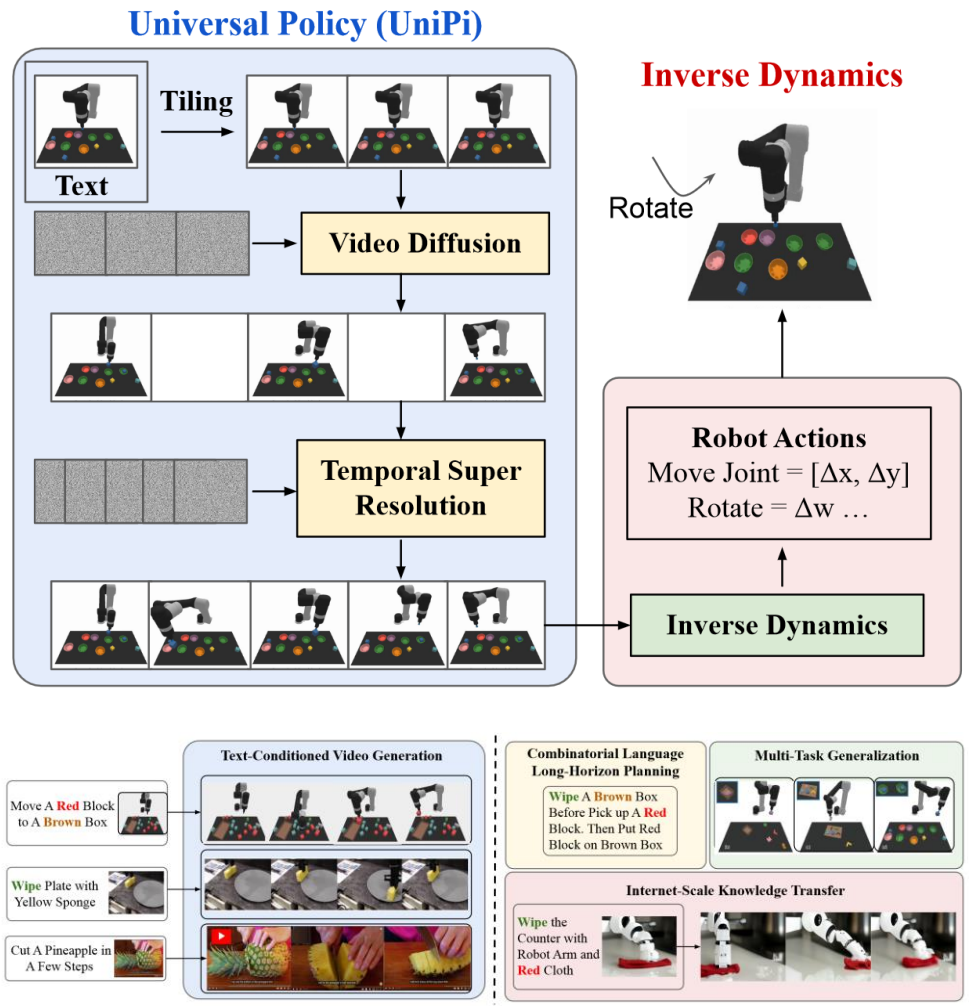
- SE(3)-DiffusionFields: Learning smooth cost functions for joint grasp and motion optimization through diffusion, ICRA, 2023.
  - ✓ Grasping Poses Generation: learning task-specific generators to guide optimization by providing approximate end-effector goals;
  - ✓ **Grasping Energy Computation: learning smooth objective functions for robot trajectory optimization;**
  - ✓ SE(3)-Diffusion Formulation: theoretical supporting for learning data distribution in SE(3) space.

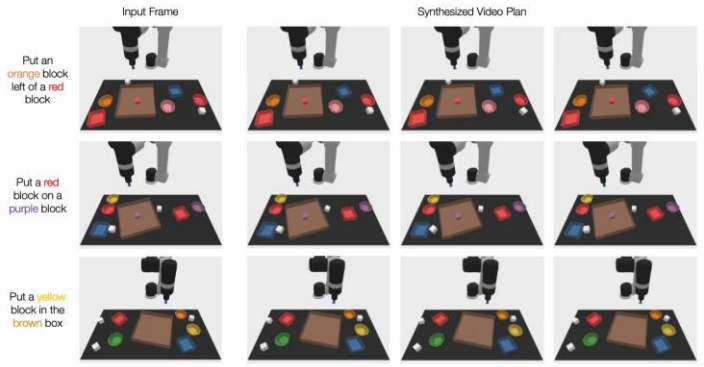


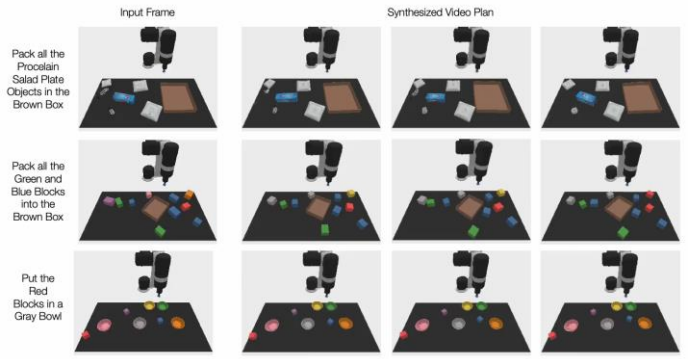
Diffusion Model in Robotics

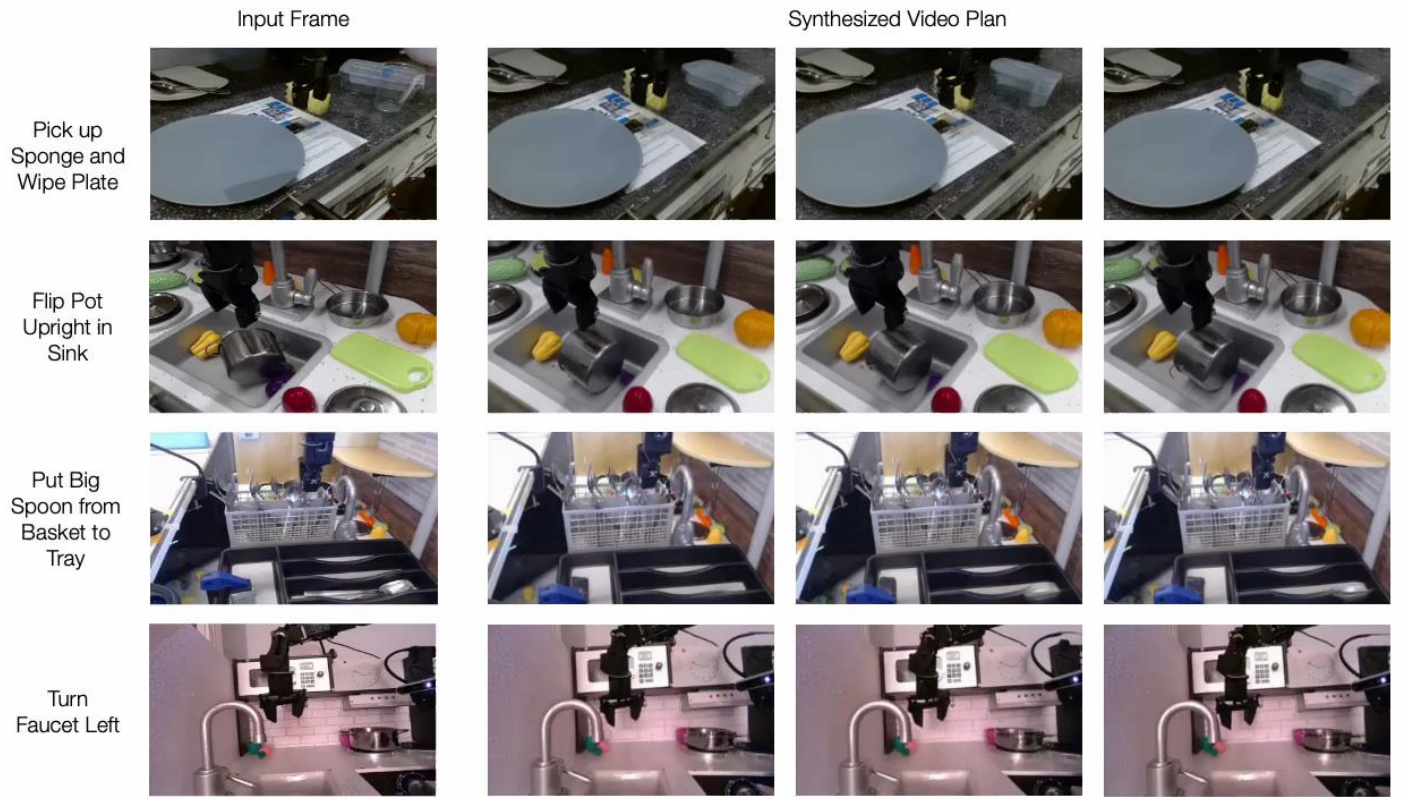
Video Learning

- Learning Universal Policies via Text-Guided Video Generation, NeurIPS, 2023. **policy-as-video**





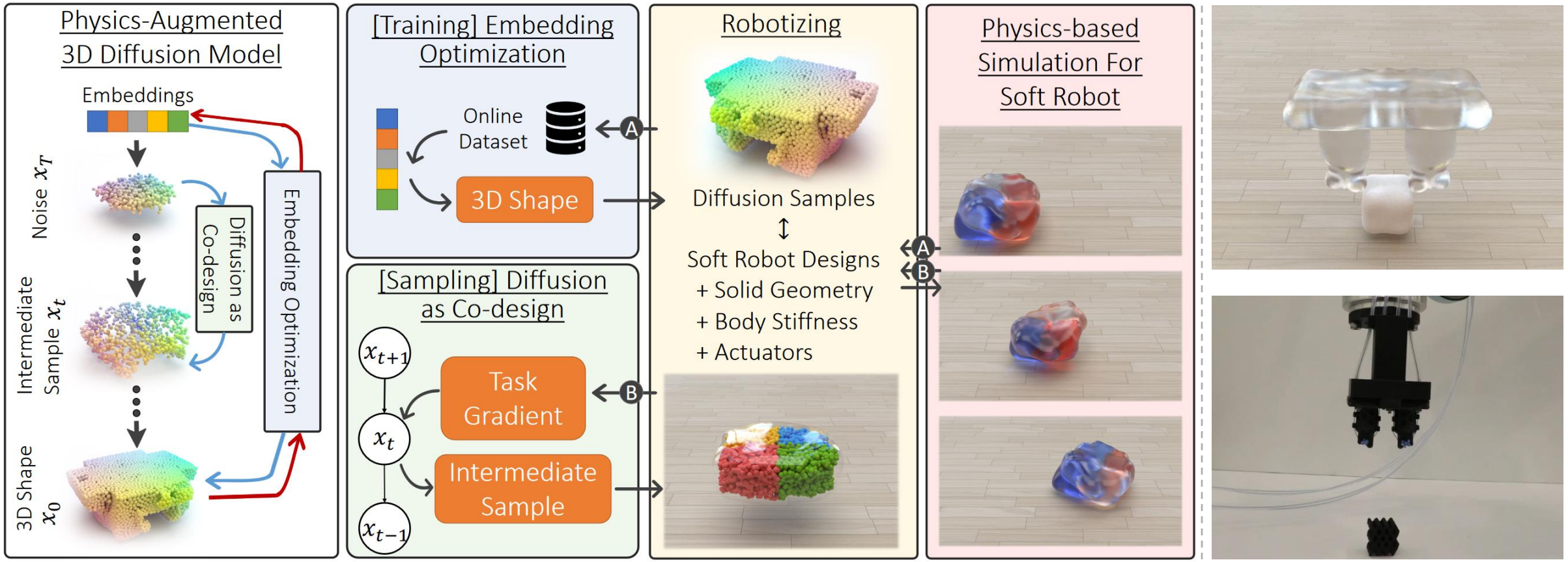




Diffusion Model in Robotics

☐ Mechanical Design

- DiffuseBot: Breeding Soft Robots With Physics-Augmented Generative Diffusion Models, NeurIPS, 2023.



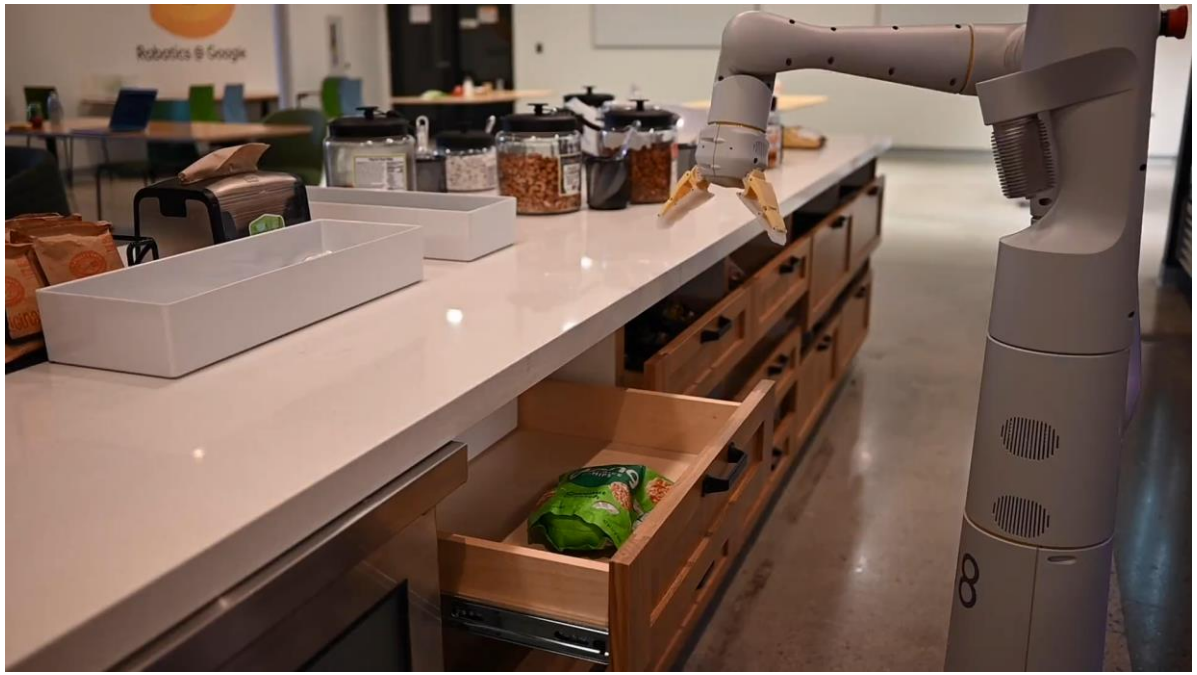
DiffuseBot: Breeding Soft Robots With Physics-Augmented Generative Diffusion Models

Diffusion Model in Robotics

Data Augmentation

- Scaling Robot Learning with Semantically Imagined Experience, RSS, 2023.

Google propose using text-guided diffusion models for data augmentation for robot learning. These augmentations can produce photorealistic images for learning downstream tasks such as manipulation.



ROSIE can also pinpoint the augmentation to a small region of the image, as is shown in the video, where we change the object in the drawer. Furthermore, we are able to augment in-hand objects, as is shown in the last part of this video.

Scaling Robot Learning with Semantically Imagined Experience (diffusion-rosie.github.io)

Thanks

Sixu Yan