Hands-on AI based 3D Vision
Summer Semester 25

Lecture 10_0 – Diffusion Models

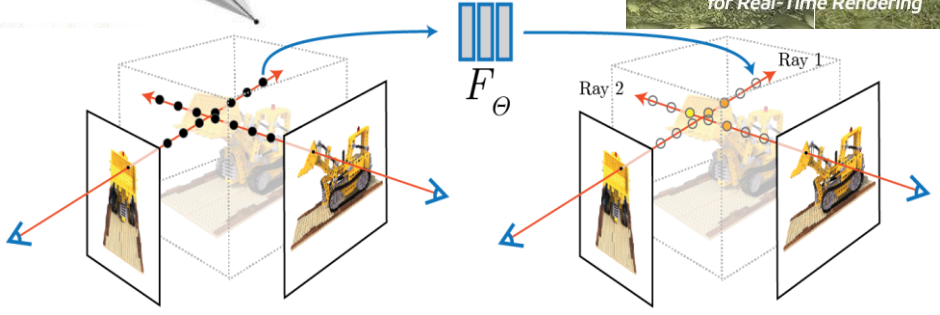
Prof. Dr.-Ing Gerard Pons-Moll
University of Tübingen / MPI-Informatics

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Motivations for Generative Models

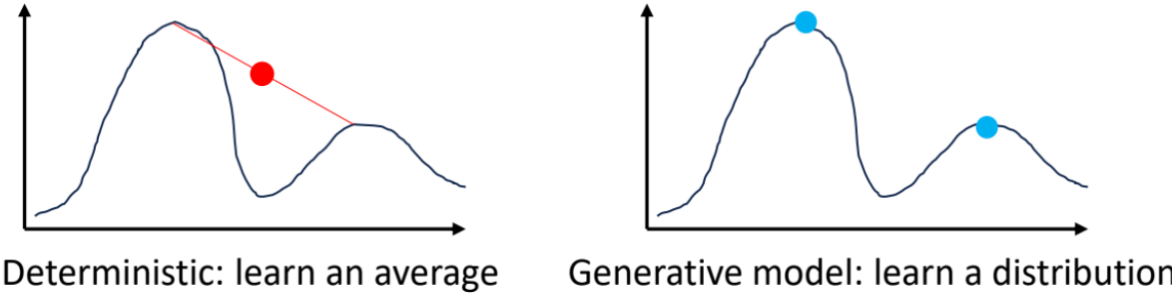
Motivations for Generative Models

- Until now we saw methods that can **reconstruct** scene and objects from multiple images (SFM, Multiview Stereo, NeRF, Gaussian Splatting...)



Motivation for generative models

- However, tasks like **single view reconstruction** are ill-posed.
- Deterministic model might collapse to average value.



- We should learn a distribution of all possible configurations instead of simply regression or fitting to observations
- Today we will explore diffusion model for conditional generation!

Generative Models in 3D

Generative Models in 3D

Being able to generate content is a very powerful capability, and has many applications.

Text to 3D object

Two-story brick house with red roof and fence.

Vintage copper rotary telephone with intricate detailing.

Bronze owl sculpture perched on a branch.

Bronze owl sculpture perched on a branch.

Trellis: Structured 3D Latents
for Scalable and Versatile 3D Generation. Xiang et al. CVPR 2025

Image to 3D object

Trellis: Structured 3D Latents
for Scalable and Versatile 3D Generation. Xiang et al. CVPR 2025

Generate 3D Scenes

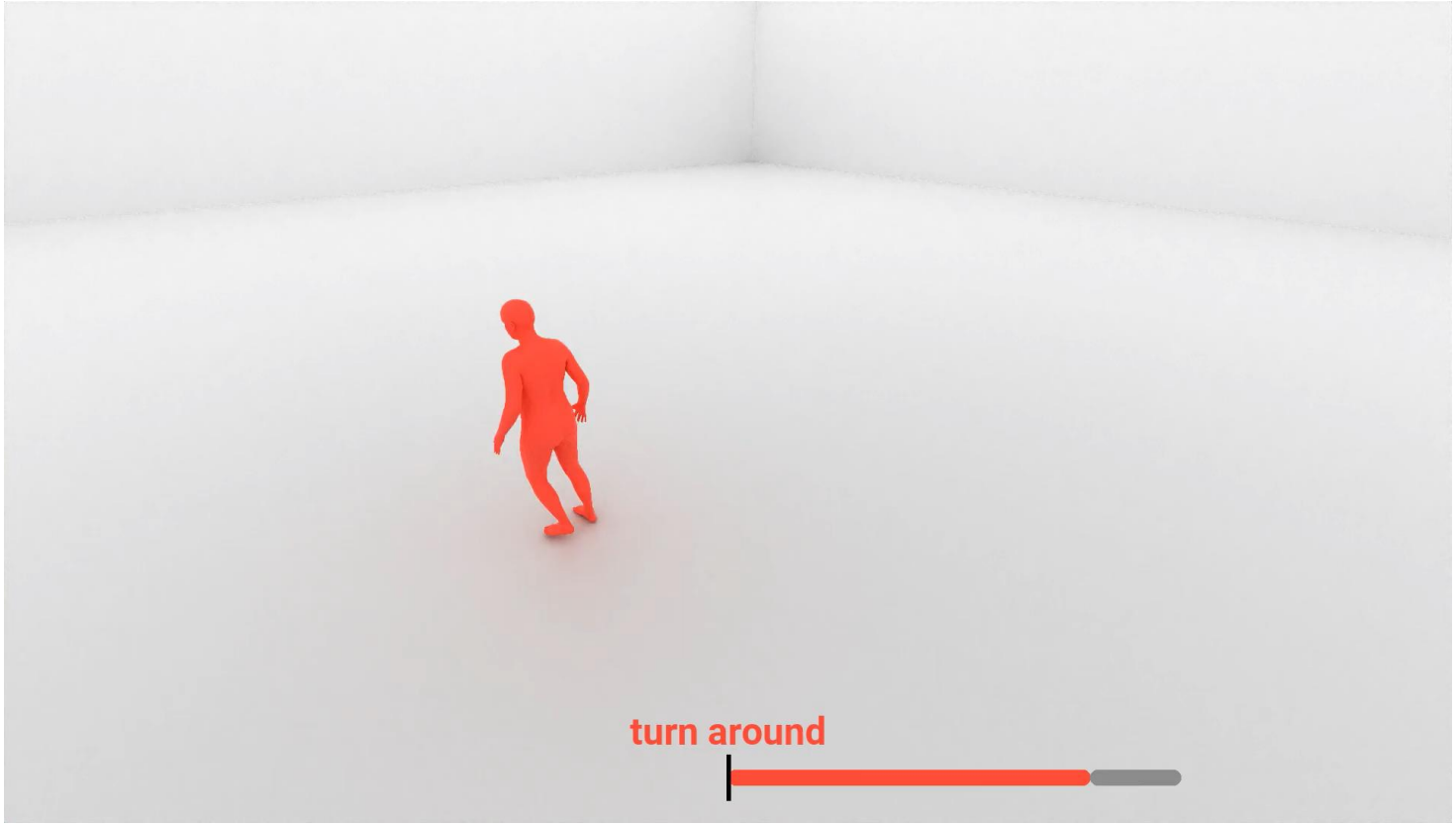
SynCity. Engstler et al. CVPR 2025

Generate 3D Scene... Still some work to do

WonderWorld. Yu et al. CVPR 2025

Scene Splatter. Zhang et al. CVPR 2025

Motion Generation



UniMotion. Li et al. 3DV 2025

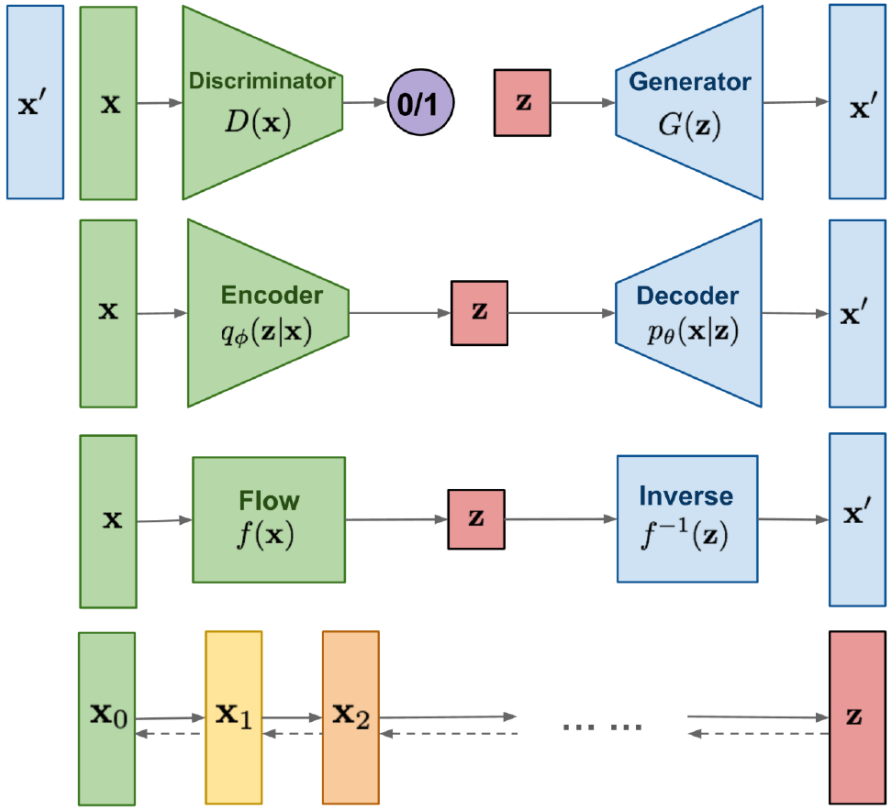
Generative Models – Main Architectures

GAN: Adversal training

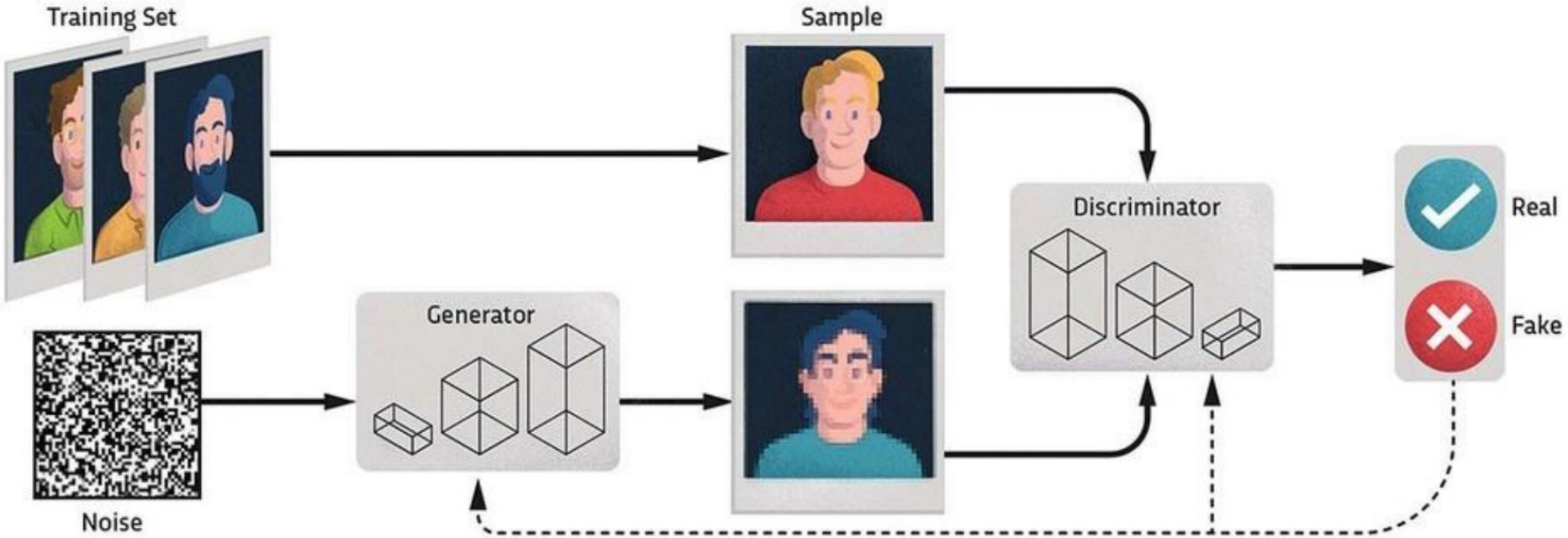
VAE: Maximize variational lower bound

Flow-based models: Invertible transform fo distribution

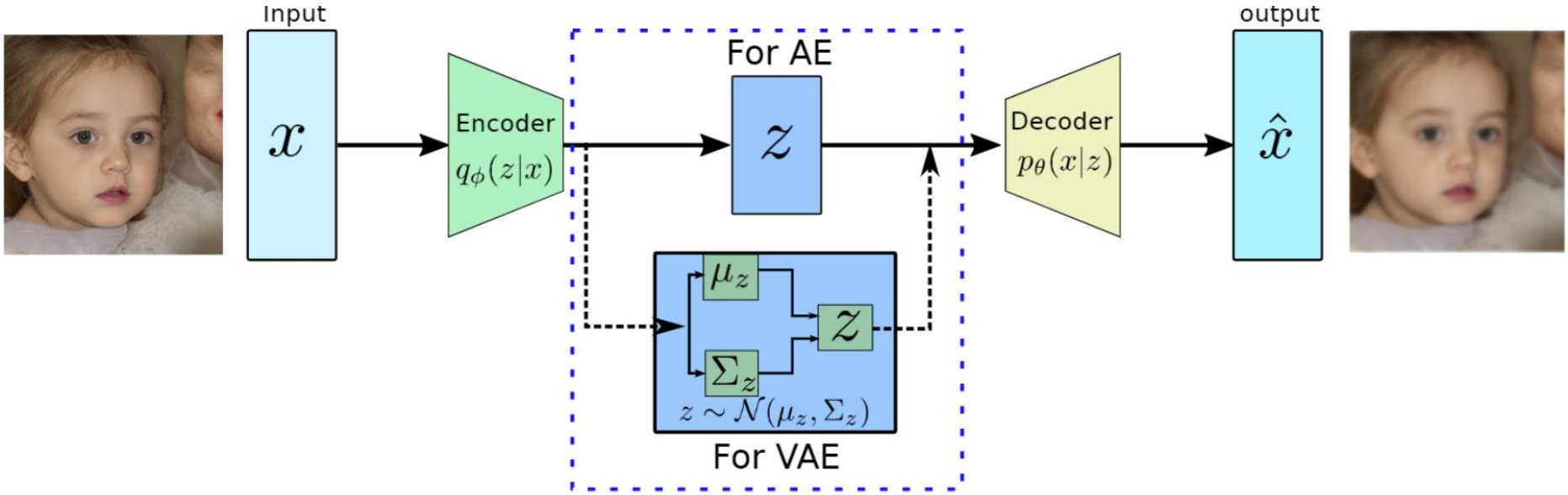
Diffusion Models: Gradually add Gaussian noise and then reverse



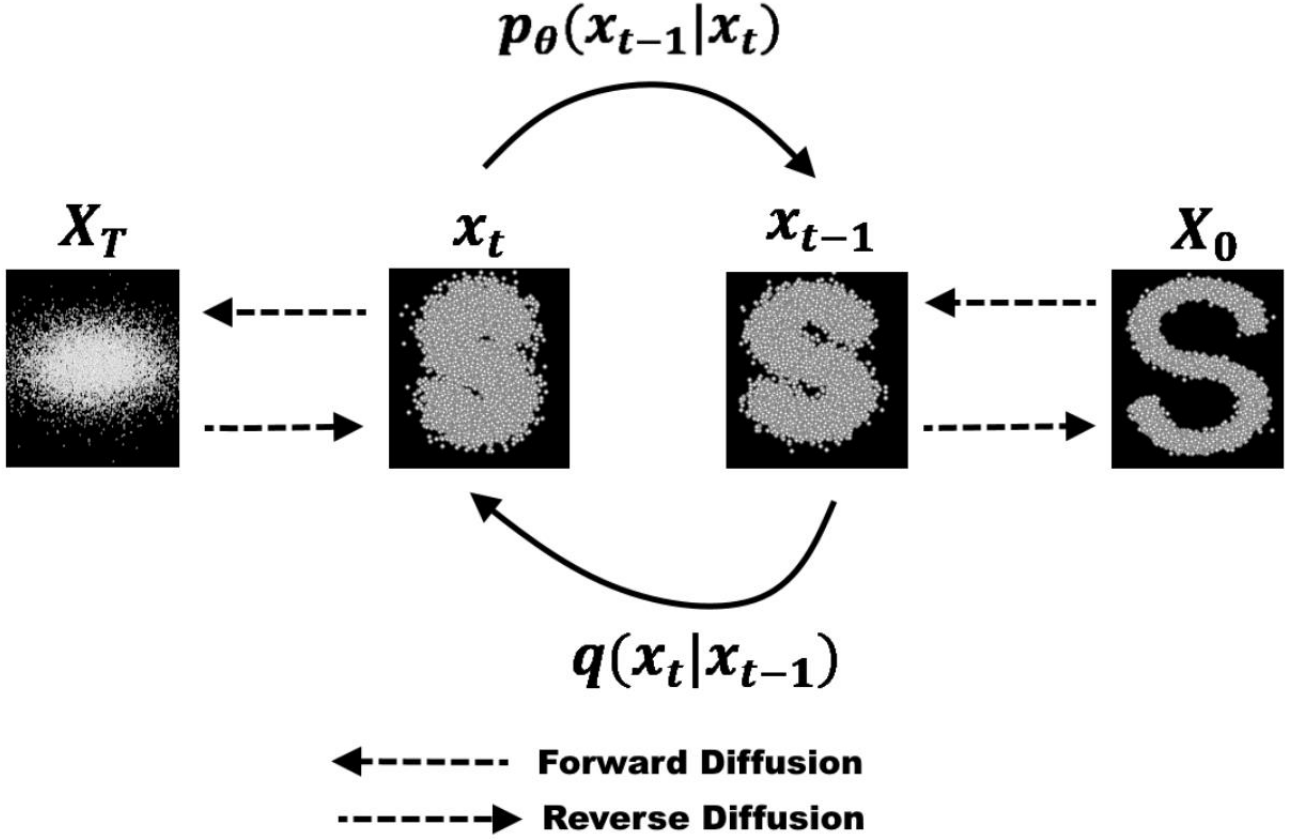
GAN


Training Set
Sample
Discriminator
Real
Fake
Generator
Noise

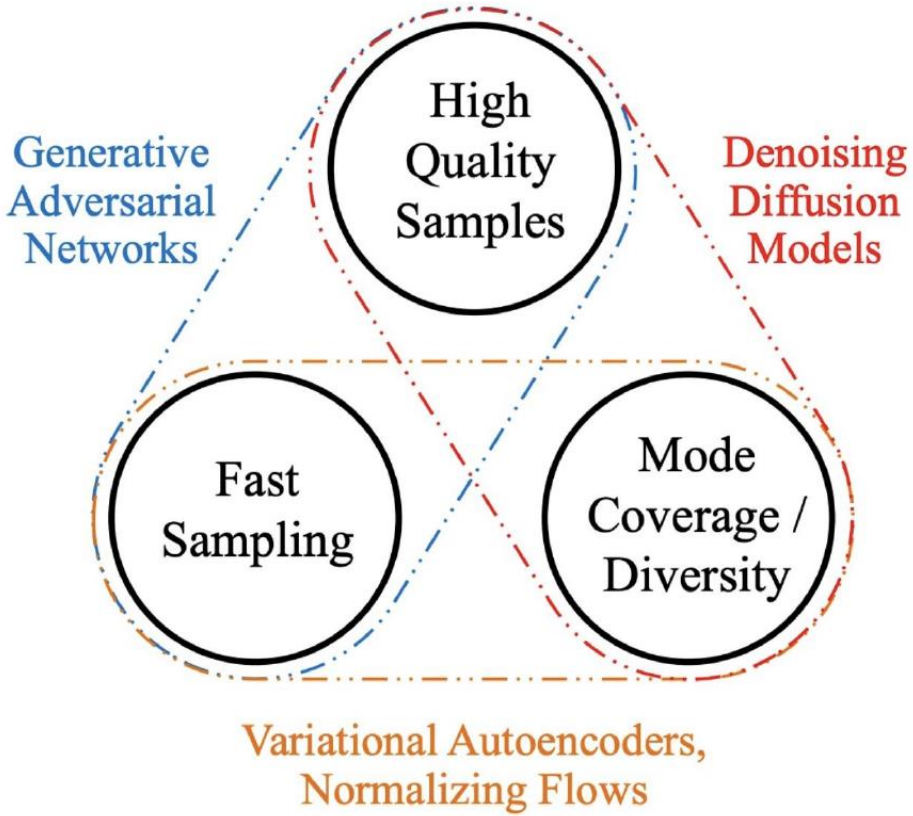
VAE


Input
x
Encoder
$q_\phi(z|x)$
For AE
z
μ_z
Σ_z
z
$z \sim \mathcal{N}(\mu_z, \Sigma_z)$
For VAE
Decoder
$p_\theta(x|z)$
output
$\hat{x}$

Diffusion Models



Trilemma: Quality, Diversity Speed



Diffusion Models

Diffusion Models for Image Generation

- Diffusion model is SOTA on image generation

- Stable Diffusion
- Flux
- Mid-Journey
- Dall-E
- ...

Diffusion Models for Image Editing

(a) **Context image** generated with FLUX.1.

(b) Image context from Figure 1a: *"The bird is now sitting in a bar and enjoying a beer."*

(c) Image context from Figure 1b: *"There are now two of these birds."*

(d) From Figure 1c: *"Watch them from behind."*

- Flux.1 Kontext (Jun 24 2025 !!!)

Diffusion Models for Image Editing

(a) Input image

(b) *"remove the thing from her face"*

(c) *"she is now taking a selfie in the streets of Freiburg, it's a lovely day out."*

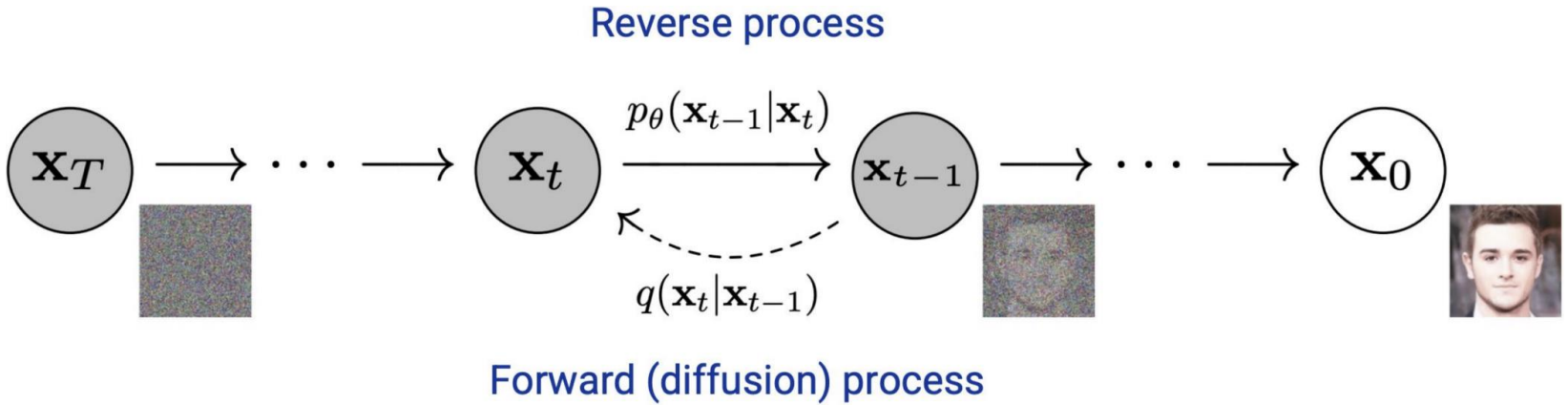
(d) *"it's now snowing, everything is covered in snow."*

- Flux.1 Kontext (Jun 24 2025 !!!)

Diffusion - Theory

Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Forward step**: (Iteratively) Add noise to the original sample
    - → The sample x_0 converges to the complete noise x_T (e.g., $\sim \mathcal{N}(0,1)$)
  - **Reverse step**: Recover the original sample from the noise
    - → Note that it is the “generation” procedure



Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Forward step**: (Iteratively) Add noise to the original sample
    - → Technically, it is a product of conditional noise distribution $q(x_t|x_{t-1})$
  - Usually, the parameters β_t are fixed (one can jointly learn them, but it is not beneficial)
  - Noise annealing (i.e., reducing noise scale $\beta_t < \beta_{t-1}$) is crucial to the performance

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

Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Forward step**: (Iteratively) Add noise to the original sample
    - → Technically, it is a product of conditional noise distribution $q(x_t|x_{t-1})$

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

  - **Reverse step**: Recover the original sample from the noise
    - → It is also a product of conditional (de)noise distribution $p_\theta(x_{t-1}|x_t)$
  - Use the learned parameters: denoiser μ_θ(main part) and randomness Σ_θ

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Forward step**: (Iteratively) Add noise to the original sample
  - **Reverse step**: Recover the original sample from the noise

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

  - **Training**: Minimize variational lower bound of the model $p_\theta(\mathbf{x}_0)$

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

Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Training**: Minimize variational lower bound of the model

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

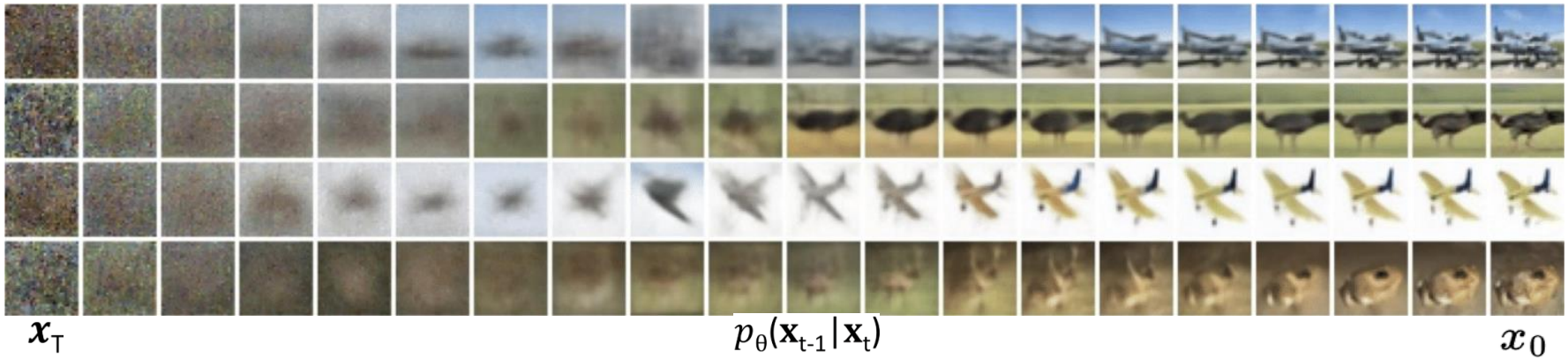
    - It can be decomposed to teh step-wise losses (for each step *t*)

$$\mathbb{E}_q \Bigg[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p(\mathbf{x}_T))}_{L_T} + \sum_{t>1} \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0} \Bigg]$$

1) Prior Matching 2) Denoising Matching 3) Reconstruction term

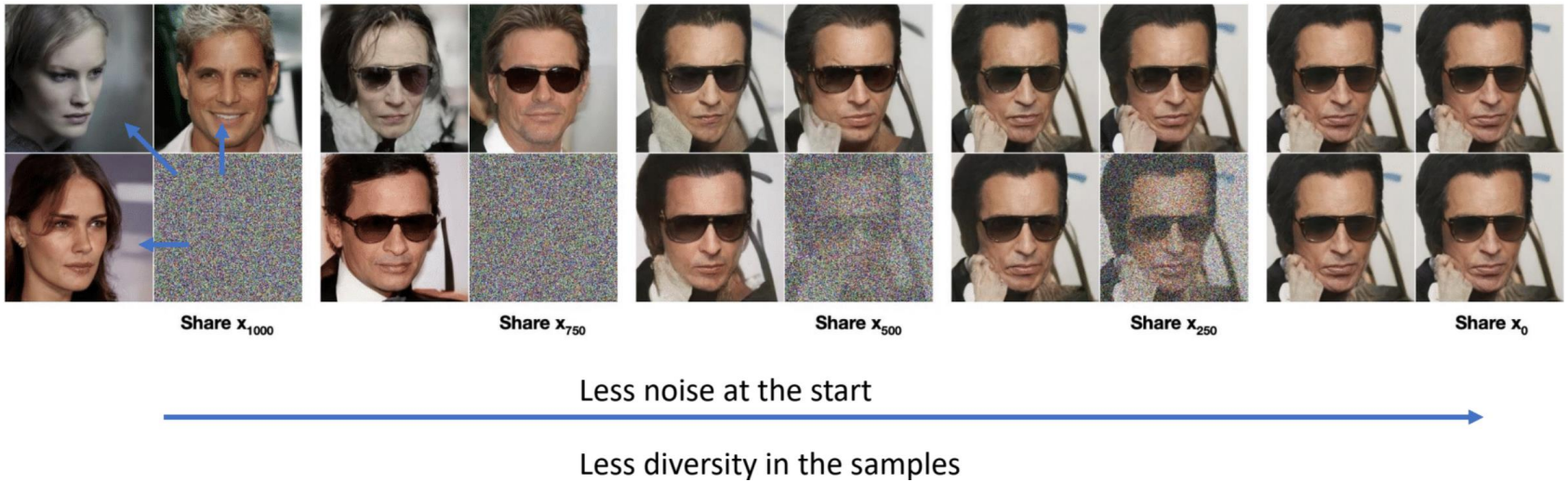
Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Sampling**: Draw a random noise $\boldsymbol{x}_T$ then apply the reverse step $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
  - It often requires the 1000 reverse steps (very slow)



Diffusion Probabilistic Models

- Diffusion model aims to learn the **reverse** of **noise generation** procedure
  - **Sampling**: Draw a random noise $\boldsymbol{x}_T$ then apply the reverse step $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
  - It often requires the 1000 reverse steps (very slow)



Denoising Diffusion Probabilistic Models (DDPM)

- DDPM reparametrizes the reverse distributions of diffusion models
  - Key idea: Predict the noise instead of the denoised signal $\mathbf{x}_{t-1}$
    - Since $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$ share most information, us it is redundant
    - → Instead, predict the **residual** $\epsilon_\theta(\mathbf{x}_t, t)$ and add to the original
  - Formally, DDPM reparametrizes the learned reverse distribution as

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

- and the step
  - wise objective L_{t-1} can be reformulated as

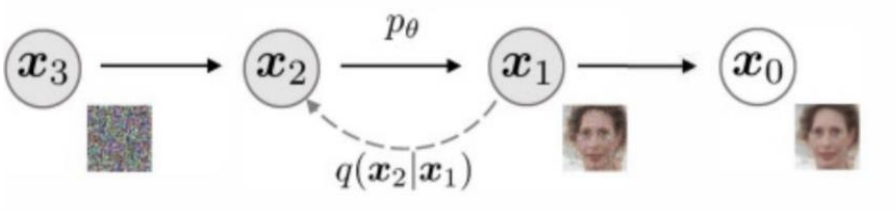
$$\mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}} \left[\left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2 \right]$$

Denoising Diffusion Implicit Models (DDIM)

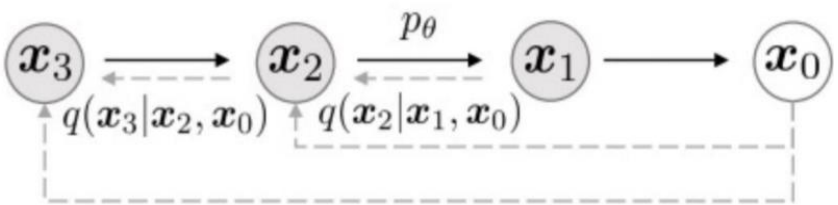
- DDIM roughly sketches the final sample, then refine it with the reverse process
  - **Motivation:**
    - Diffusion model is slow due to the iterative procedure
    - GAN/VAE creates the sample by one-shot forward operation
    - ⇒ Can we combine the advantages for fast sampling of diffusion models?
  - Technical spoiler:
    - Instead of naïvely applying diffusion model upon GAN/VAE, DDIM proposes a principled approach of rough sketch + refinement

Denoising Diffusion Implicit Models (DDIM)

- DDIM roughly sketches the final sample, then refine it with the reverse process
  - **Key Idea:**
    - Given $\boldsymbol{x}_T$, generate the rough sketch x_0 and refine $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
    - Unlike original diffusion model, it is not a Markovian structure



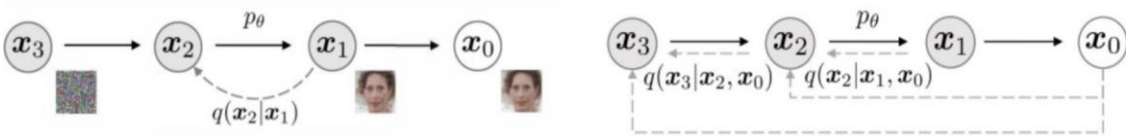
Original Diffusion



Non-Markovian

Denoising Diffusion Implicit Models (DDIM)

- DDIM roughly sketches the final sample, then refine it with the reverse process
  - **Key Idea:**
    - Given $\boldsymbol{x}_T$, generate the rough sketch x_0 and refine $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$



  - **Formulation:** Define the forward distribution $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ as

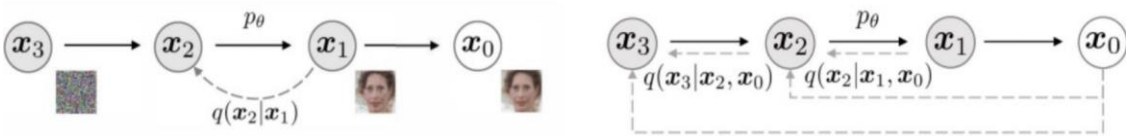
$$q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\boldsymbol{x}_0 + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\cdot\frac{\boldsymbol{x}_t - \sqrt{\alpha_t}\boldsymbol{x}_0}{\sqrt{1-\alpha_t}}, \sigma_t^2\boldsymbol{I}\right)$$

  - then, the **forward process** is derived from Bayes' rule

$$q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = \frac{q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}$$

Denoising Diffusion Implicit Models (DDIM)

- DDIM roughly sketches the final sample, then refine it with the reverse process
  - **Key Idea:**
    - Given $\boldsymbol{x}_T$, generate the rough sketch x_0 and refine $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$

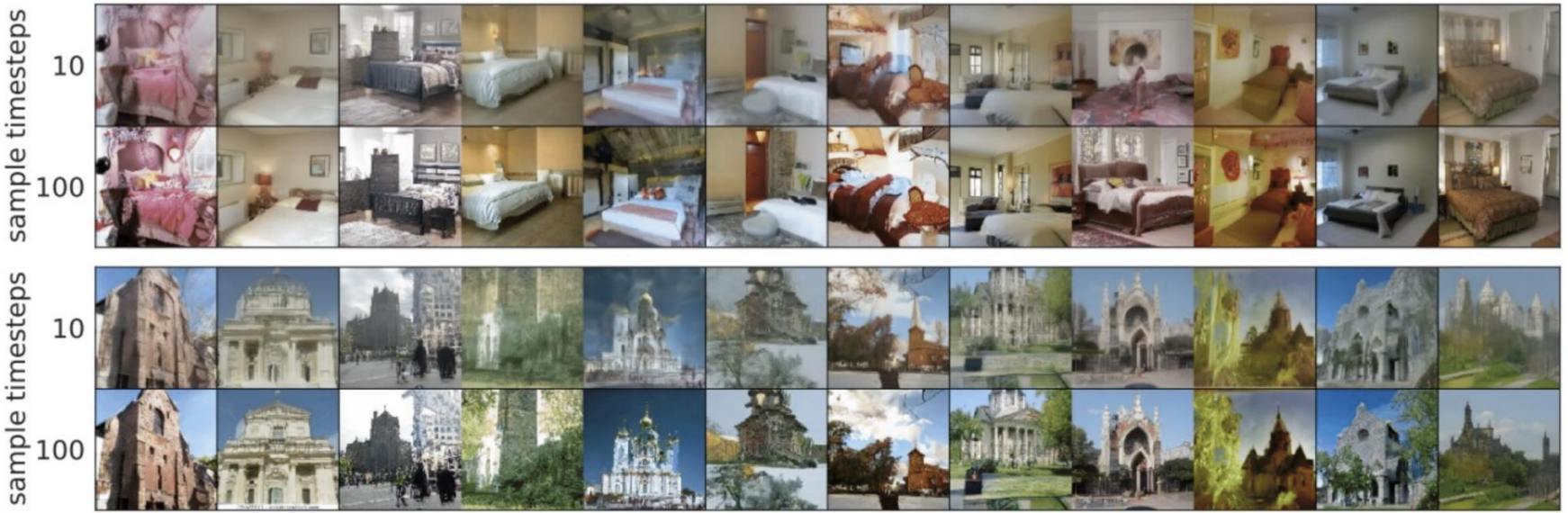


  - **Formulation:** Forward process is $$q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = \frac{q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)q_\sigma(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}$$
    - And the reverse process is

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}} \right)}_{\text{“ predicted } \boldsymbol{x}_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{“direction pointing to } \boldsymbol{x}_t\text{”}} + \underbrace{\sigma_t \epsilon_t}_{\text{random noise}}$$

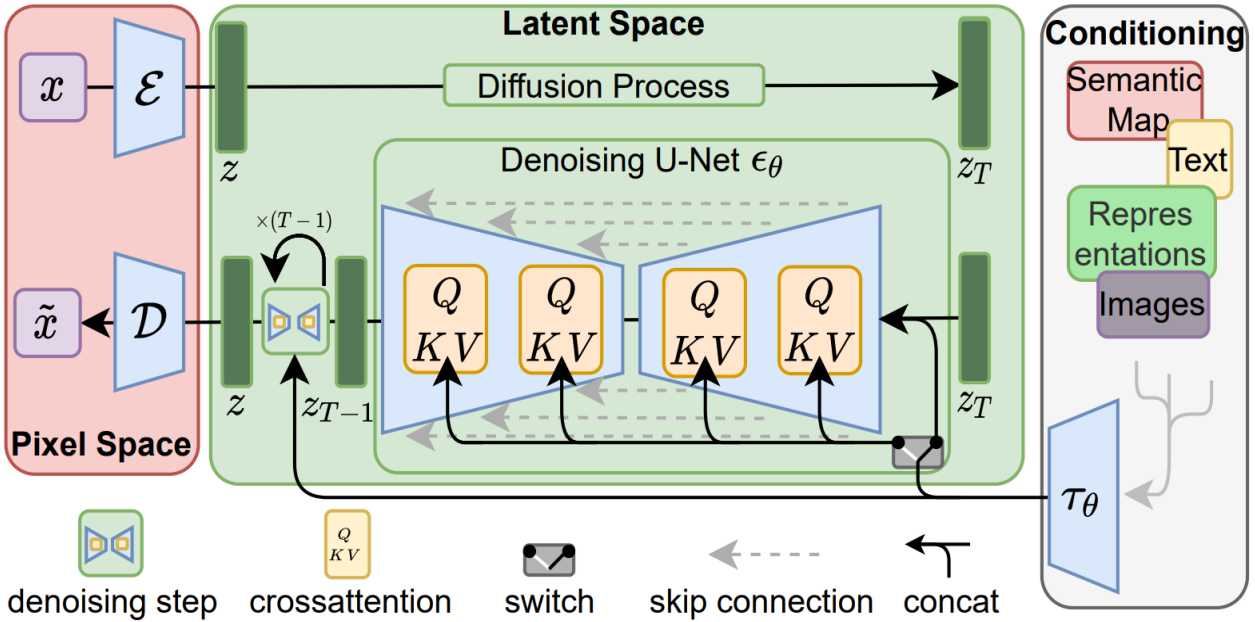
Denoising Diffusion Implicit Models (DDIM)

- DDIM significantly reduces the sampling steps of diffusion model
  - Creates the outline of the sample after only 100 steps (DDPM needs thousands)



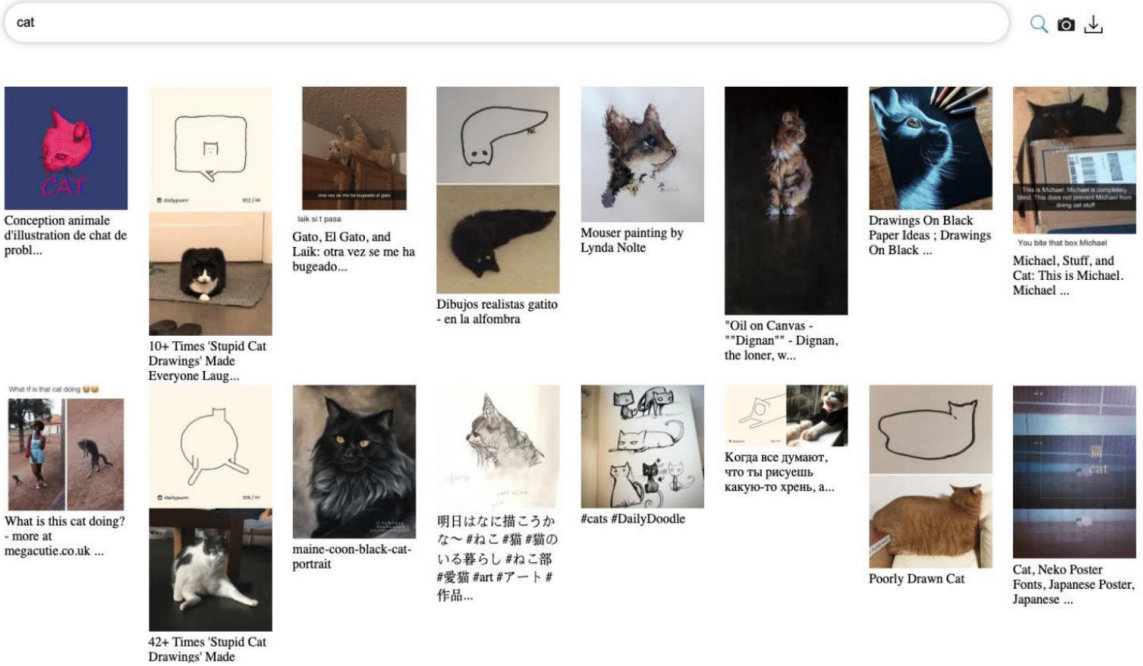
Stable Diffusion

- Stable Diffusion, Rombach et al. 2022
  - **Key Idea:** Run the Diffusion Process in a **latent space**.



Stable Diffusion

- Stable Diffusion, Rombach et al. 2022
  - **Large-Scale Training:** Laion 5B, containing 5 Billion Images with text annotations



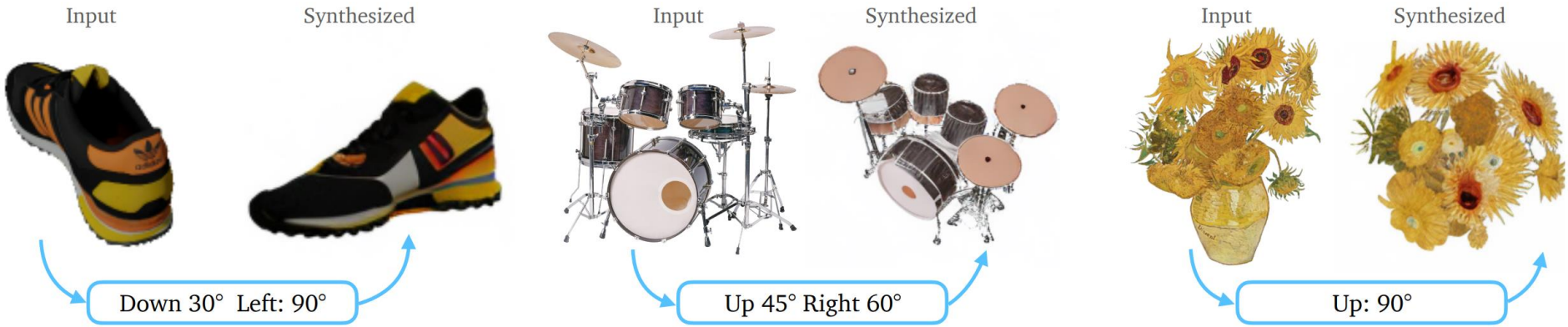
Multiview Image Diffusion

From Image Diffusion to 3D

- Do 2D Diffusion Models have knowledge on 3D domain?
- How can we leverage image diffusion for 3D?

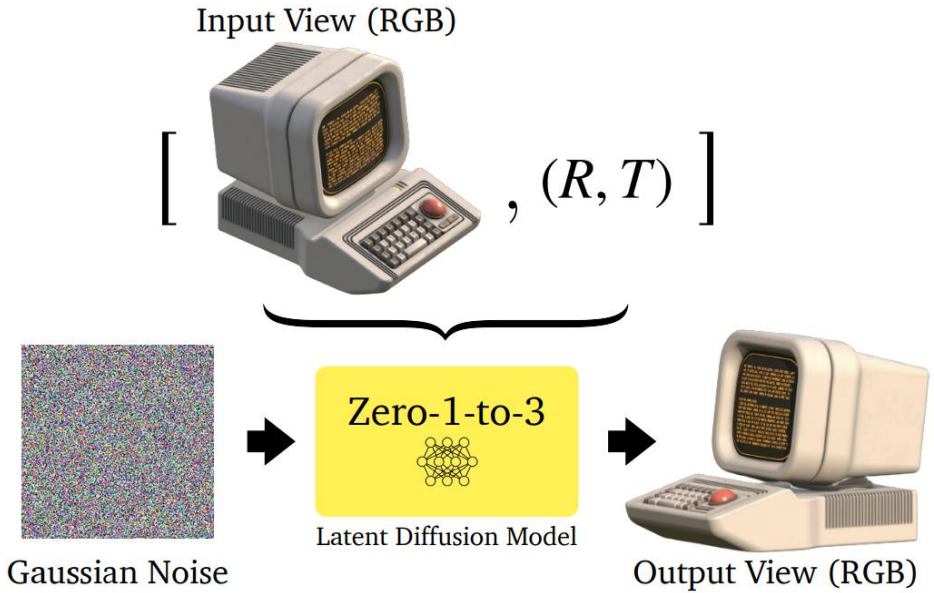
Multiview Image Generation

- Multiview Diffusion
  - **Goal:** to produce **multiple views** of the same object or scene in a way that's **geometrically consistent**.



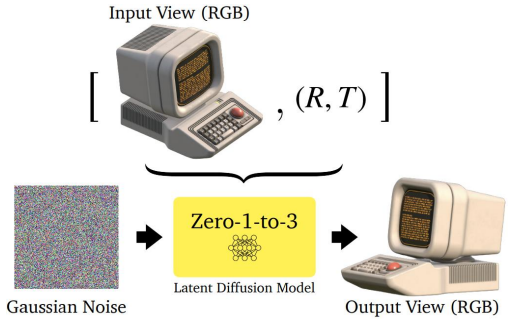
Zero-1-to-3

- Zero-1-to-3: Train a diffusion model to generate novel views of an object.
  - **Key Idea:** Condition the geneation on an image of the object and rotation an(R, T) .
  - Trained on Objaverse



Zero-1-to-3

- Zero-1-to-3: Train a diffusion model to generate novel views of an object.
  - **Input**:
    - Image of the object and rotation angles (R, T).



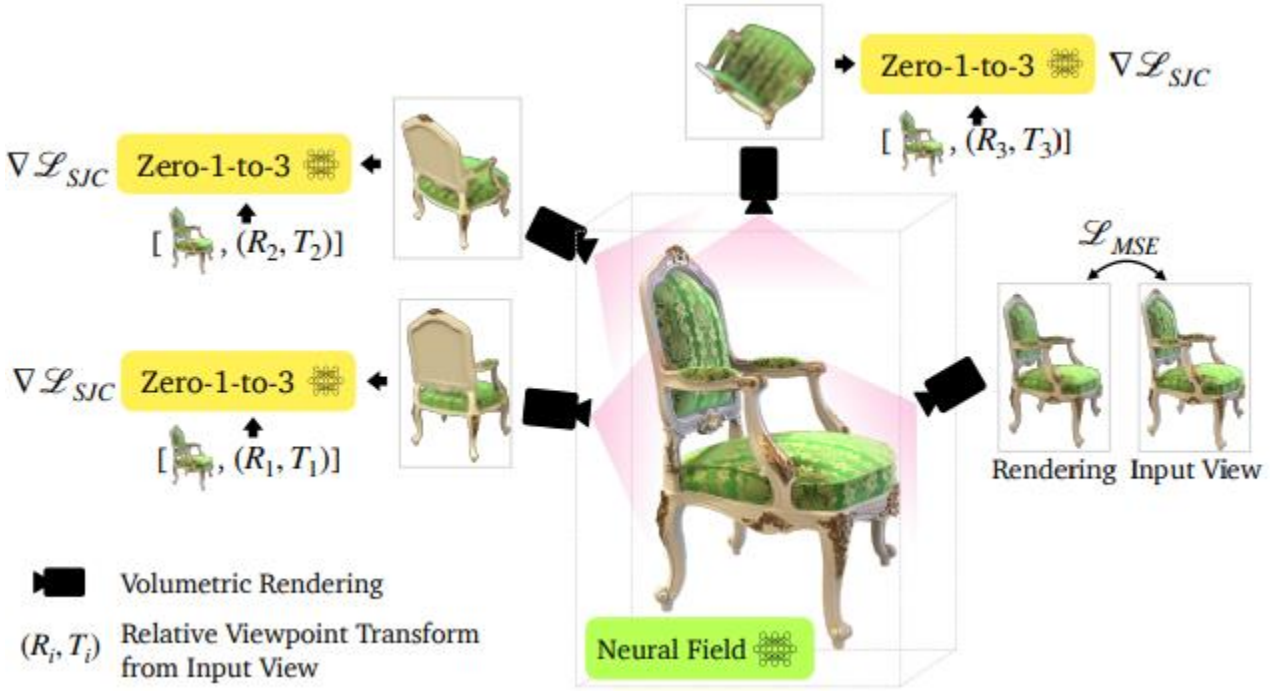
  - **Training Objective**:

$$\min_{\theta} \mathbb{E}_{z\sim\mathcal{E}(x),t,\epsilon\sim\mathcal{N}(0,1)}||\epsilon - \epsilon_\theta(z_t, t, c(x, R, T))||_2^2.$$

Where $c(x, R, T)$ is the embedding of the input view and relative camera extrinsics

Zero-1-to-3

- Zero-1-to-3: Train a diffusion model to generate novel views of an object.
  - **Reconstruction**:



Zero-1-to-3

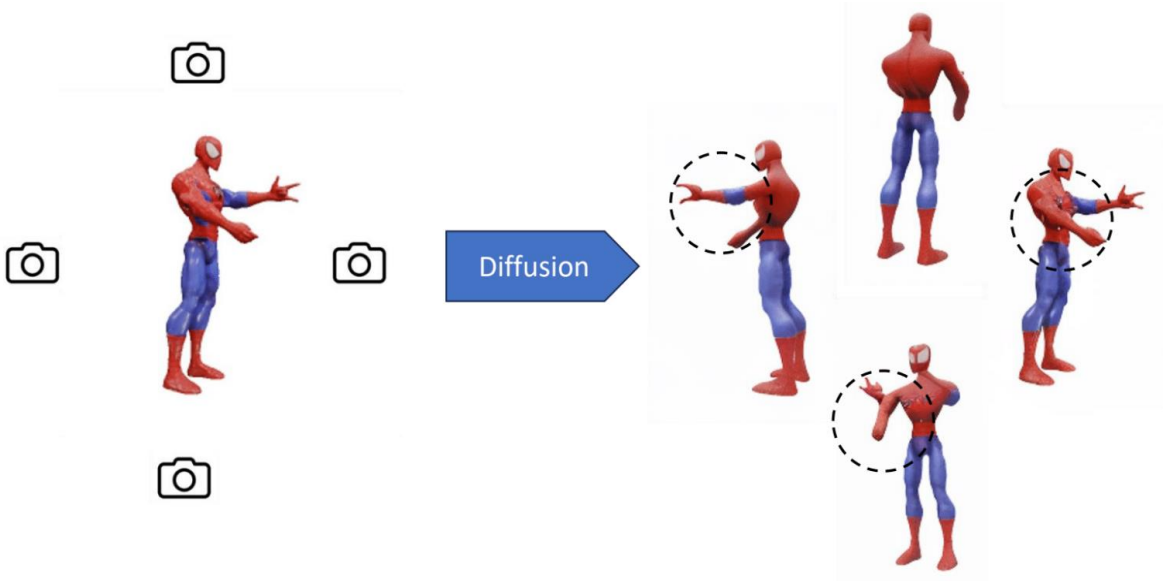
- Zero-1-to-3: Train a diffusion model to generate novel views of an object.

Zero-1-to-3

- Zero-1-to-3: Train a diffusion model to generate novel views of an object.
  - **Problem: consistency**

Zero-1-to-3

- 2D Multi View Diffusion has no explicit 3D representation (e.g. NeRF or 3DGS). Thus, the 3D consistency of the generated images are not constrained.



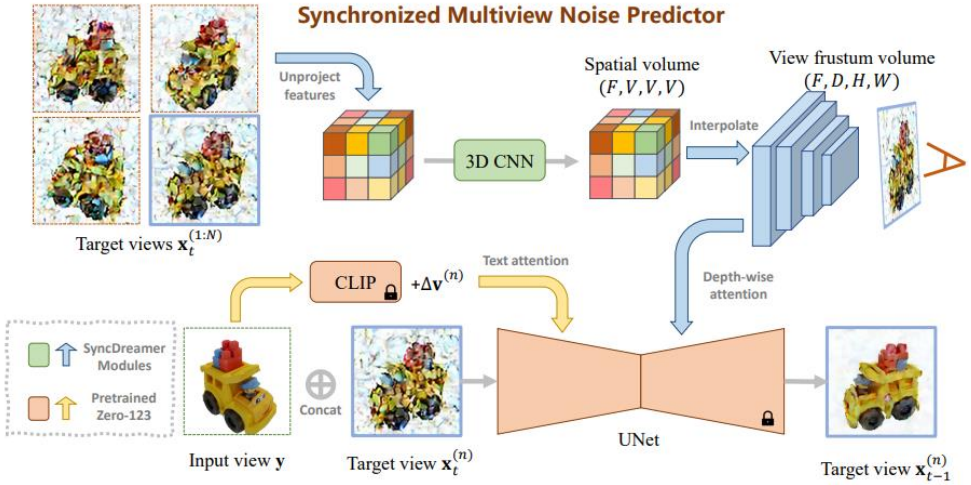
Multiview Image Generation

- Problem: novel views are generated **independently**.

- Recent works that improve consistency:
  - **Synchronized Multiview Noise Prediction**
    - SyncDreamer, Liu et al. ICLR 24
  - **Epipolar Geometry**
    - EpiDiff, Huang et al. CVPR 24
  - **Temporal Consistency**
    - SV3D, Voleti et al. ECCV 24
  - **Leverage 3D Consistency**
    - Gen-3dffusion, Xie et al.

SyncDreamer, Liu et al. ICLR 24

- **Goal:** Given an image $\mathbf{y}$, we want to generate $\{\mathbf{x}_0^{(1)}, ..., \mathbf{x}_0^{(N)}\}$ novel views by learning the joint distribution $p_\theta(\mathbf{x}_0^{(1:N)}|\mathbf{y}) := p_\theta(\mathbf{x}_0^{(1)}, ..., \mathbf{x}_0^{(N)}|\mathbf{y})$
- **Key Idea**: Condition a novel view with images generated from other views.



SyncDreamer, Liu et al. ICLR 24

- **Forward Process**: is a direct extension of the vanilla DDPM, where noises are added to every view independently.

$$q(\mathbf{x}_{1:T}^{(1:N)}|\mathbf{x}_0^{(1:N)}) = \prod_{t=1}^{T} q(\mathbf{x}_t^{(1:N)}|\mathbf{x}_{t-1}^{(1:N)}) = \prod_{t=1}^{T}\prod_{n=1}^{N} q(\mathbf{x}_t^{(n)}|\mathbf{x}_{t-1}^{(n)})$$

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

SyncDreamer, Liu et al. ICLR 24

- **Backward Process**: Similarly, it is constructed as

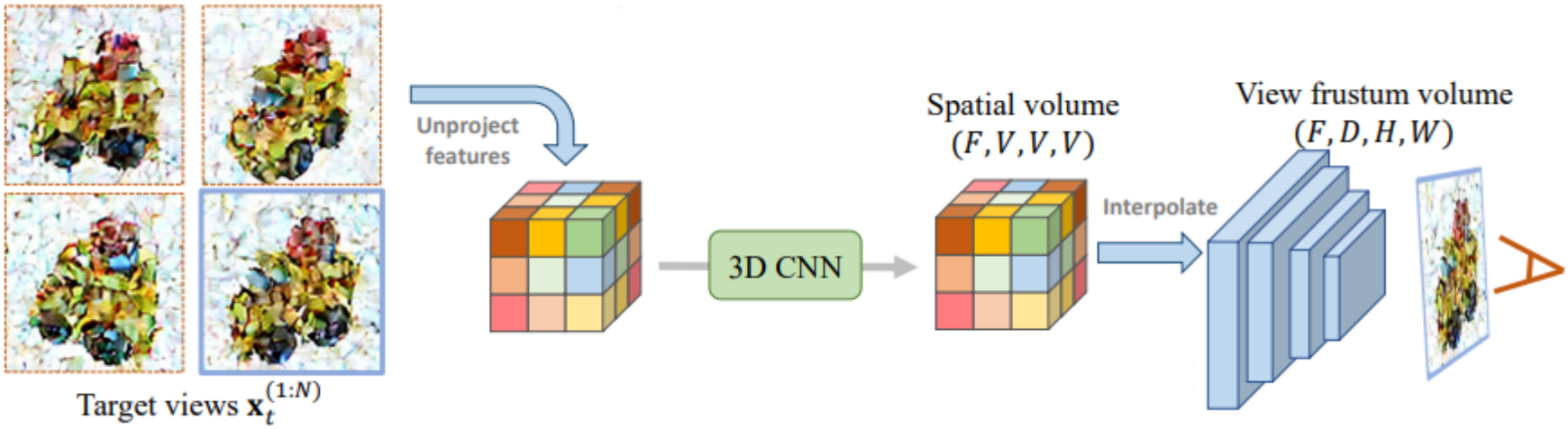
$$p_\theta(\mathbf{x}_{0:T}^{(1:N)}) = p(\mathbf{x}_T^{(1:N)}) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}^{(1:N)} | \mathbf{x}_t^{(1:N)}) = p(\mathbf{x}_T^{(1:N)}) \prod_{t=1}^{T} \prod_{n=1}^{N} p_\theta(\mathbf{x}_{t-1}^{(n)} | \mathbf{x}_t^{(1:N)}),$$

$$\text{where } p_\theta(\mathbf{x}_{t-1}^{(n)} | \mathbf{x}_t^{(1:N)}) = \mathcal{N}(\mathbf{x}_{t-1}^{(n)}; \mu_\theta^{(n)}(\mathbf{x}_t^{(1:N)}, t), \sigma_t^2 \mathbf{I})$$

$$\mu_\theta^{(n)}(\mathbf{x}_t^{(1:N)}, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t^{(n)} - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta^{(n)}(\mathbf{x}_t^{(1:N)}, t) \right).$$

SyncDreamer, Liu et al. ICLR 24

- **Backward Process**:
  - To condition the generation of a view on all the other N views at the same time, they project features in 3D, apply convolutions, and then construct a view frustum by interpolating features.



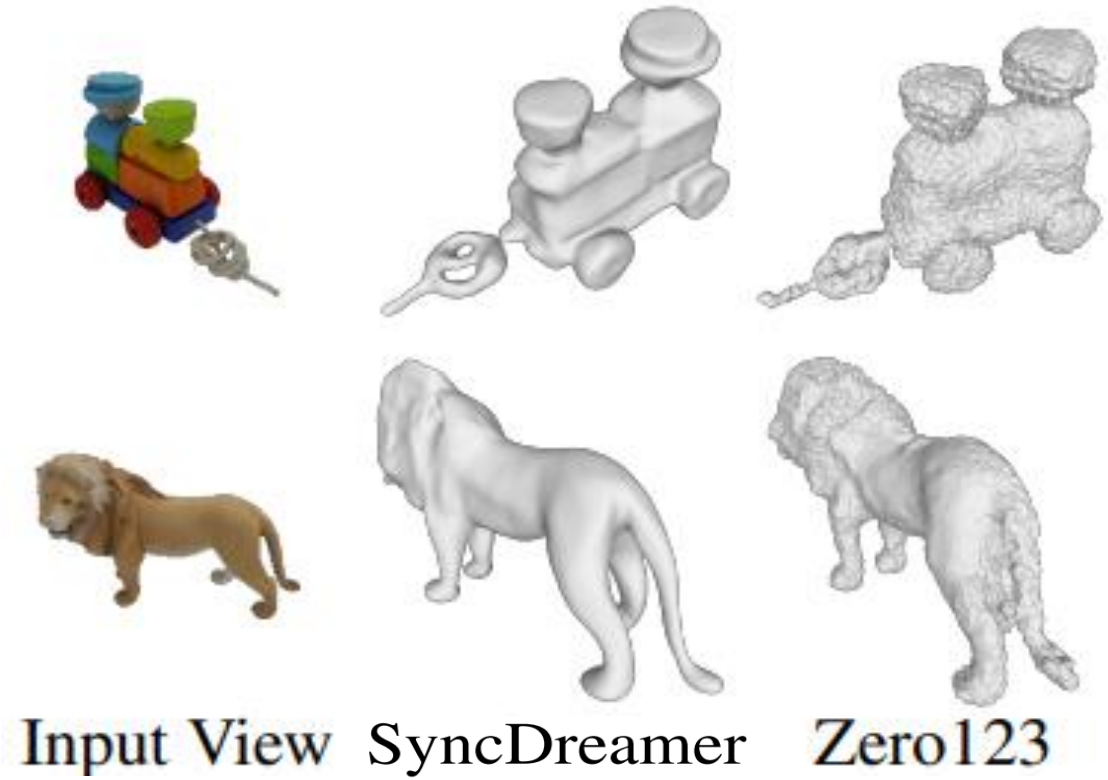
SyncDreamer, Liu et al. ICLR 24

- **Loss**:

$$\ell = \mathbb{E}_{t,\mathbf{x}_0^{(1:N)},n,\epsilon^{(1:N)}}\left[\|\epsilon^{(n)} - \epsilon_\theta^{(n)}(\mathbf{x}_t^{(1:N)}, t)\|_2\right]$$

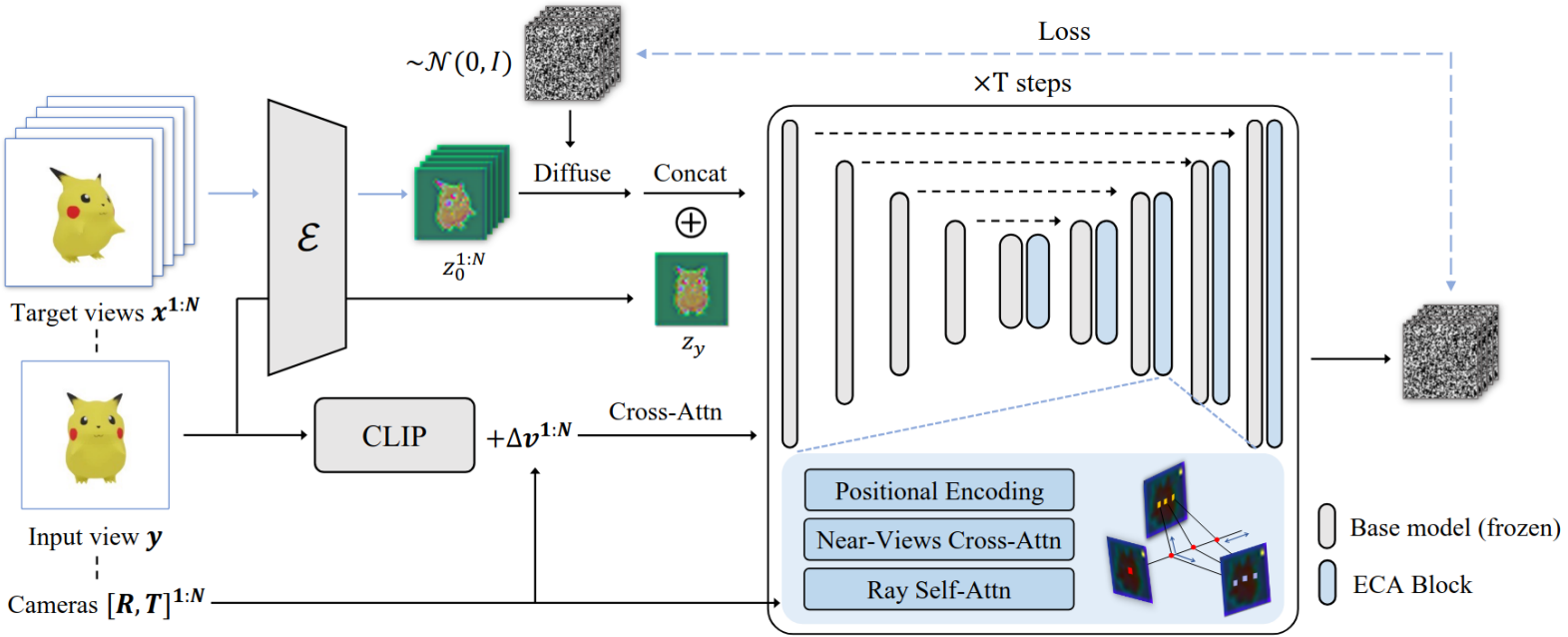
SyncDreamer, Liu et al. ICLR 24

- **Better 3D reconstructions**



EpiDiff, Huang et al. CVPR 24

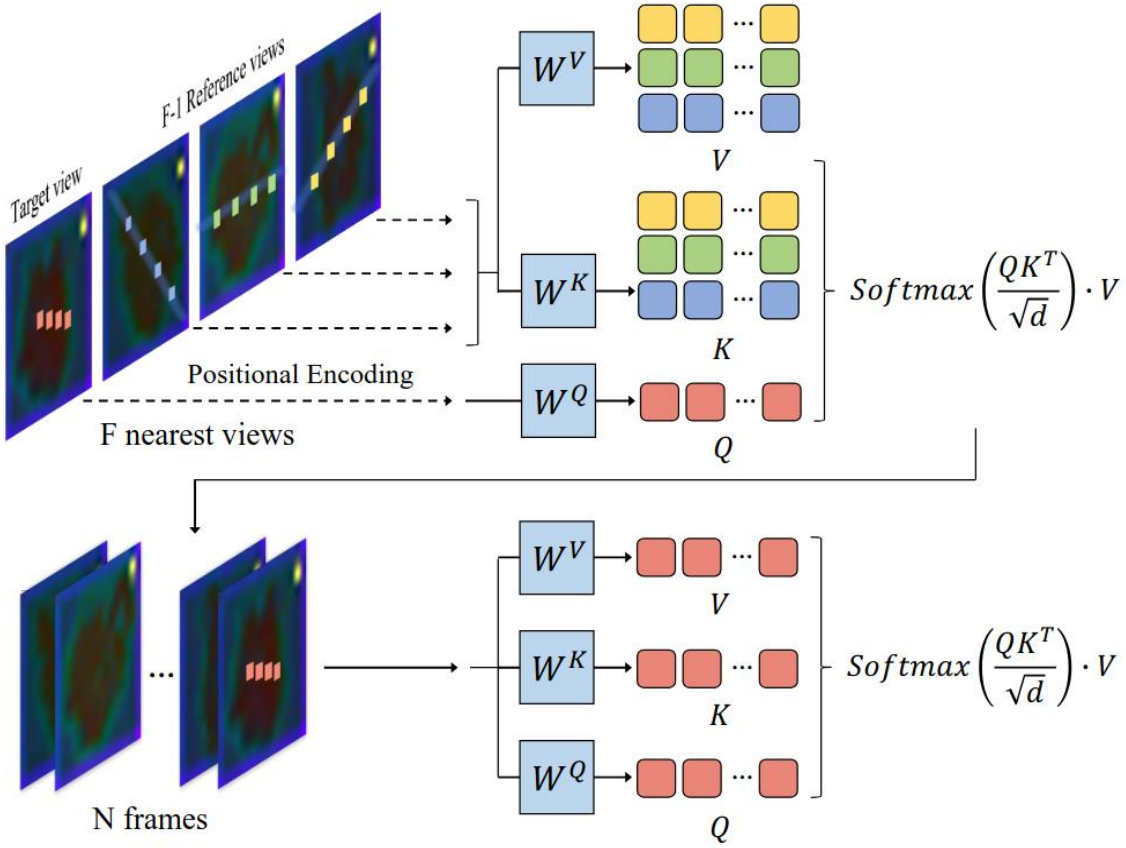
- **Key Idea:** Exploit Depth-aware 3D attention and Epipolar constraints to ensure depth consistency across different novel views



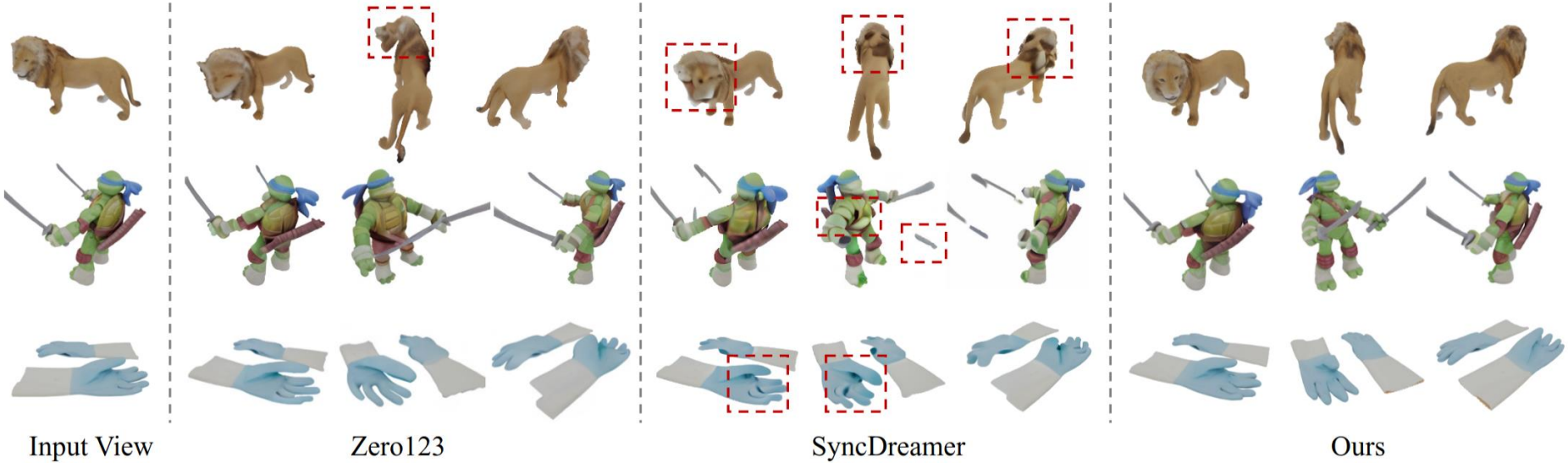
EpiDiff, Huang et al. CVPR 24

1) Cross attention with neighboring epipolar lines

2) Self attention for spatial consistency

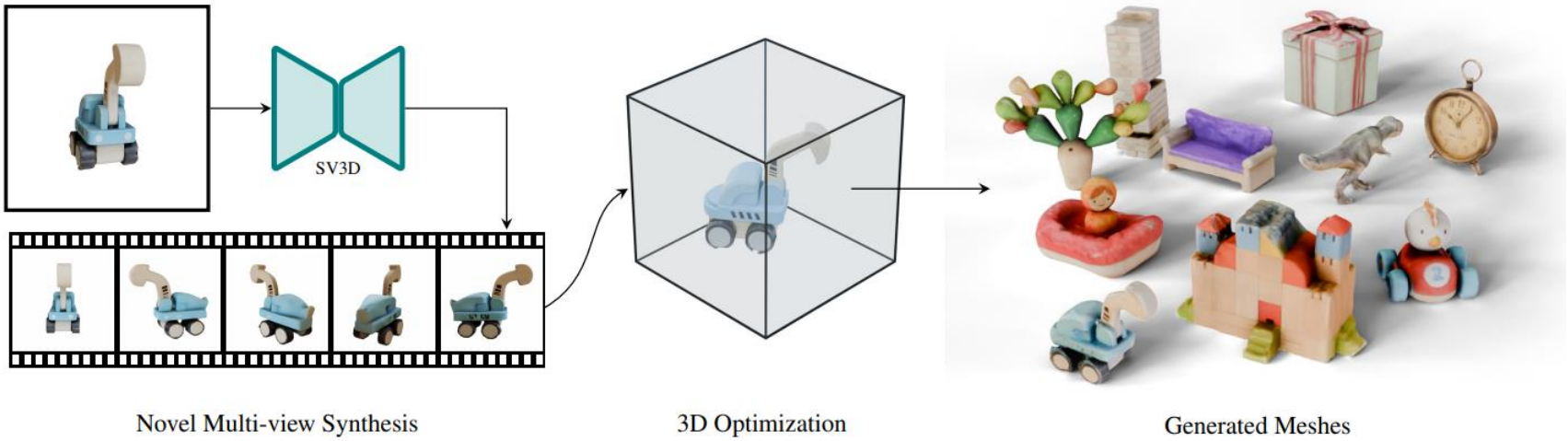


EpiDiff, Huang et al. CVPR 24



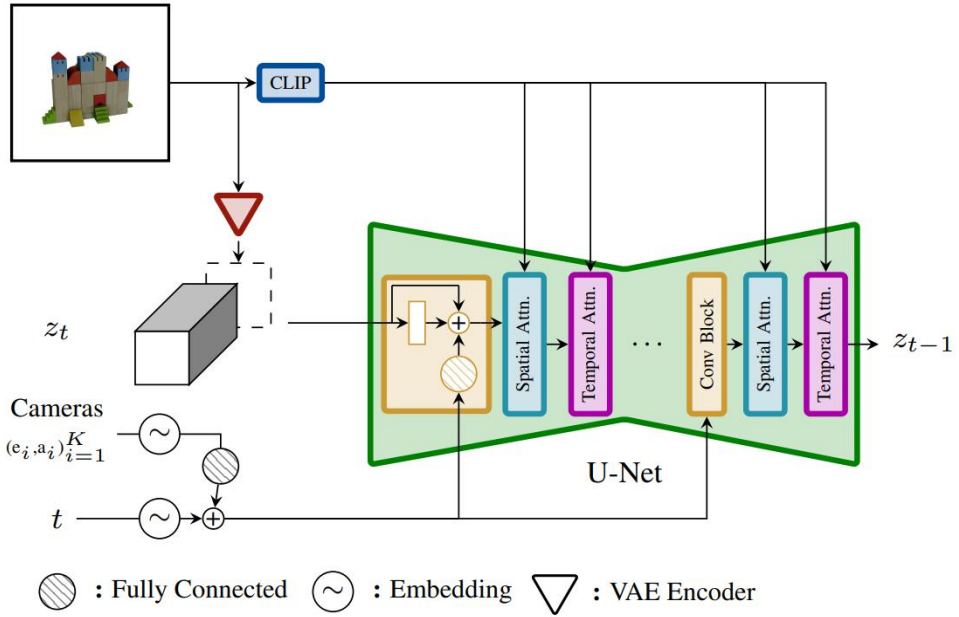
SV3D, Voleti et al. CVPR 24

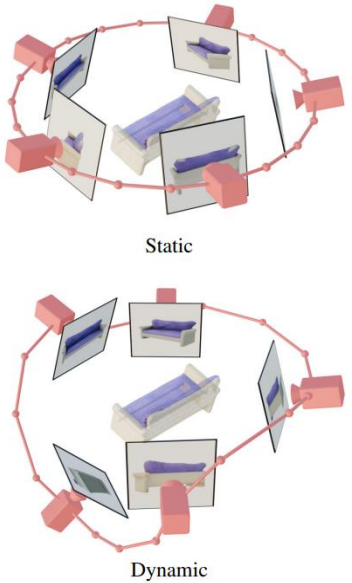
- **Key Idea:** Finetune **Stable Video Diffusion** on videos of rotating object. Condition on elevation and azimuth angles to control the camera movement.



SV3D, Voleti et al. CVPR 24

- **Conditioning:** to control the camera movement, they condition on camera positions.





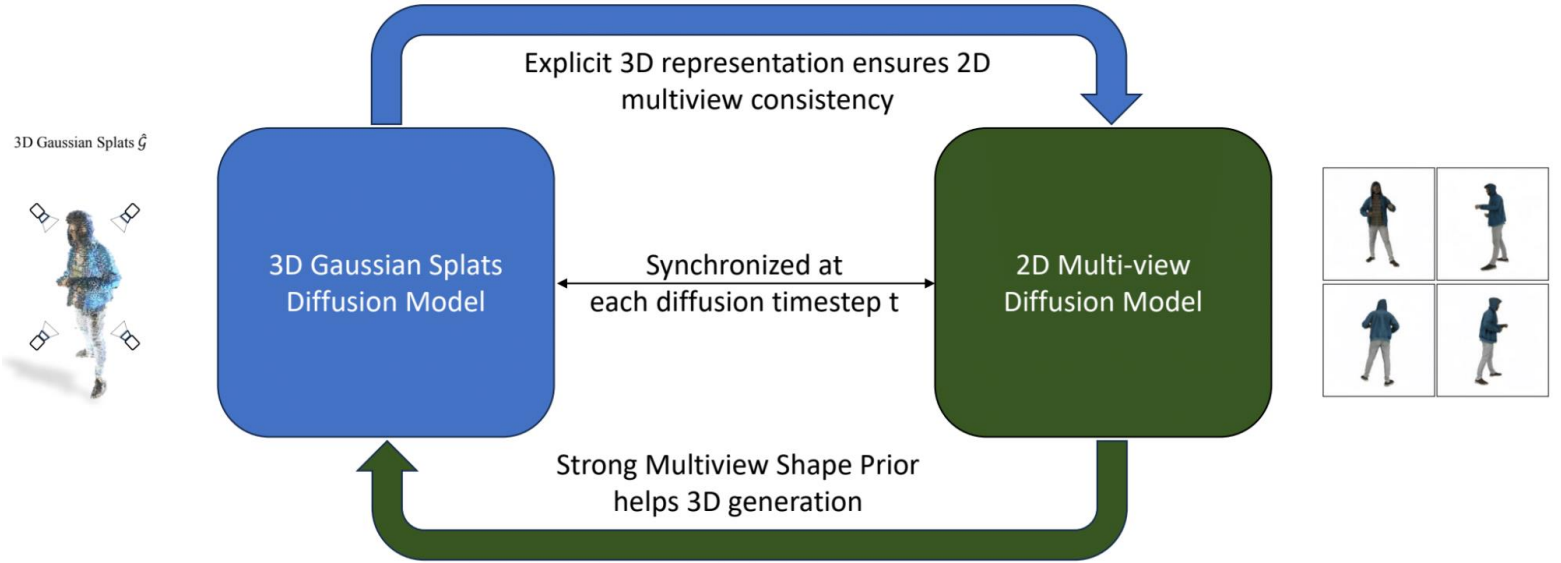
SV3D, Voleti et al. CVPR 24

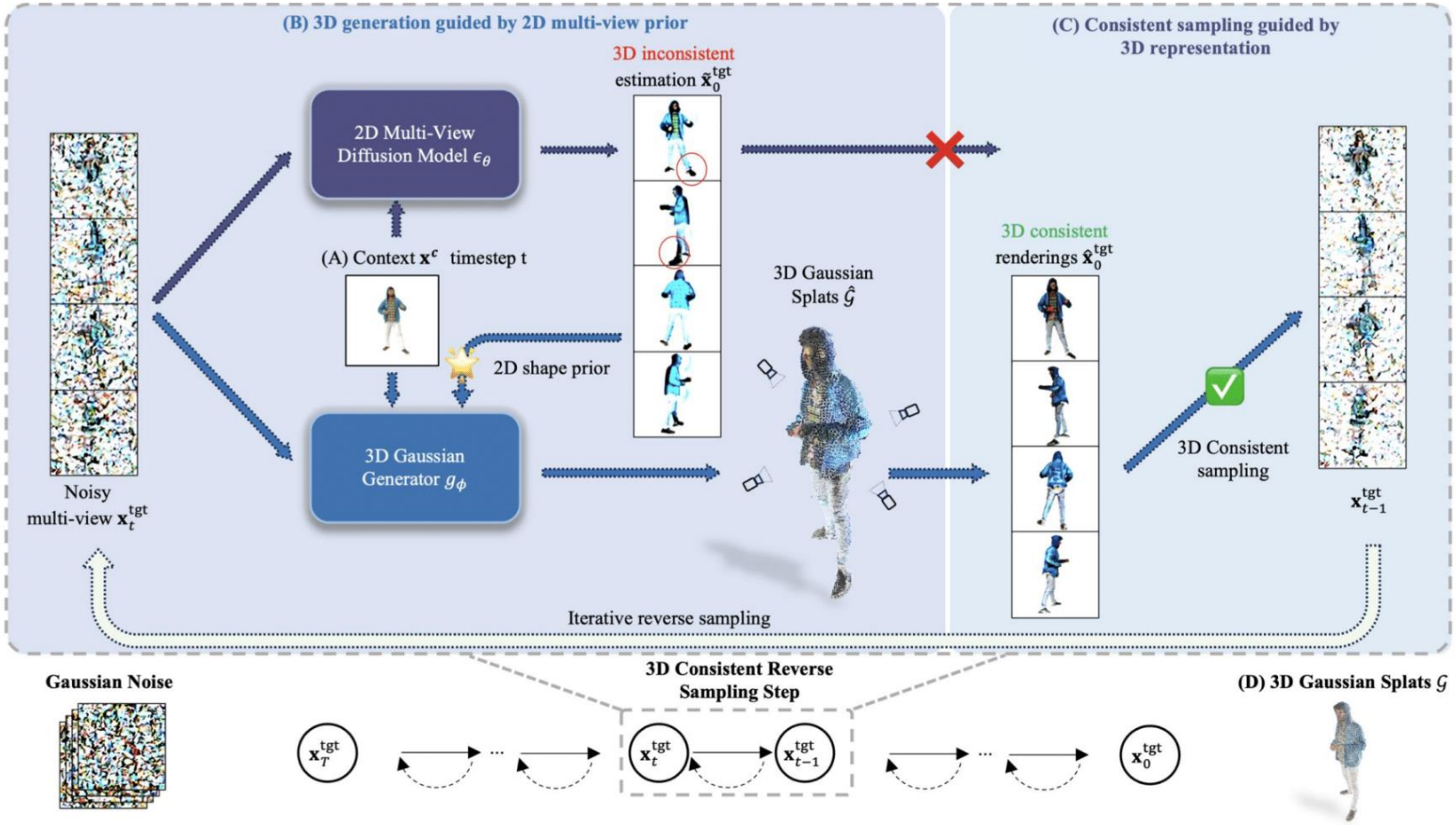
- **Consistency:** SV3D is based on Stable Video Diffusion (SVD), which already implements some techniques to improve consistency across the generated video.
    - SVD is a variant of Stable Diffusion adapted for video generation.
    - It reaches temporal coherence by adding **temporal attention and 3D convolutions** to Stable Diffusion's U-Net.

SV3D, Voleti et al. CVPR 24

Gen-3Diffusion: Sync 2D Diffusion & 3D Recon

- **Key Idea:** Synchronizing 2D Multiview Diffusion and 3D diffusion-based Generative models




(B) 3D generation guided by 2D multi-view prior
(C) Consistent sampling guided by 3D representation
3D inconsistent estimation $\hat{\mathbf{x}}_0^{\text{tgt}}$
2D Multi-View Diffusion Model ϵ_θ
(A) Context $\mathbf{x}^c$ timestep t
2D shape prior
3D Gaussian Generator g_ϕ
Noisy multi-view $\mathbf{x}_t^{\text{tgt}}$
3D Gaussian Splats $\hat{\mathcal{G}}$
3D consistent renderings $\hat{\mathbf{x}}_0^{\text{tgt}}$
3D Consistent sampling
$\mathbf{x}_{t-1}^{\text{tgt}}$
Iterative reverse sampling
Gaussian Noise
3D Consistent Reverse Sampling Step
$\mathbf{x}_T^{\text{tgt}}$
$\mathbf{x}_t^{\text{tgt}}$
$\mathbf{x}_{t-1}^{\text{tgt}}$
$\mathbf{x}_0^{\text{tgt}}$
(D) 3D Gaussian Splats $\mathcal{G}$

Algorithm

Algorithm 1 Joint 2D & 3D Diffusion Training

Input: Dataset of posed multi-view images $\mathbf{x}_0^{\text{tgt}}$, π^{tgt}, $\mathbf{x}_0^{\text{novel}}$, π^{novel}, a context image $\mathbf{x}^{\text{c}}$, text description y

Output: Optimized 2D multi-view diffusion model ϵ_θ and 3D-GS generative model g_ϕ

repeat

$\{\mathbf{x}_0^{\text{tgt}}, \mathbf{x}_0^{\text{novel}}, \mathbf{x}^{\text{c}}, y\} \sim q(\{\mathbf{x}_0^{\text{tgt}}, \mathbf{x}_0^{\text{novel}}, \mathbf{x}^{\text{c}}, y\})$

$t \sim \text{Uniform}(\{1, \dots, T\})$; $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

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

$\tilde{\mathbf{x}}_0^{\text{tgt}} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t^{\text{tgt}} - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^{\text{tgt}}, \mathbf{x}^{\text{c}}, y, t))$

$\hat{\mathcal{G}} = g_\phi\left(\mathbf{x}_t^{\text{tgt}}, t, \mathbf{x}^{\text{c}}, \tilde{\mathbf{x}}_0^{\text{tgt}}\right)$ // Enhance conditional 3D generation with 2D diffusion prior $\tilde{\mathbf{x}}_0^{\text{tgt}}$ from ϵ_θ

$\{\hat{\mathbf{x}}_0^{\text{tgt}}, \hat{\mathbf{x}}_0^{\text{novel}}\} = \texttt{renderer}\left(\hat{\mathcal{G}}, \{\pi^{\text{tgt}}, \pi^{\text{novel}}\}\right)$

Compute loss $\mathcal{L}_{total}$ (Eq. (9))

Gradient step to update ϵ_θ, g_ϕ

until converged

Algorithm 2 3D Consistent Guided Sampling

Input: A context image $\mathbf{x}^c$ and text y; Converged 2D diffusion model ϵ_θ and 3D generative model g_ϕ

Output: 3D Gaussian Splats $\mathcal{G}$ of the 2D image $\mathbf{x}^c$

$\mathbf{x}_T^{\text{tgt}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

for $t = T, \dots, 1$ **do**

$\tilde{\mathbf{x}}_0^{\text{tgt}} = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t^{\text{tgt}} - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t^{\text{tgt}}, \mathbf{x}^{\text{c}}, y, t))$

$\hat{\mathcal{G}} = g_\phi\left(\mathbf{x}_t^{\text{tgt}}, t, \mathbf{x}^{\text{c}}, \tilde{\mathbf{x}}_0^{\text{tgt}}\right)$

$\hat{\mathbf{x}}_0^{\text{tgt}} = \texttt{renderer}\left(\hat{\mathcal{G}}, \pi^{\text{tgt}}\right)$

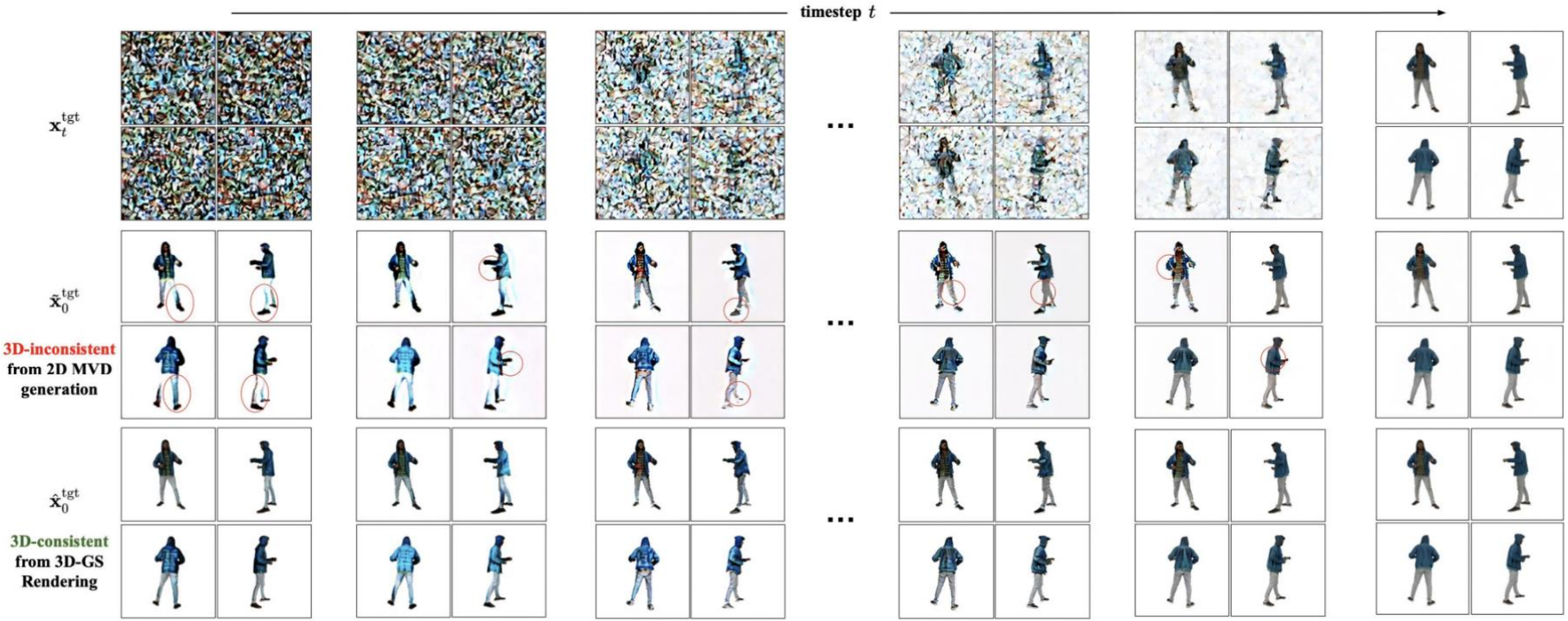
$\mu_{t-1}(\mathbf{x}_t^{\text{tgt}}, \hat{\mathbf{x}}_0^{\text{tgt}}) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t^{\text{tgt}} + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0^{\text{tgt}}$ // Guide 2D sampling with 3D consistent multi-view renderings

$\mathbf{x}_{t-1}^{\text{tgt}} \sim \mathcal{N}\left(\mathbf{x}_{t-1}^{\text{tgt}}; \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t^{\text{tgt}}, \hat{\mathbf{x}}_0^{\text{tgt}}\right), \tilde{\beta}_{t-1}\mathbf{I}\right)$

end for

return $\mathcal{G} = g_\phi\left(\mathbf{x}_0^{\text{tgt}}, \tilde{\mathbf{x}}_0^{\text{tgt}}, \mathbf{x}^{\text{c}}, t = 0\right)$

Explicit 3DGS helps 2D Diffusion



Reconstruction avatar appearance

Input Image

Gen-3Diffusion

SiTH

SiFU

Reconstruction avatar geometry

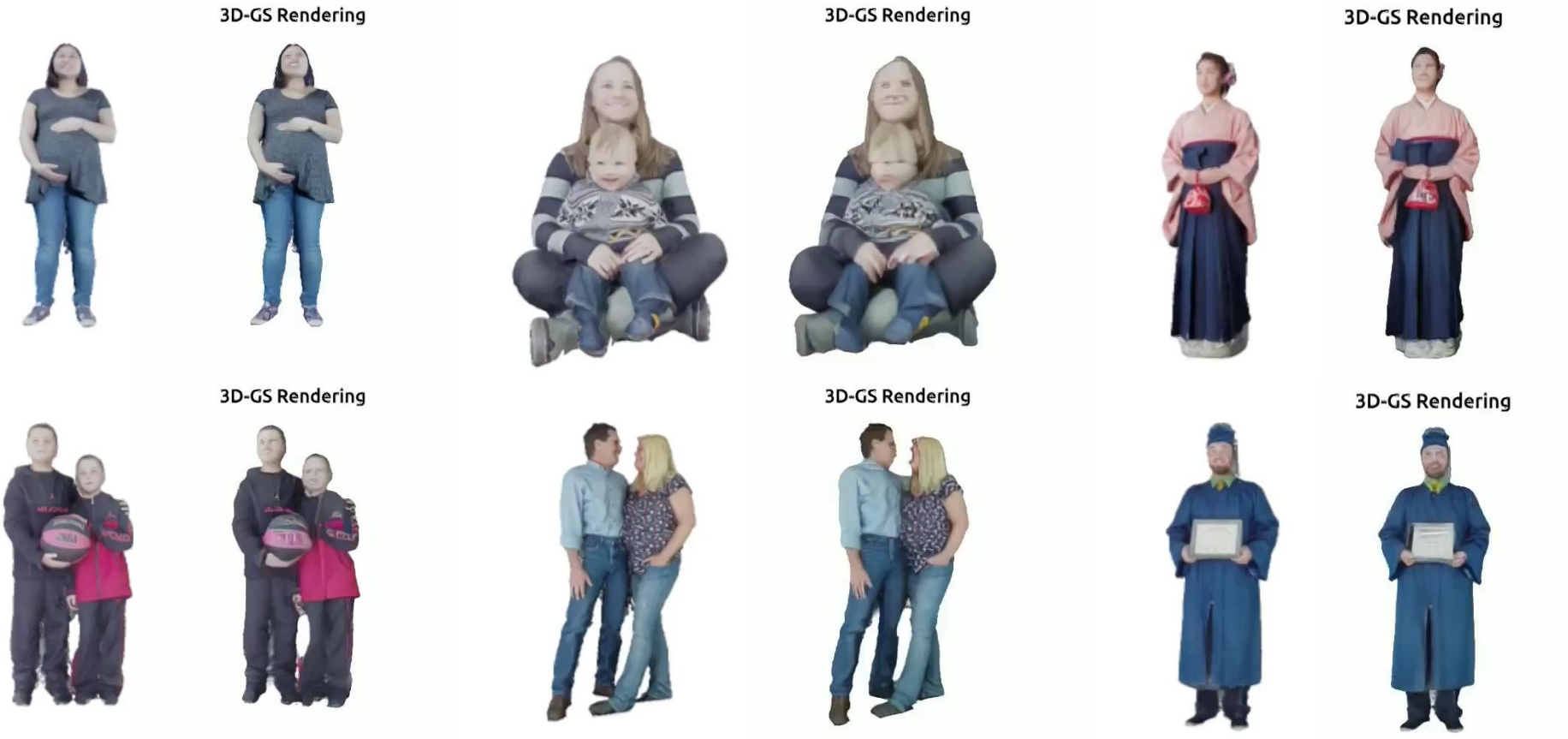
Input Image

Gen-3Diffusion

ICON

ECON

Strong generalization



Limitations of Multiview Diffusion

Multiview Diffusion and Ground Truth

- As we said at the beginning of the lecture, reconstructing an object from a single image is an **ill-posed** problem.
- One of the main limitations of multiview diffusion is that benchmarks are based on Ground Truth data
  - Problem: What is Ground Truth in a Generative task?

Let's play a game. Which is the GT?

Input View

180° Generated Views

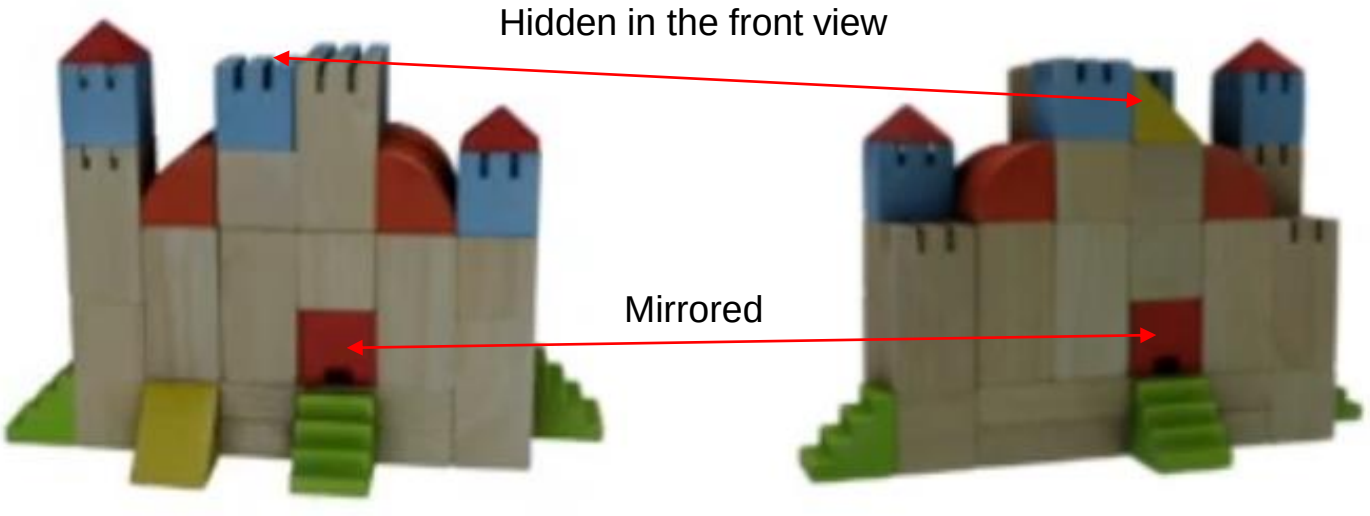
Let's play a game. Which is the GT?

Input View

Generated Views

GT

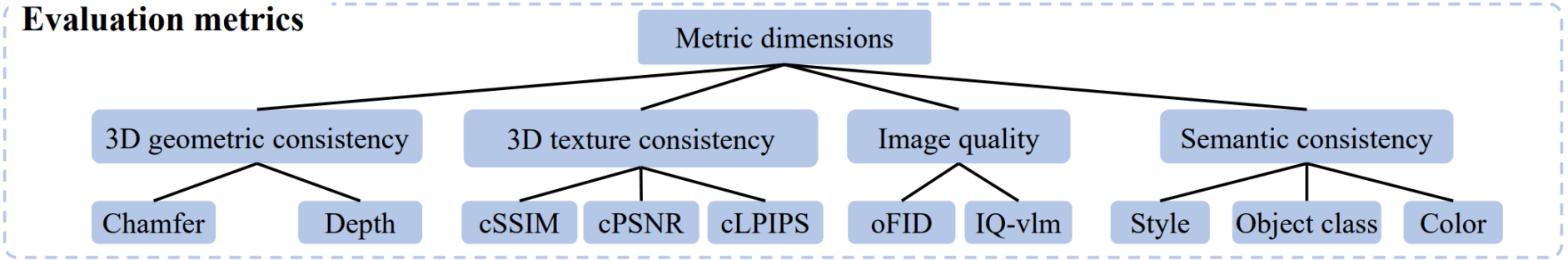
Let's play a game. Which is the GT?



It does not make sense to compare a generated view to a fixed GT view.

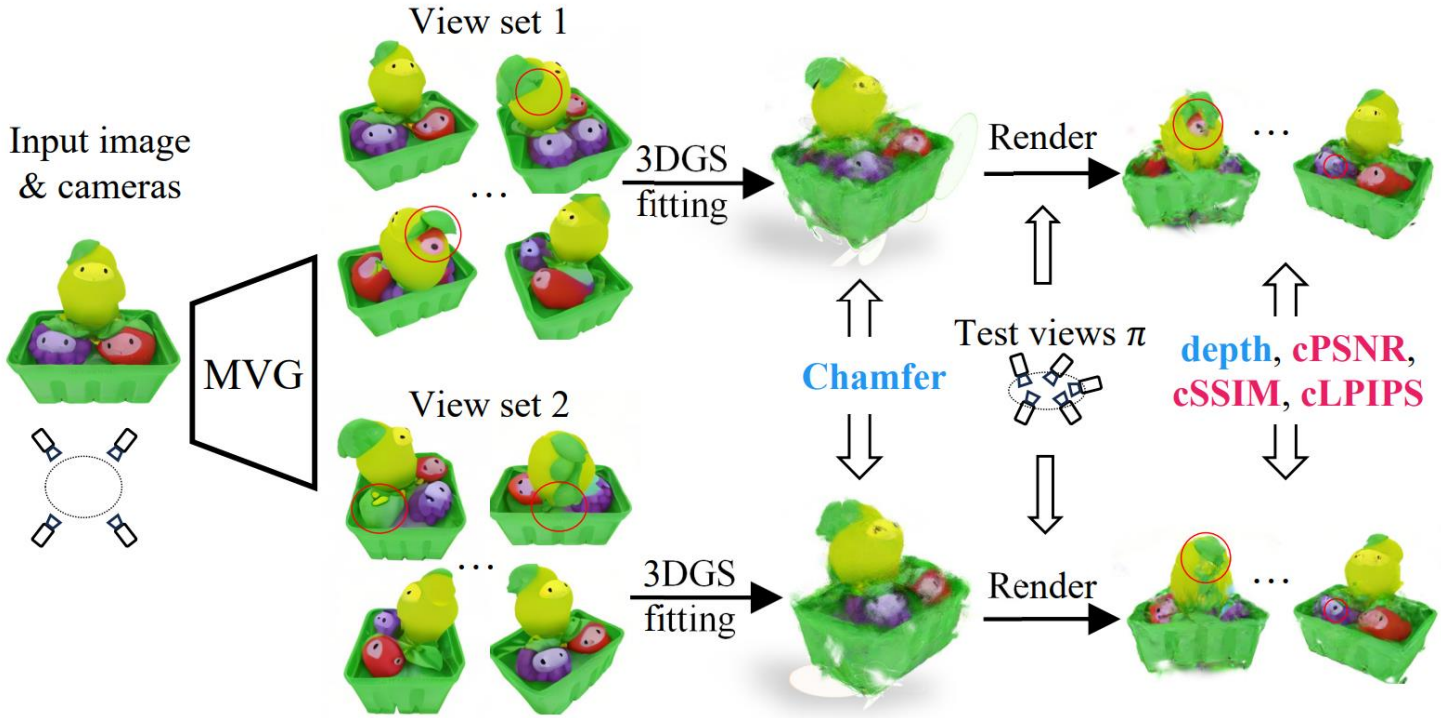
MVGBench, Xie et al.

- Comprehensive benchmark that evaluates
  - 3D **geometric** consistency
  - 3D **texture** consistency
  - Image **Quality**
  - **Semantic** Consistency



MVGBench, Xie et al.

- 3D **geometric** and **texture** consistency



MVGBench, Xie et al.

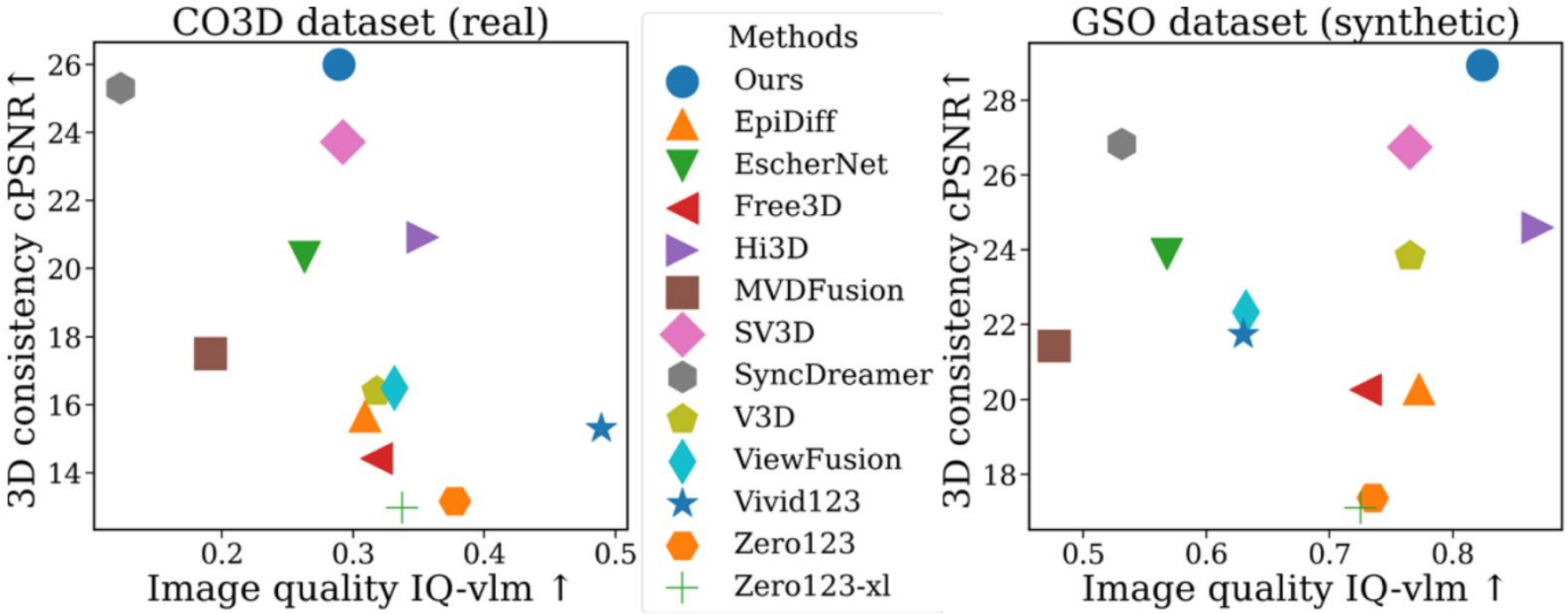
- **Semantic** Metrics
  - **FID score**

$$FID(r, g) = \left\|\mu_r - \mu_g\right\|_2^2 + Tr\left(\sum_r + \sum_g - 2\sqrt{\sum_r \sum_g}\right),$$

where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean and covariance of the real data and model distributions, respectively.

  - **Pretrained VLM** for question answering on the overall quality of the generated view
- **Image Quality** Metrics
  - **Pretrained VLM** to assess
    - **Class** consistency
    - **Color** consistency
    - **Style** consistency

MVGBench, Xie et al.



Conclusion

Conclusion

- In this lecture we saw
  - how Diffusion Models work
  - How can we leverage diffusion models to generate multiview images
  - We saw some limitations of Multiview Diffusion.
- Next Lecture:
  - Multiview Diffusion is not the only way to exploit Diffusion for 3D
    - Score Distillation Sampling
  - Methods that do not leverage on Image Diffusion
    - PointDiffusion
    - Hunyuan3D 2.0
    - Trellis

The End