Advanced Generative Models: From VAE to Diffusion and Flow Matching

Networked Intelligence for Comprehensive Efficiency (NICE)
College of Information Science and Electronic Engineering
Zhejiang University
Teams: http://nice.rongpeng.info/

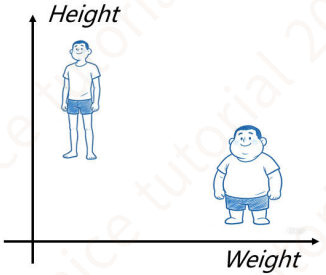
Jan. 18, 2026

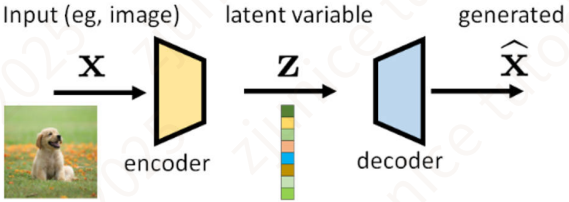
Table of Contents

Basics of VAE

Autoencoder (AE) Introduction

- Autoencoder (AE)
  - Learns to represent data with fewer features
  - Encoder: high-dimensional → low-dimensional latent space
  - Decoder: latent space → high-dimensional reconstruction

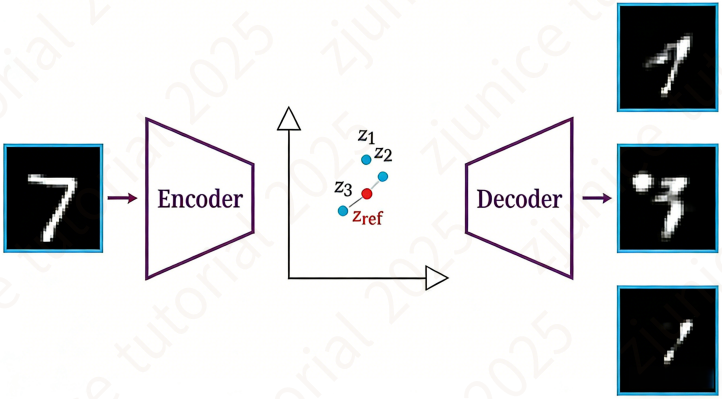




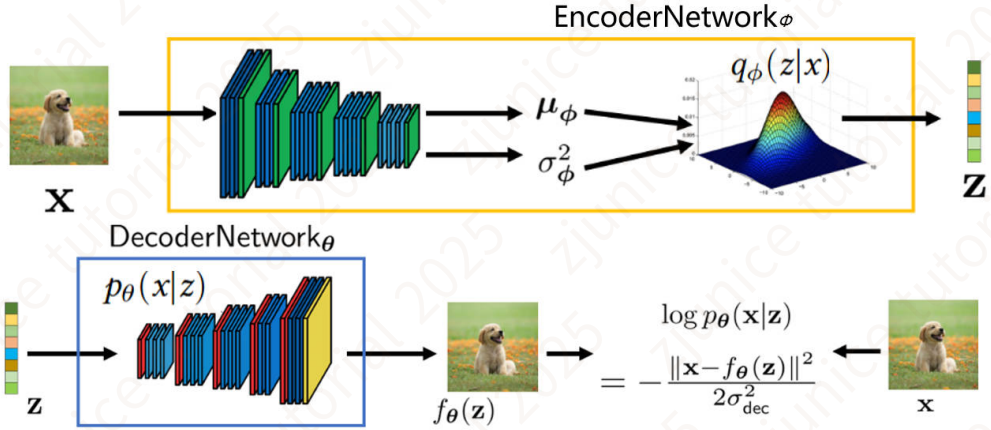
[1] Chan S H. Tutorial on Diffusion Models for Imaging and Vision[A/OL]. arXiv, 2025.

Limitations of AE

- Lacks generative capability
  Latent space is “discontinuous”: points between training data encodings map to meaningless/blurry outputs.



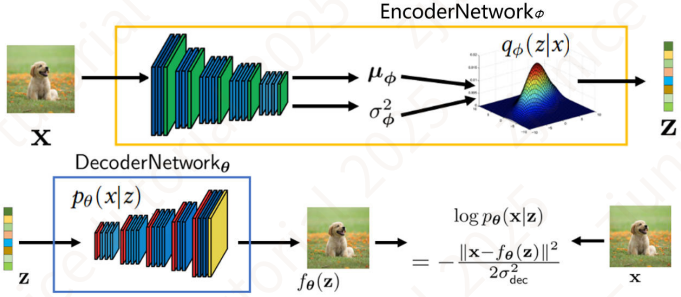
What is VAE?



[1] Chan S H. Tutorial on Diffusion Models for Imaging and Vision[A/OL]. arXiv, 2025.

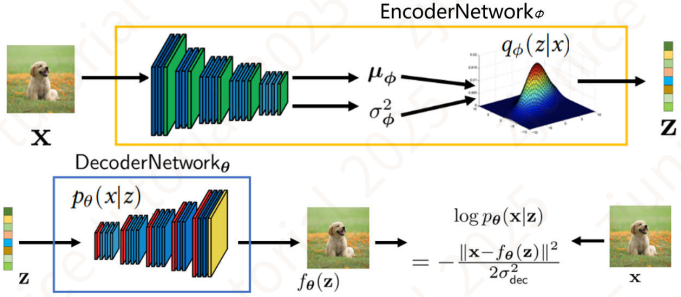
The distributions of VAE

- Key distributions:
  - $P_{data}(x)$: True data distribution (unknown).
  - $P(x)$: Evidence, probability of x under the model.
  - $P(z)$: Prior over latent z, chosen as standard Gaussian $\mathcal{N}(0,1)$ in VAE.
  - $P(z|x)$: True posterior (intractable).
  - $q_\phi(z|x)$: Approximate posterior (encoder output), Gaussian in VAE.
  - $p(x|z)$: True likelihood (intractable).
  - $p_\theta(x|z)$: Likelihood model (decoder output), often Gaussian.

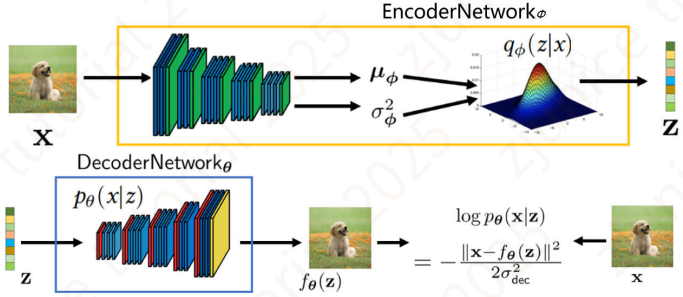


The distributions of VAE

- Key distributions:
  - $P_{data}(x)$: True data distribution (unknown).
  - $P(x)$: Evidence, probability of x under the model.
  - $P(z)$: Prior over latent z, chosen as standard Gaussian $\mathcal{N}(0, 1)$ in VAE.
  - $P(z|x)$: True posterior (intractable).
  - $q_\phi(z|x)$: Approximate posterior (encoder output), Gaussian in VAE.
  - $p(x|z)$: True likelihood (intractable).
  - $p_\theta(x|z)$: Likelihood model (decoder output), often Gaussian.



- Why Gaussian?
  - Why set $p(z)$ as standard Gaussian?
    To encourage a **continuous and complete** latent space, VAEs use a suitable loss (ELBO) to ensure both reconstruction quality and posterior regularity.
  - Why choose $q_\phi(z|x)$ as Gaussian?
    True posterior $p(z|x)$ is intractable. VAE uses **variational inference** to approximate it with a simple, tractable distribution – a multivariate Gaussian. Maximizing ELBO ensures approximation quality.



ELBO Derivation

Goal: Maximize data likelihood $\log p(x)$.

$$
\begin{aligned}
D(p_{data}(x)||p(x)) &= \int p_{data}(x) \log \frac{p_{data}(x)}{p(x)} dx \\
&= \underbrace{E_{p_{data}(x)}(\log p_{data}(x))}_{constant} - \underbrace{E_{P_{data}(x)}(\log p(x))}_{log\ likehood}
\end{aligned}
$$

Minimize KL divergence$\Rightarrow$ **Maximize** $E_{p_{data}(x)}[\log p(x)]$

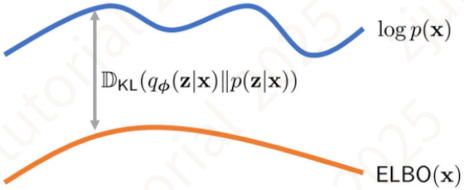
ELBO from $\log p(x)$

$$
\begin{aligned}
\log p(x) &= \log p(x) \times \underbrace{\int q_\phi(z|x)\, dz}_{=1} \\
&= \int \underbrace{\log p(x)}_{\text{some constant wrt } z} \times \underbrace{q_\phi(z|x)}_{\text{distribution in } z}\, dz \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{p(\mathbf{z}|\mathbf{x})}\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{p(\mathbf{z}|\mathbf{x})} \times \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \\
&= \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right]}_{\text{ELBO}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\right]}_{\mathbb{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}|\mathbf{x}))}
\end{aligned}
$$

$$\log p(x) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right]}_{\text{ELBO}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\right]}_{\mathbb{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}|\mathbf{x}))}$$

Since KL divergence ≥ 0, maximizing ELBO has two effects:

- Indirectly **maximizes the log-likelihood** by raising its lower bound.
- **Minimizes KL divergence** between $q(z|x)$ and $p(z|x)$, making the encoder approximate the true posterior.



$$
\begin{aligned}
\mathrm{ELBO}(\mathbf{x}) &\overset{\text{def}}{=} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] && \text{(definition)} \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] && (p(\mathbf{x},\mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})) \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] && \text{(split expectation)} \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathbb{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})) && \text{(definition of KL)}
\end{aligned}
$$

Two components of ELBO

$$
\mathrm{ELBO}(\mathbf{x}) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\overbrace{\log p_\theta(\mathbf{x}|\mathbf{z})}^{\text{a Gaussian}}]}_{\text{how good your decoder is}} - \underbrace{\mathbb{D}_{\mathrm{KL}}(\overbrace{q_\phi(\mathbf{z}|\mathbf{x})}^{\text{a Gaussian}} \| \overbrace{p(\mathbf{z})}^{\text{a Gaussian}})}_{\text{how good your encoder is}}
$$

- Two components of ELBO
  - Reconstruction term ensures the decoded output matches the original input.
  - KL divergence term regularizes the latent space to follow a prior distribution.

- VAE Training

$$\mathrm{ELBO}_{\phi,\theta}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathbb{D}_{\mathrm{KL}}\left(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})\right).$$

Here, the **reconstruction term** is:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \approx -\frac{1}{M}\sum_{m=1}^{M} \frac{\|\mathbf{x} - f_\theta(\mu_\phi(\mathbf{x}) + \sigma_\phi(\mathbf{x})\epsilon^{(m)})\|^2}{2\sigma_{\mathrm{dec}}^2},$$

whereas the **prior matching term** is

$$\mathbb{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})) = \frac{1}{2}\left(\sigma_\phi^2(\mathbf{x})d - d + \|\mu_\phi(\mathbf{x})\|^2 - 2\log\sigma_\phi(\mathbf{x})\right).$$

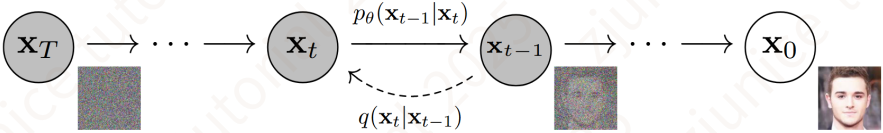
[1] Chan S H. Tutorial on Diffusion Models for Imaging and Vision[A/OL]. arXiv, 2025.

Advanced Diffusion Models: Image and Video Generation

Intuitive Understanding: From Signal to Noise

The Physical Intuition: Diffusion models mimic the process of thermodynamics but in reverse.

- **Forward (q)**: Like a drop of ink diffusing into water. We slowly destroy the data structure until it becomes pure random noise.
- **Reverse (p)**: Like rewinding the video. We learn to remove the noise step-by-step to recover the original structure.



Ho J, Jain A, Abbeel P, "Denoising diffusion probabilistic models,"in Advances in neural information processing systems, 33: 6840-6851, 2020.

DDPM Fundamentals: The Forward Process

The Markov Chain: The forward process gradually adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$.

Transition Kernel (Distribution Form)

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Reparameterization Trick (Sampling): To sample $\mathbf{x}_t$, we inject standard Gaussian noise $\boldsymbol{\epsilon}$:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

Basic Notation:

- $\beta_t \in (0, 1)$: Variance of noise added at step t.
- $\alpha_t := 1 - \beta_t$: Signal retained at step t (Used in next slide).

Key Property: One-Step Sampling

From Recursive to Closed-Form: Using $\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1}$, let's expand one step further:

$$\begin{aligned}\mathbf{x}_t &= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_{t-2}) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\ &= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \underbrace{\sqrt{\alpha_t(1-\alpha_{t-1})}\boldsymbol{\epsilon}_{t-2} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1}}_{\text{Two Gaussians merge: } \mathcal{N}(0,\sigma_1^2+\sigma_2^2)}\end{aligned}$$

*Variance math: $\alpha_t(1-\alpha_{t-1}) + (1-\alpha_t) = 1 - \alpha_t\alpha_{t-1}$

Defining Cumulative Schedule: Let $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$. The merged noise variance becomes $1 - \bar{\alpha}_t$.

Marginal Distribution $q(\mathbf{x}_t|\mathbf{x}_0)$

We can now sample $\mathbf{x}_t$ directly from $\mathbf{x}_0$:

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

Step 1: The Perspective of Bayes' Theorem

Conditioned on $\mathbf{x}_0$, the posterior is tractable via Bayes' theorem:

$$\underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}_{\text{Posterior}} = \frac{\overbrace{q(\mathbf{x}_t|\mathbf{x}_{t-1})}^{\text{Forward Step}} \cdot \overbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}^{\text{Marginal } t-1}}{\underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Marginal } t}}$$

All three terms on the RHS are Known Gaussians:

- **1. Transition:** $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$
- **2. Marginal at $t-1$:** $q(\mathbf{x}_{t-1}|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0, (1-\bar{\alpha}_{t-1})\mathbf{I})$
- **3. Marginal at t:** $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$

Since Gaussians are closed under multiplication and division, the posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ must also be a Gaussian.

Step 2: Analytical Solution for the Posterior

Substituting the Gaussian definitions into Bayes' formula, we derive the closed-form distribution:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I})$$

The mean $\tilde{\boldsymbol{\mu}}_t$ is a linear combination of $\mathbf{x}_t$ (current noisy image) and $\mathbf{x}_0$ (original image):

The Posterior Mean and Variance

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$$

$$\tilde{\boldsymbol{\beta}}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

The Problem: In the generative process (inference), we have $\mathbf{x}_t$, but we **do not know** $\mathbf{x}_0$.
$\Rightarrow$ We need to approximate $\mathbf{x}_0 = f_\theta(\mathbf{x}_t, t)$.

Step 3: Learning to Predict $\mathbf{x}_0$ (or Noise)

Since $\mathbf{x}_0$ is unknown, we train a neural network to predict it based on $\mathbf{x}_t$.

1. **Recall relationship:** $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \implies \mathbf{x}_0 = \frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}{\sqrt{\bar{\alpha}_t}}$
2. **Neural Network:** Instead of predicting $\mathbf{x}_0$ directly, we usually predict the noise $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$.
3. **Substitution:** Replace true $\boldsymbol{\epsilon}$ with predicted $\boldsymbol{\epsilon}_\theta$ to estimate $\mathbf{x}_0$, then compute the mean.

The Learned Reverse Process $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right), \beta_t\mathbf{I}\right)$$

This allows us to sample $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$ iteratively.

ELBO Derivation (1/2): Expansion

Step 1: Variable Substitution (VAE → DDPM) In standard VAE, latent is $\mathbf{z}$. In DDPM, latent is trajectory $\mathbf{x}_{1:T}$.

$$\text{VAE: } \mathbf{z} \to \mathbf{x}_{1:T}, \quad \text{Data: } \mathbf{x} \to \mathbf{x}_0$$

The ELBO objective transforms accordingly:

$$\log p(\mathbf{x}_0) \geq \mathbb{E}_q \left[\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right]$$

Step 2: The Markov Property Expansion
Applying chain rule and Markov property:

$$\textbf{Forward: } q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \qquad \textbf{Reverse: } p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

Step 3: Resulting Form (Plugging products back into ELBO):

Full Expansion Result

$$\text{ELBO} = \mathbb{E}_q \left[\log p(\mathbf{x}_T) + \sum_{t=1}^{T} \log \frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} \right]$$

ELBO Derivation (2/2): Decomposition

Step 4: Regrouping Terms

By rewriting the summation and using Bayes' rule on $q\left(q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\right)$, we can separate the terms by time step t:

$$\text{ELBO} = \mathbb{E}_q\left[\log p_\theta(\mathbf{x}_0|\mathbf{x}_1) - D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T)) - \sum_{t=2}^{T}\log\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\right]$$

Step 5: Final Interpretation

The objective naturally splits into three interpretable components:

The Decomposed Objective

$$\mathcal{L}_{\text{VLB}} = \underbrace{\mathbb{E}_q[\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)]}_{L_0:\text{Reconstruction}} - \underbrace{D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T))}_{L_T:\text{Constant Prior}} - \sum_{t=2}^{T}\underbrace{D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}:\text{Denoising Matching}}$$

Insight: L_{t-1} is the core term, forcing the model p_θ to match the true posterior of the forward process.

Simplifying the Loss: KL between Gaussians

Key Insight: Both the posterior q and our model p_θ are **Gaussians**.

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I})$$
$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \tilde{\beta}_t \mathbf{I})$$

Simplification: KL $\rightarrow$ MSE

The KL divergence between two Gaussians with the same variance is simply the **Mean Squared Error (MSE)** between their means:

$$L_{t-1} \propto \mathbb{E}_q \left[\| \underbrace{\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0)}_{\text{Target Mean}} - \underbrace{\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)}_{\text{Predicted Mean}} \|^2 \right]$$

Implication: To minimize the ELBO, the neural network $\boldsymbol{\mu}_\theta$ just needs to predict the **posterior mean** $\tilde{\boldsymbol{\mu}}_t$.

Deriving the Noise Prediction Objective

1. Express Target $\tilde{\mu}_t$ via Noise ϵ

Using the reparameterization trick $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon})$, we can rewrite the posterior mean solely in terms of $\mathbf{x}_t$ and $\boldsymbol{\epsilon}$:

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \boldsymbol{\epsilon}) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\right)$$

2. Parameterize Model $\boldsymbol{\mu}_\theta$

To match the structure of $\tilde{\boldsymbol{\mu}}_t$, we parameterize the network output as:

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

3. Final Objective (Simplification)

Substituting both equations back into the MSE objective:

$$\|\tilde{\boldsymbol{\mu}}_t - \boldsymbol{\mu}_\theta\|^2 \propto \left\|\frac{\beta_t}{\dots}(\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta)\right\|^2 \propto \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2$$

We ignore the weighting coefficients for better training stability (Simple Loss).

Optimization & Algorithms

Training Objective

We optimize the variational lower bound by minimizing the simple mean-squared error between the true noise $\boldsymbol{\epsilon}$ and the predicted noise $\boldsymbol{\epsilon}_\theta$:

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

Algorithm 1: Training

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

Algorithm 2: Sampling

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

A Generalized Perspective

Revisiting the Reverse Step: In DDPM, the posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ was derived using Bayes' rule with a fixed Markovian forward process.

The Key Insight (Non-Markovian): The training objective only depends on marginals $q(\mathbf{x}_t|\mathbf{x}_0)$. We can derive a broader family of posteriors by **assuming a general Gaussian form** with undetermined coefficients:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \kappa_t \mathbf{x}_t + \lambda_t \mathbf{x}_0, \sigma_t^2 \mathbf{I})$$

$$\int p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) p(\mathbf{x}_t \mid \mathbf{x}_0) d\mathbf{x}_t = p(\mathbf{x}_{t-1} \mid \mathbf{x}_0)$$

The General Solution (Solving for Coefficients)

By enforcing consistency with the marginals $q(\mathbf{x}_t|\mathbf{x}_0)$, we solve for κ_t and λ_t, leaving σ_t as a free parameter:

$$\mathbf{x}_{t-1} = \underbrace{\sqrt{\bar{\alpha}_{t-1}} \left(\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta}{\sqrt{\bar{\alpha}_t}} \right)}_{\text{Predicted } \mathbf{x}_0 \text{ (Denoised)}} + \underbrace{\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2} \cdot \boldsymbol{\epsilon}_\theta}_{\text{Direction pointing to } \mathbf{x}_t} + \underbrace{\sigma_t \mathbf{z}}_{\text{Random Noise}}$$

DDIM: Deterministic Acceleration

The Special Case: $\sigma_t = 0$ (DDIM): Setting $\sigma_t = 0$ removes randomness, making the process deterministic.

1. ODE Interpretation:

- The trajectory from $\mathbf{x}_T$ to $\mathbf{x}_0$ is fixed.
- Implicitly models a probability flow ODE.

2. Subsequence Acceleration:

- Since marginals $\bar{\alpha}_t$ are known, we can generate via a subset of steps τ (e.g., $1000 \to \cdots \to 0$).
- **Result**: 10-50$\times$ speedup.

Model	Variance σ_t	Property
DDPM	$\sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t}$	Stochastic, Markovian
DDIM	**0**	**Deterministic, Non-Markovian**

Song J, Meng C, Ermon S. "Denoising diffusion implicit models," ICLR 2021.

Classifier Guidance

The Intuition: To generate data conditioned on class $\mathbf{y}$, we can modify the reverse transition using Bayes' rule:

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}) \propto p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{y}|\mathbf{x}_{t-1})$$

- We use a pre-trained **noisy classifier** $p_\phi(\mathbf{y}|\mathbf{x}_t)$ to guide the diffusion.
- The gradient $\nabla_{\mathbf{x}_t} \log p_\phi(\mathbf{y}|\mathbf{x}_t)$ pushes the sample towards high-likelihood regions for class $\mathbf{y}$.

Perturbed Mean with Gradient Scale s

Using Taylor expansion, the new sampling mean shifts by the classifier gradient:

$$\hat{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{y}) \approx \boldsymbol{\mu}_\theta(\mathbf{x}_t) + s \cdot \sigma_t^2 \nabla_{\mathbf{x}_t} \log p_\phi(\mathbf{y}|\mathbf{x}_t)$$

Limitation: Requires training a separate classifier on noisy data, which is computationally expensive and complex to align.

Classifier-Free Guidance

The Solution: Instead of an explicit classifier, we train a single model ϵ_θ to handle both conditional ($\mathbf{c}$) and unconditional ($\emptyset$) generation.

Mechanism: From Implicit Gradient to Sampling

Substituting the implicit gradient $\nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\mathbf{x}_t) \propto \epsilon_\theta(\mathbf{x}_t, \mathbf{c}) - \epsilon_\theta(\mathbf{x}_t, \emptyset)$ yields the extrapolated noise prediction:

$$\tilde{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) = \underbrace{\epsilon_\theta(\mathbf{x}_t, \emptyset)}_{\text{Uncond.}} + w \cdot \underbrace{(\epsilon_\theta(\mathbf{x}_t, \mathbf{c}) - \epsilon_\theta(\mathbf{x}_t, \emptyset))}_{\text{Guidance Direction}}$$

Trade-off (w):

- $w = 1$: Standard conditional model.
- $w > 1$: High fidelity to $\mathbf{c}$, lower diversity.

Ho J, Salimans T. "Classifier-free diffusion guidance," NeurIPS Workshop 2021.

Guidance Paradigms: Perception vs. Physics

Different guidance mechanisms serve distinct goals in Generative AI. We choose the paradigm based on whether we prioritize **visual fidelity** or **physical/logical constraints**.

1. Implicit Guidance (CFG)

Paradigm: Classifier-Free Guidance.

Focus: **Visual Perception** (Text-to-Video).

- implicitly learns distributions via $\epsilon_\theta(x, \emptyset)$.
- Prioritizes texture, lighting, and semantic alignment.

Status: **Standard for Content Generation** (e.g., Sora, Stable Diffusion) where "looking real" is the primary metric.

2. Explicit Guidance (CG / Value)

Paradigm: Gradient-Guided Optimization.

Core Task: **Constraint-Satisfying Gen.**

- Tasks requiring outputs to satisfy **verifiable rules** (physics, chemistry, logic).
- "Correctness" $\gg$ "Aesthetics".

Mechanism: Injects external gradients $\nabla_{\mathbf{x}} \mathcal{J}(\mathbf{x})$ (e.g., from Physics Engines, Reward Models, Classifiers).

Insight: While CFG dominates visual generation, CG remains the key interface for injecting **explicit physical constraints** in World Models.

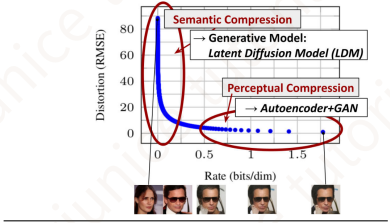
Stable Diffusion (Latent Diffusion Models)

Motivation: Why Latent Space?

Pixel space diffusion is computationally prohibitive. LDM solves this by shifting the process into a compressed representation.

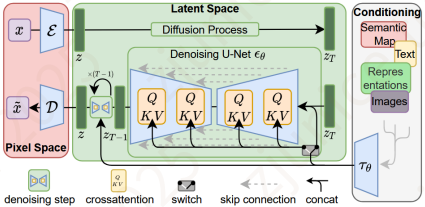
Step 1: Perceptual

Use a VAE to encode images into a lower-dimensional **latent space**, preserving details.



Step 2: Semantic

Perform the diffusion process in this **efficient space** to generate semantic content.



Rombach R, Blattmann A, Lorenz D, et al. "High-resolution image synthesis with latent diffusion models,"in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition,10684-10695, 2022.

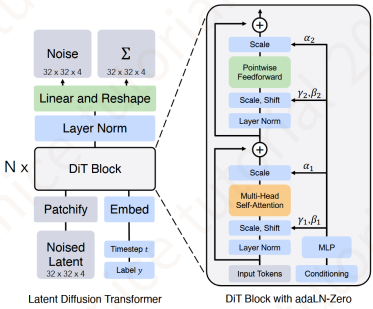
Diffusion Models with Transformers (DiT)

1. Architectural Shift (CNN → Transformer)

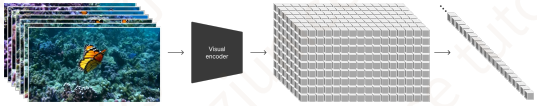
- Traditional diffusion models rely on a U-Net (CNN-based) backbone.
- DiT replaces this with a Transformer, an architecture proven to scale effectively (like LLMs).

2. Image/Video as a Sequence of Patches

- Re-imagines visual data, converting inputs into a unified sequence of patches.
- This flexible representation enables handling variable resolutions and durations.



The DiT Architecture



Spacetime Patches

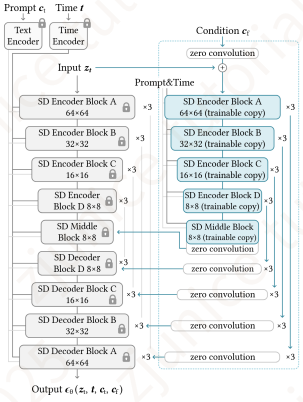
Peebles W, Xie S. "Scalable diffusion models with transformers ,"in Proceedings of the IEEE/CVF international conference on computer vision, 2023: 4195-4205.

ControlNet: Precise Spatial Control

Motivation: Standard diffusion models excel at prompts but struggle with precise spatial control (e.g., specific pose, depth).

Mechanism

- **Locked Copy**: Frozen backbone preserves pre-trained quality.
- **Trainable Copy**: Cloned branch learns features from condition **c**.
- **Zero Convolution**: 1×1 convs initialized with zeros.
- *Why Zero?* Ensures **Safe Injection** (model behaves like original at start).

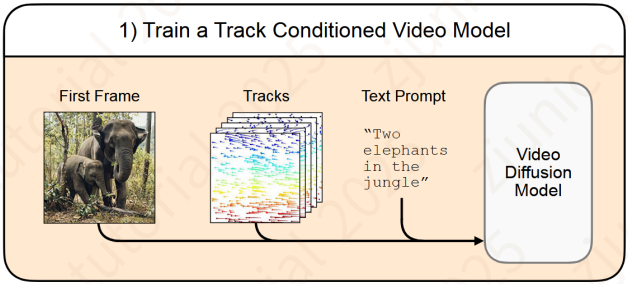


Zhang L, Rao A, Agrawala M. "Adding conditional control to text-to-image diffusion models, " in Proceedings of the IEEE/CVF international conference on computer vision. 2023: 3836-3847.

Motion Prompting: Trajectory Control

Framework: Adding fine-grained motion control.

- **Condition**: Motion represented as sparse point tracks.
- **Pipeline**:
  1. Representation: Motion is modeled as a set of sparse point trajectories.
  2. Embed tracks into spatiotemporal volume.
  3. Use ControlNet adapter to generate video.

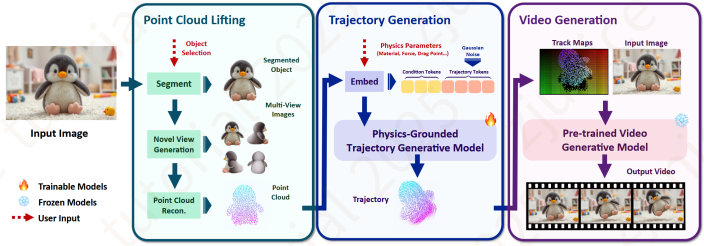


Geng D, Herrmann C, Hur J, et al. "Motion prompting: Controlling video generation with motion trajectories" in Proceedings of the Computer Vision and Pattern Recognition Conference, 2025: 1-12.

PhysCtrl: Physics-Grounded Generation

Overview: A framework for physics grounded image-to-video generation. **Method Pipeline:**

- **1. Point Cloud Lifting**: Lift the input image into a **3D point cloud** to capture explicit geometry.
- **2. Trajectory Generation**: A Diffusion Transformer (DiT) predicts the **future 3D trajectory** based on physics conditions.
- **3. Video Generation**: The predicted **trajectory** serves as strong guidance for the video generation model.



Wang C, Chen C, Huang Y, et al. "PhysCtrl: Generative Physics for Controllable and Physics-Grounded Video Generation," arXiv preprint arXiv:2509.20358, 2025.

Flow Matching and Its Evolutionary Paradigms

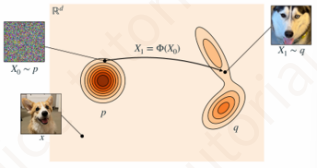
Flow Matching: From SDE to ODE

The Core Idea: Instead of modeling noise removal (SDE), we model a deterministic particle transport (ODE) from noise $\pi(\mathbf{x}_0)$ to data $q(\mathbf{x}_1)$.

1. The Flow Map & Vector Field

We define a time-dependent diffeomorphism $\phi_t : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ where $\phi_0 \sim \pi$ (Noise) and $\phi_1 \sim q$ (Data).
The flow follows an Ordinary Differential Equation (ODE):

$$\frac{d}{dt}\phi_t(\mathbf{x}) = v_t(\phi_t(\mathbf{x})), \quad v_t \text{ is the velocity field.}$$



2. The Continuity Equation

The probability density path p_t satisfies mass conservation:

$$\frac{\partial p_t}{\partial t} + \nabla \cdot (p_t v_t) = 0$$

*This establishes the link between v_t and p_t.

Deriving the Objective: Conditional Flow Matching

- **The Problem**: We cannot solve for the marginal $v_t(\mathbf{x})$ directly because intermediate densities p_t are intractable.
- **The Solution**: We regress onto a *conditional* vector field $v_t(\mathbf{x}|\mathbf{x}_1)$ (path of a single sample), which is analytically tractable.

Conditional Linear Path (Optimal Transport)

Define a straight path between noise $\mathbf{x}_0$ and data $\mathbf{x}_1$:

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1 \implies \frac{d}{dt}\mathbf{x}_t = \mathbf{x}_1 - \mathbf{x}_0$$

$$\text{Target Velocity } v_t(\mathbf{x}|\mathbf{x}_0, \mathbf{x}_1) = \mathbf{x}_1 - \mathbf{x}_0$$

Final Loss Function $\mathcal{L}_{\text{CFM}}$

We minimize the regression error over all times t and pairs $(\mathbf{x}_0, \mathbf{x}_1)$:

$$\mathcal{L}(\theta) = \mathbb{E}_{t, q(\mathbf{x}_1), p(\mathbf{x}_0)} \left[\| v_\theta(\mathbf{x}_t, t) - (\mathbf{x}_1 - \mathbf{x}_0) \|^2 \right]$$

Flow Matching: Training & Inference Algorithms

Algorithm 1: FM Training

repeat
  Sample data $\mathbf{x}_1 \sim q(\mathbf{x}_{\text{data}})$
  Sample noise $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Sample time $t \sim \text{Uniform}(0, 1)$
  Interpolate: $\mathbf{x}_t = (1 - t)\mathbf{x}_0 + t\mathbf{x}_1$
  Target: $v_{\text{tgt}} = \mathbf{x}_1 - \mathbf{x}_0$
  Descent on $\nabla_\theta \| v_\theta(\mathbf{x}_t, t) - v_{\text{tgt}} \|^2$
until converged

Algorithm 2: ODE Sampling

$\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Define step size $\Delta t = 1/N$
for $t = 0$ to 1 step Δt **do**
  // Euler Solver Step
  $\mathbf{x}_{t+\Delta t} = \mathbf{x}_t + v_\theta(\mathbf{x}_t, t) \cdot \Delta t$
end for
return $\mathbf{x}_1$

Lipman, Yaron, et al. "Flow Matching for Generative Modeling." The Eleventh International Conference on Learning Representations.

Paradigm Shift: Diffusion vs. Flow Matching

Diffusion (DDPM/SDE)

Perspective: *Denoising*

- **Path**: Curved & Stochastic

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$$

- **Target**: Predict Noise

$$\text{Target} = \boldsymbol{\epsilon}$$

- **Dynamics**: Requires complex SDE scheduling (β_t).

$\boldsymbol{\epsilon}$-prediction implies indirect velocity.

Flow Matching (ODE)

Perspective: *Vector Transport*

- **Path**: Straight & Deterministic

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$$

- **Target**: Predict Velocity

$$\text{Target} = \mathbf{x}_1 - \mathbf{x}_0$$

- **Dynamics**: Simple geometry. Noise ($t=0$) $\rightarrow$ Data ($t=1$).

Directly regresses the transport direction.

Rectified Flow: Learning Straight Generation Paths

Problem: Flow matching often learns curved paths → slow sampling and error accumulation.

Solution: Rectified Flow enforces straight-line trajectories.

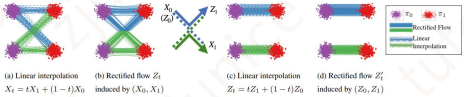
Core Training Objective

$$\min_v \mathbb{E}_{t\sim\mathcal{U}[0,1],x_0\sim p_0,x_1\sim p_1}\left[\|v_t(z_t)-(x_1-x_0)\|^2\right]$$

where $z_t = (1-t)x_0 + tx_1$ defines straight paths

Key Insights:

- Learns constant velocity: $v_t(z_t) \approx x_1 - x_0$
- Linear interpolation between data and noise
- Enables few-step sampling (e.g., 1-2 steps)



Curved → straight paths via rectification

Liu, X., Gong, C. and Liu, Q., 2023, January. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In The Eleventh International Conference on Learning Representations (ICLR).

Recitified flow Results

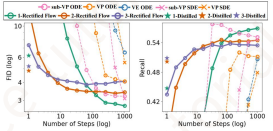
- **ODE Performance:** 1-Rectified Flow achieves **FID 2.58** and **Recall 0.57** with RK45 solver – best among ODE methods.
- **Efficiency:** Rectified Flows require **fewer NFE** than VP/sub-VP ODEs for comparable quality.
- **One-Step Generation:** Distilled k-Rectified Flows ($k = 1, 2, 3$) outperform previous distilled ODEs.

Method	NFE(↓)	IS (↑)	FID (↓)	Recall (↑)
ODE	*One-Step Generation (Euler solver, N=1)*			
1-Rectified Flow (+*Distill*)	1	1.13 (**9.08**)	378 (*6.18*)	0.0 (*0.45*)
2-Rectified Flow (+*Distill*)	1	8.08 (*9.01*)	12.21 (**4.85**)	0.34 (*0.50*)
3-Rectified Flow (+*Distill*)	1	8.47 (*8.79*)	8.15 (*5.21*)	0.41 (**0.51**)
VP ODE [73] (+*Distill*)	1	1.20 (*8.73*)	451 (*16.23*)	0.0 (*0.29*)
sub-VP ODE [73] (+*Distill*)	1	1.21 (*8.80*)	451 (*14.32*)	0.0 (*0.35*)
ODE	*Full Simulation (Runge–Kutta (RK45), Adaptive N)*			
1-Rectified Flow	127	**9.60**	**2.58**	**0.57**
2-Rectified Flow	110	9.24	3.36	0.54
3-Rectified Flow	104	9.01	3.96	0.53
VP ODE [73]	140	9.27	3.93	0.51
sub-VP ODE [73]	146	9.46	3.16	0.55
SDE	*Full Simulation (Euler solver, N=2000)*			
VP SDE [73]	2000	9.58	2.55	0.58
sub-VP SDE [73]	2000	9.56	2.61	0.58

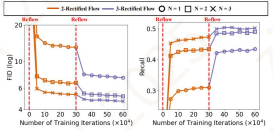
(a) Results using the DDPM++ architecture.

Method	NFE(↓)	IS (↑)	FID (↓)	Recall (↑)
GAN	*One-Step Generation*			
SNGAN [52]	1	8.22	21.7	0.44
StyleGAN2 [28]	1	9.18	8.32	0.41
StyleGAN-XL [66]	1		1.85	0.47
StyleGAN2 + ADA [28]	1	9.40	2.92	0.49
StyleGAN2 + DiffAug [58]	1	9.40	5.79	0.42
TransGAN + DiffAug [26]	1	9.02	9.26	0.41
GAN with U-Net	*One-step Generation*			
TDPM (T=1) [69]	1	8.65	8.91	0.46
Denoising Diffusion GAN (T=1) [9]	1	8.93	14.6	0.19
ODE	*One Step Generation (Euler solver, N=1)*			
DDIM Distillation [47]	1	8.36	9.36	0.51
NCSN++ (VE ODE) [73] (+*Distill*)	1	1.18 (2.57)	463 (154)	0.0 (*0.0*)
ODE	*Full Simulation (Runge–Kutta (RK45), Adaptive N)*			
NCSN++ (VE ODE) [73]	176	9.35	5.38	0.56
SDE	*Full Simulation (Euler solver)*			
DDPM [23]	1000	9.46	3.21	0.57
NCSN++ (VE SDE) [73]	2000	9.83	2.38	0.59

(b) Recent results with different architectures reported in literature.



(a) FID and Recall vs. Number of Euler discretization steps N



(b) FID and Recall vs. Training Iterations

MeanFlow: One-step generation

Definition & MeanFlow Identity

- Define average velocity $u(z_t, r, t)$ between time r and t:

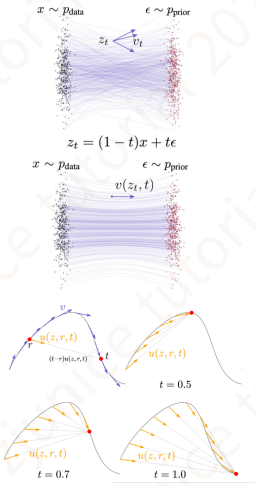
$$u(z_t, r, t) \triangleq \frac{1}{t-r} \int_r^t v(z_\tau, \tau) d\tau$$

- Rearranging and differentiating with respect to t (Leibniz rule):

$$\frac{d}{dt}\left[(t-r)u(z_t, r, t)\right] = \frac{d}{dt} \int_r^t v(z_\tau, \tau) d\tau$$

- Applying product rule leads to the **MeanFlow Identity**:

$$\underbrace{u}_{\text{avg.}} = \underbrace{v}_{\text{inst.}} - \underbrace{(t-r)\frac{du}{dt}}_{\text{deriv.}}$$



Geng Z, Deng M, Bai X, Kolter JZ, He K. Mean flows for one-step generative modeling. Neurips 2025.

MeanFlow: One-step generation

Jacobian-Vector Product (JVP) for Implementation

- To solve the ODE $\frac{d}{dt} z_t = v(z_t, t)$, we compute the total derivative of u:

$$\frac{d}{dt} u(z_t, r, t) = \partial_z u \frac{dz_t}{dt} + \partial_r u \frac{dr}{dt} + \partial_t u \frac{dt}{dt} = [\partial_z u, \partial_r u, \partial_t u] \begin{bmatrix} v(z_t, t) \\ 0 \\ 1 \end{bmatrix}$$

- This is efficiently implemented as: `jvp(fn, (z, r, t), (v, 0, 1))`

Learning Objective

- We parameterize u_θ directly and use a stop-gradient (sg) on the target:

$$\mathcal{L}(\theta) = \mathbb{E} \| u_\theta(z_t, r, t) - \mathrm{sg}(u_{\mathrm{tgt}}) \|_2^2$$

$$u_{\mathrm{tgt}} = v(z_t, t) - (t - r) \underbrace{(v(z_t, t) \partial_z u_\theta + \partial_t u_\theta)}_{\text{computed by JVP}}$$

Meanflow Results

- **Promising Scalability:** As shown in Figure, MF exhibits strong scalability with model size, consistently improving 1-NFE FID from scratch.
- **SOTA 1-Step Quality:** Results demonstrate MF significantly outperforms previous 1-step methods (iCT, Shortcut, IMM).
- **Efficient Sampling:** Achieves high-fidelity results with only **1-NFE** (Number of Function Evaluations).
- **CFG Integration:** Successfully maintains 1-NFE behavior while applying Classifier-Free Guidance.

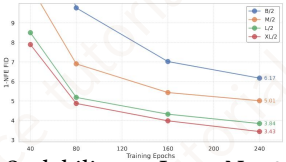


Figure: Scalability on ImageNet 256×256.

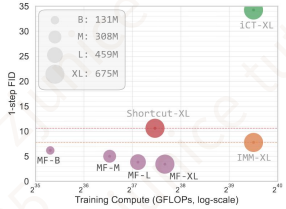


Figure: 1-NFE samples vs. previous methods.

Summary

From VAE to Diffusion and Flow Matching

- **Technical Evolution: From "Static Reconstruction" to "Dynamic Generation"**
  - Autoencoder (AE) solves data compression problems but has discontinuous latent space and no generative capability;
  - VAE reconstructs continuous latent space through variational inference, laying the probabilistic foundation for generative models;
  - DDPM breaks the modeling capability limitations of VAE and fits complex distributions through multi-step diffusion.
- **Advanced Diffusion Models: Dual Improvement of Efficiency and Controllability**
  - Architectural Optimization: Latent Diffusion (LDM) reduces computational costs, DiT achieves large-scale expansion;
  - Enhanced Controllability: ControlNet enables precise spatial control, PhysCtrl imposes physical consistency constraints to adapt to video generation scenarios.
- **Flow Matching: A New Paradigm for Continuous Flow Modeling**
  - Core Advantage: Replaces the stochastic process of diffusion with deterministic flow fields, achieving higher sampling efficiency;
  - Scenario Expansion: Rectified Flow and meanflow simplifies sampling paths.

Thank you

Zheng Shenxinkai / Fan Zhijie / Yang Shiyi
zsxk@zju.edu.cn, fanzhijie@zju.edu.cn, yangshiyi@zju.edu.cn
Networked Intelligence for Comprehensive Efficiency (NICE) Lab
College of Information Science and Electronic Engineering
Zhejiang University
https://nice.rongpeng.info