

Fall 2023

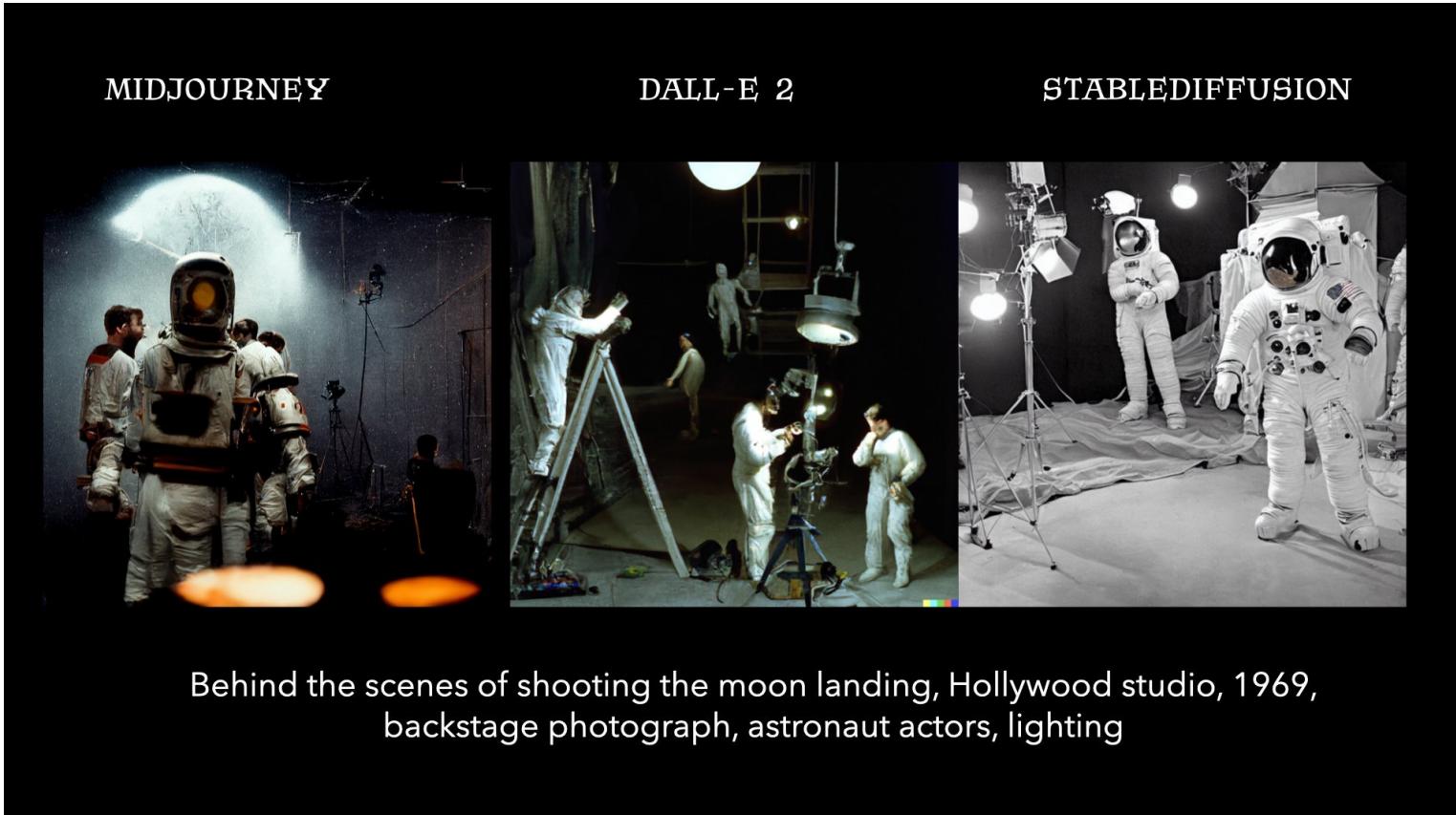
ADVANCED TOPICS IN COMPUTER VISION

Atlas Wang

Associate Professor, The University of Texas at Austin

Visual Informatics Group@UT Austin
<https://vita-group.github.io/>

\$\$\$\$\$ market: Text2Image (video, 3D...)!



This part slides were **heavily borrowed from** <https://cvpr2022-tutorial-diffusion-models.github.io/> and <https://cvpr2023-tutorial-diffusion-models.github.io> . **THANK YOU!**

The New York Times

IT HAPPENED ONLINE

How Is Everyone Making Those A.I. Selfies?

Images generated with Lensa AI are all over social media, but at what cost?

Give this article Share Bookmark



Lensa AI, a popular iPhone app, uses your selfies and artificial intelligence to create portraits in a variety of styles. Lensa AI

DALL·E 2

“a teddy bear on a skateboard in times square”



[“Hierarchical Text-Conditional Image Generation with CLIP Latents”](#)
[Ramesh et al., 2022](#)

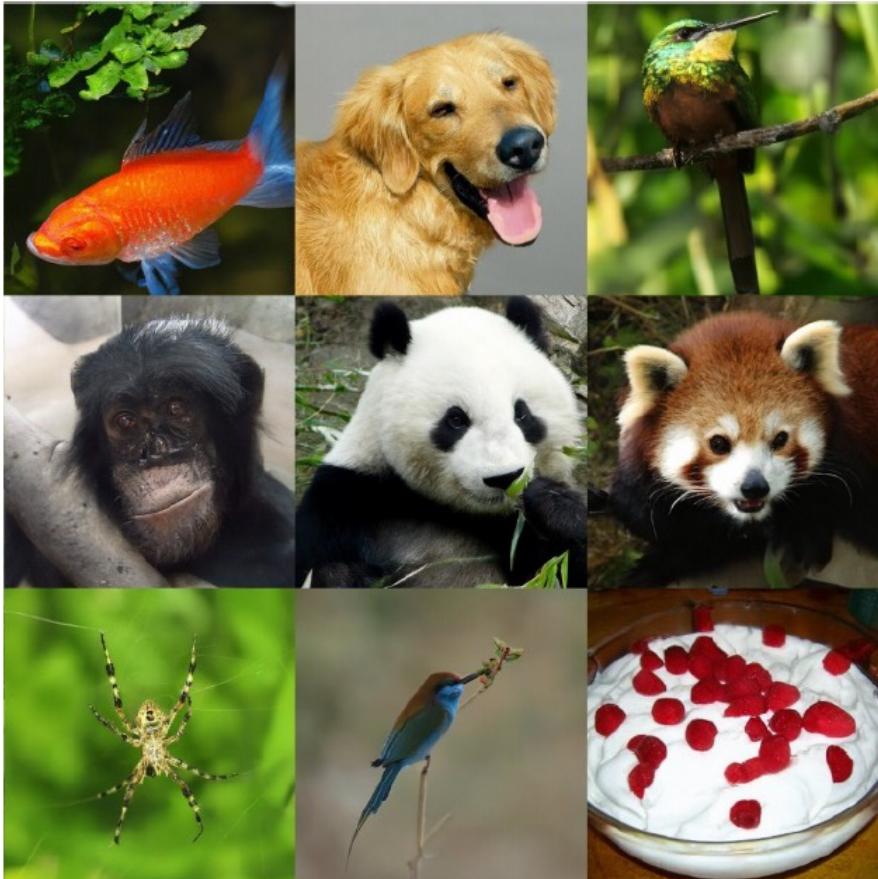
Imagen

A group of teddy bears in suit in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.



[“Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”, Saharia et al., 2022](#)

The Workhorse: *Diffusion Models*



["Diffusion Models Beat GANs on Image Synthesis"](#)
[Dhariwal & Nichol, OpenAI, 2021](#)

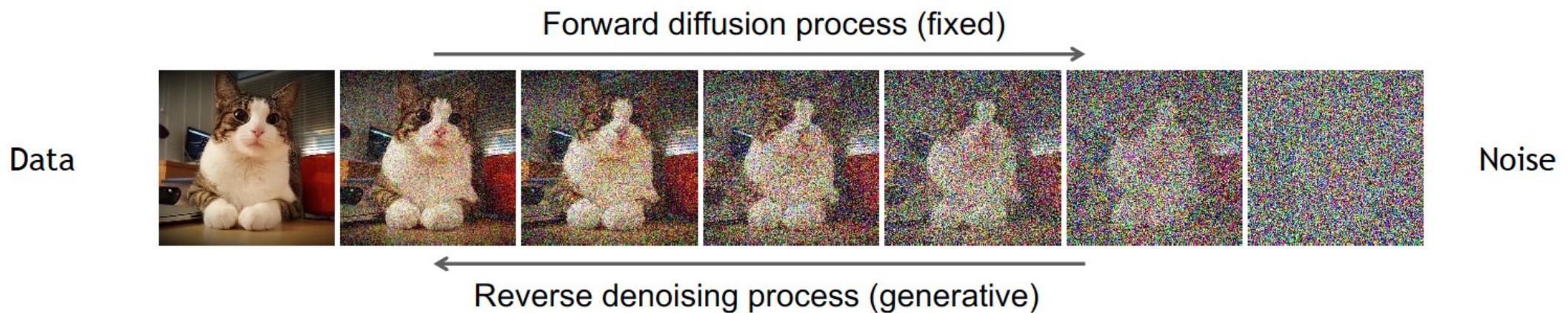


["Cascaded Diffusion Models for High Fidelity Image Generation"](#)
[Ho et al., Google, 2021](#)

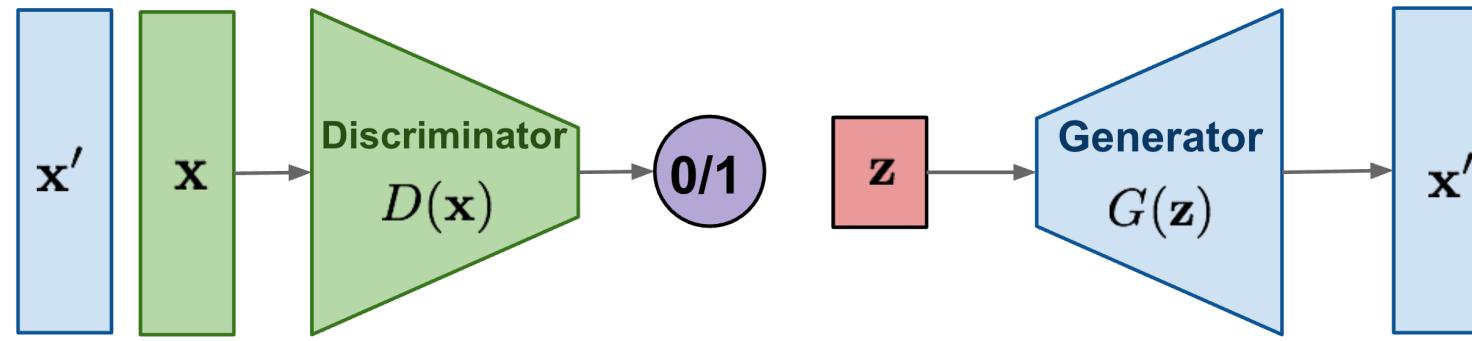
Learning to generate by denoising

Denoising diffusion models consist of two processes:

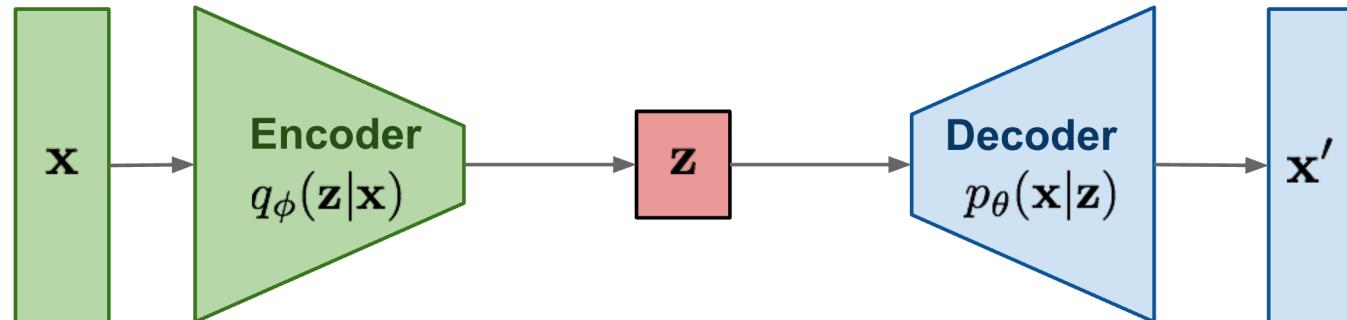
- Forward diffusion process that gradually adds noise to input
- Reverse denoising process that learns to generate data by denoising



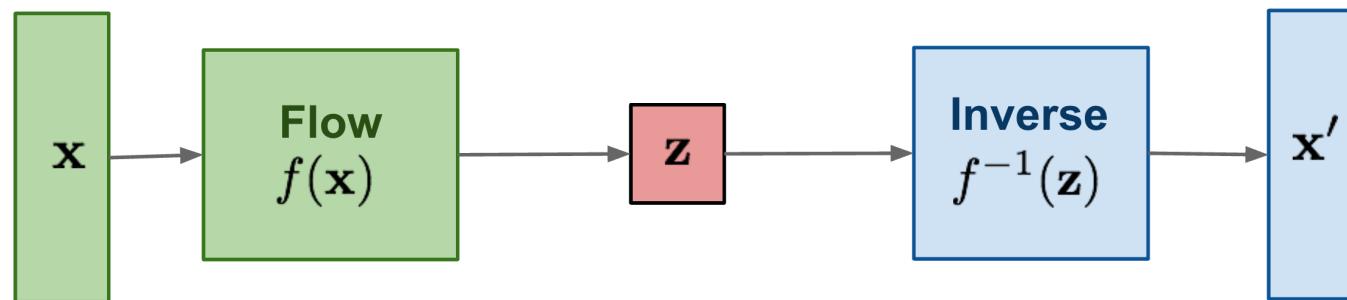
GAN: Adversarial training



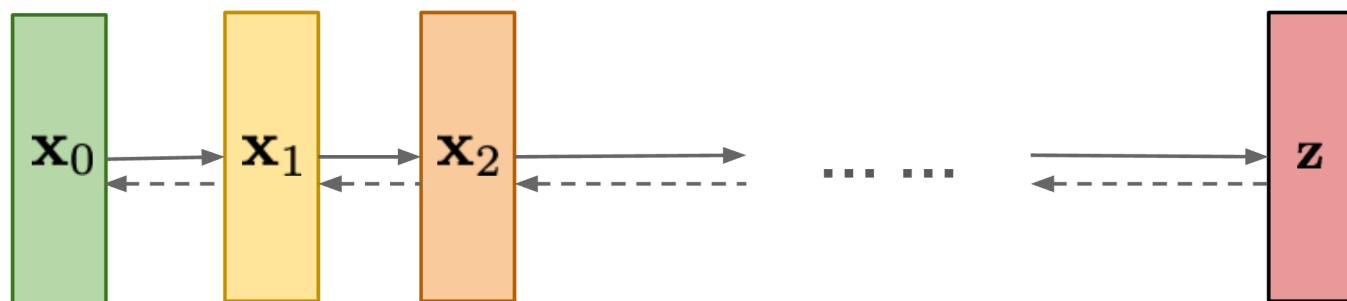
VAE: maximize variational lower bound



Flow-based models:
Invertible transform of distributions



Diffusion models:
Gradually add Gaussian noise and then reverse



Variational Autoencoders (VAEs)

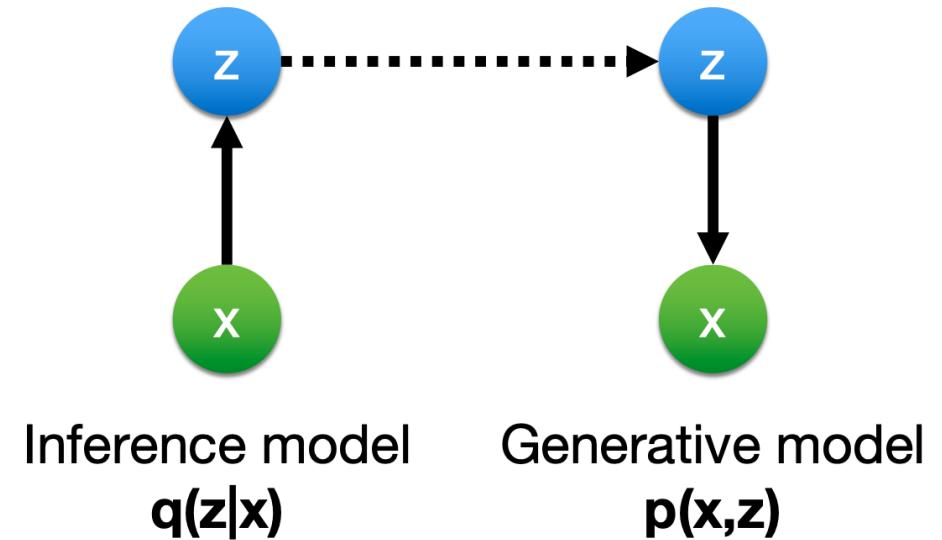
- We introduce an **inference model** $q(z|x)$

$$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \Sigma_\phi(x))$$

- This allows us to efficiently optimize the log-likelihood, through the **evidence lower bound** (ELBO).

$$\log p_{\theta,\phi}(x) \geq \text{ELBO}(x) = \mathbb{E}_{q_\phi(z|x)} \left[\log \frac{p_\theta(x, z)}{q_\phi(z|x)} \right]$$

- We optimize $q(z|x)$ and $p(x,z)$ jointly w.r.t. ELBO
- Bound is tight with the right $q(z|x)$



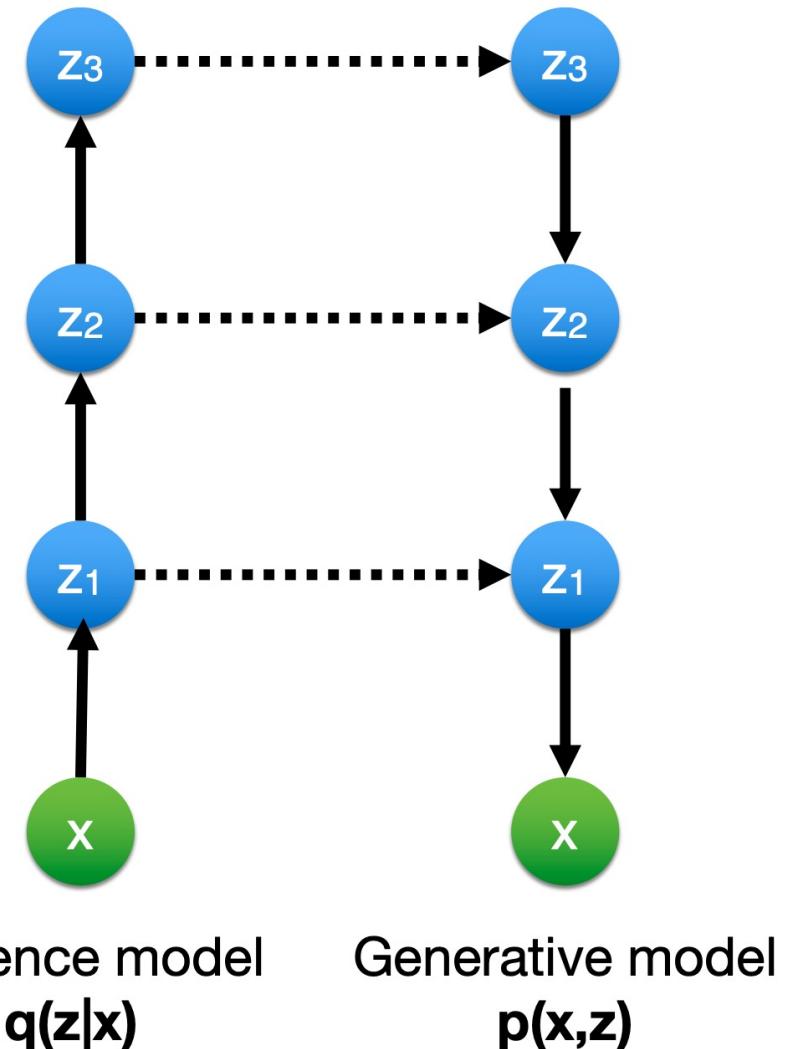
Hierarchical VAEs

- “Flat” VAEs suffer from simple priors
- Making both inference model and generative model hierarchical

$$q_\phi(\mathbf{z}_{1,2,3}|\mathbf{x}) = q_\phi(\mathbf{z}_1|\mathbf{x})q_\phi(\mathbf{z}_2|\mathbf{z}_1)q_\phi(\mathbf{z}_3|\mathbf{z}_2)$$

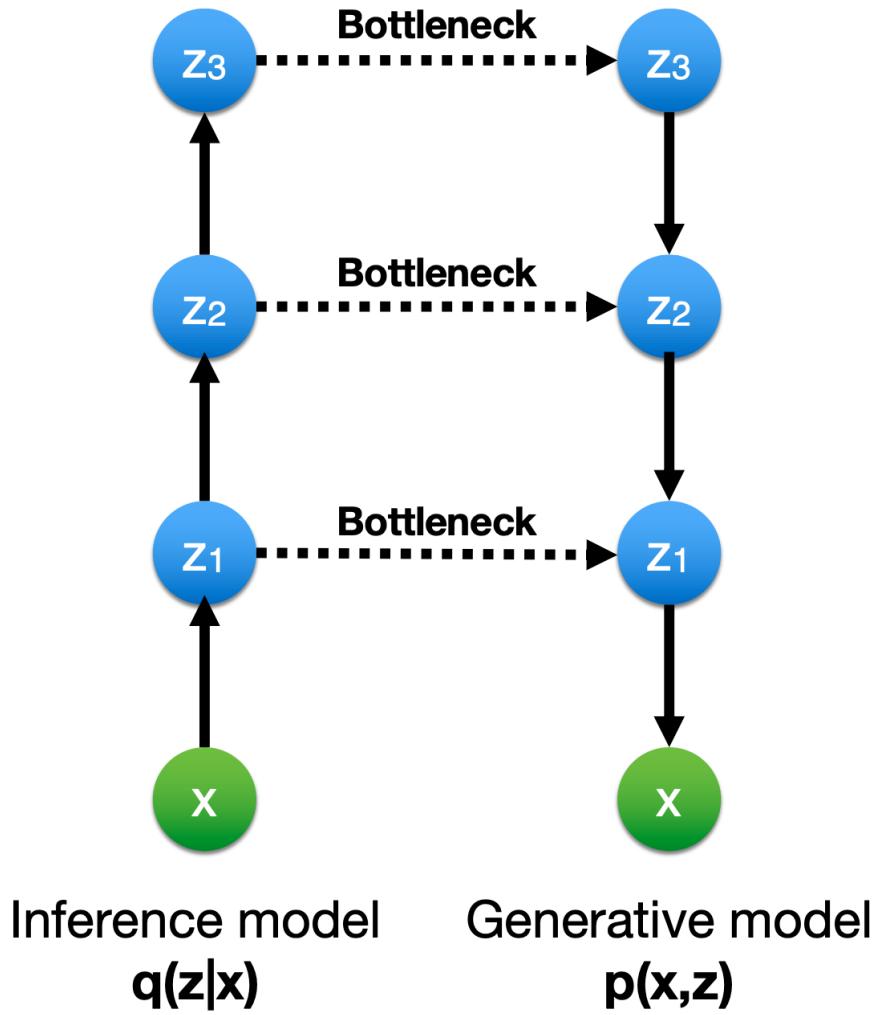
$$p_\theta(\mathbf{z}_{1,2,3}) = p_\theta(\mathbf{z}_3)p_\theta(\mathbf{z}_2|\mathbf{z}_3)p_\theta(\mathbf{z}_1|\mathbf{z}_2)p_\theta(\mathbf{x}|\mathbf{z}_1)$$

- Better likelihoods are achieved with hierarchies of latent variables



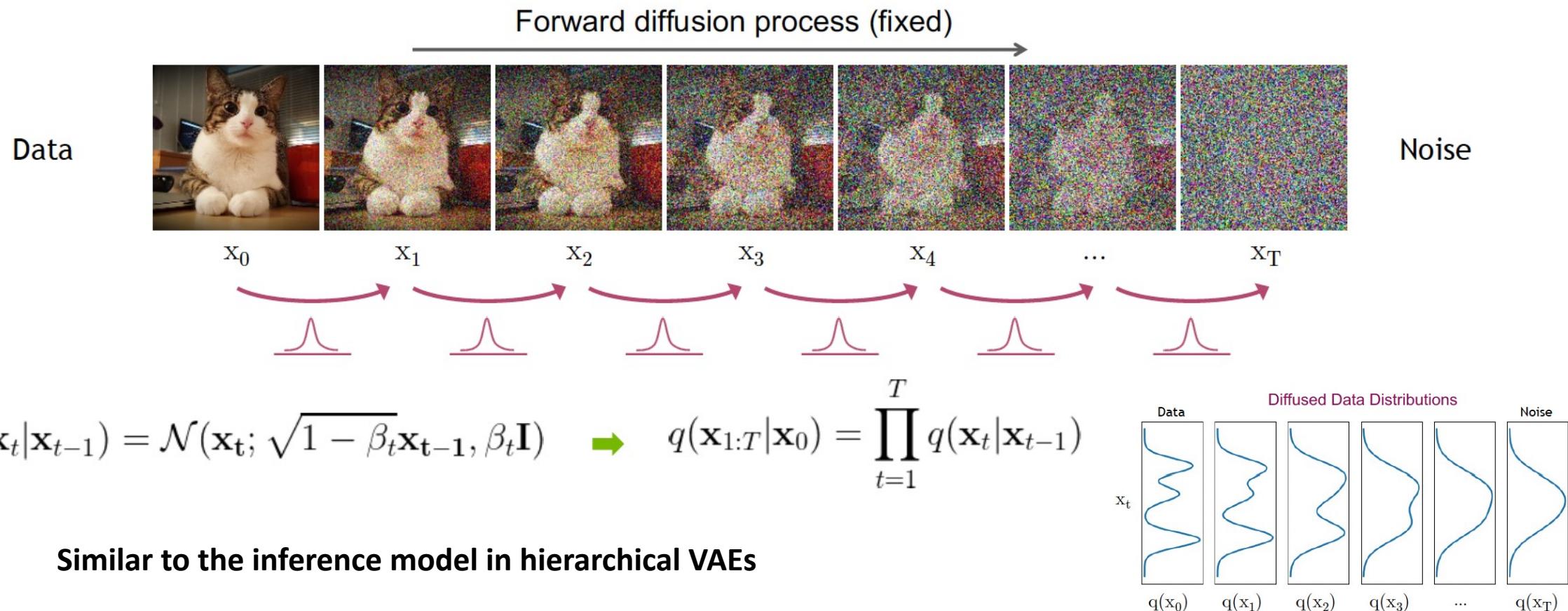
VAEs: challenges

- Optimization can be difficult for large models
- The ELBO enforces an **information bottleneck** (through its loss function) at the latent variables 'z', making VAE optimization prone to **bad local minima**.
- **Posterior collapse** is a dreaded bad local minimum where the latents do not transmit any information.

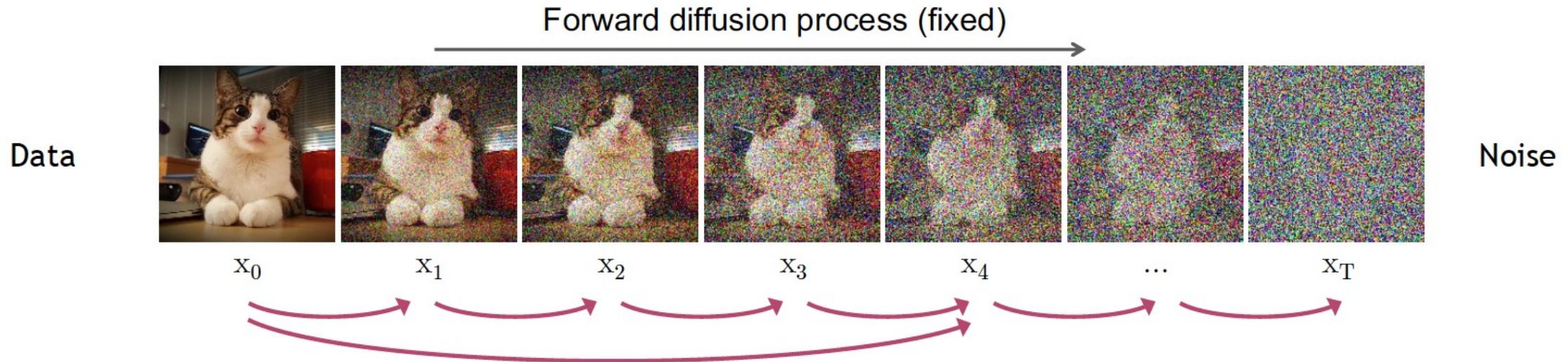


Forward Diffusion Process

The formal definition of the forward process in T steps:



Sampling at arbitrary time step with “reparameterization trick”



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

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

The diffusion kernel is Gaussian convolution.

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

Key trick: what happens if we add/merge two Gaussians?

If X_1 and X_2 are two independent normal random variables, with means μ_1, μ_2 and standard deviations σ_1, σ_2 , then their sum $X_1 + X_2$ will also be normally distributed, [proof] with mean $\mu_1 + \mu_2$ and variance $\sigma_1^2 + \sigma_2^2$

Generative Learning by Denoising

Recall, that the diffusion parameters are designed such that $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

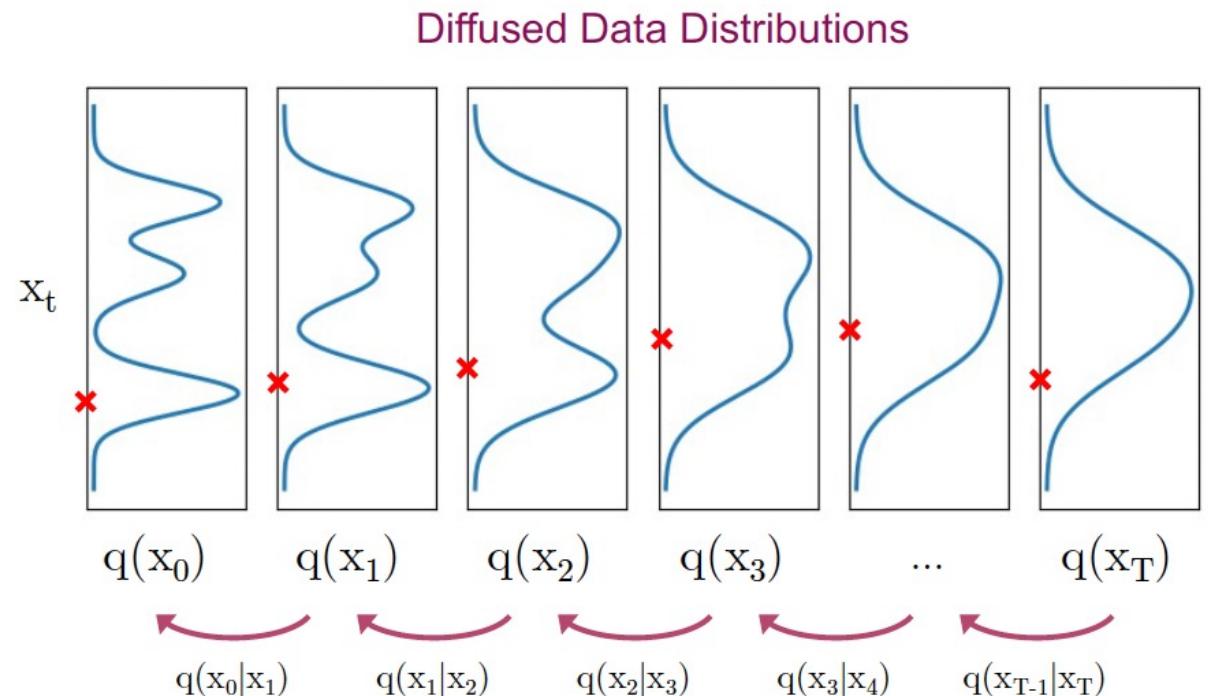
Generation:

Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1} | \mathbf{x}_t)}_{\text{True Denoising Dist.}}$

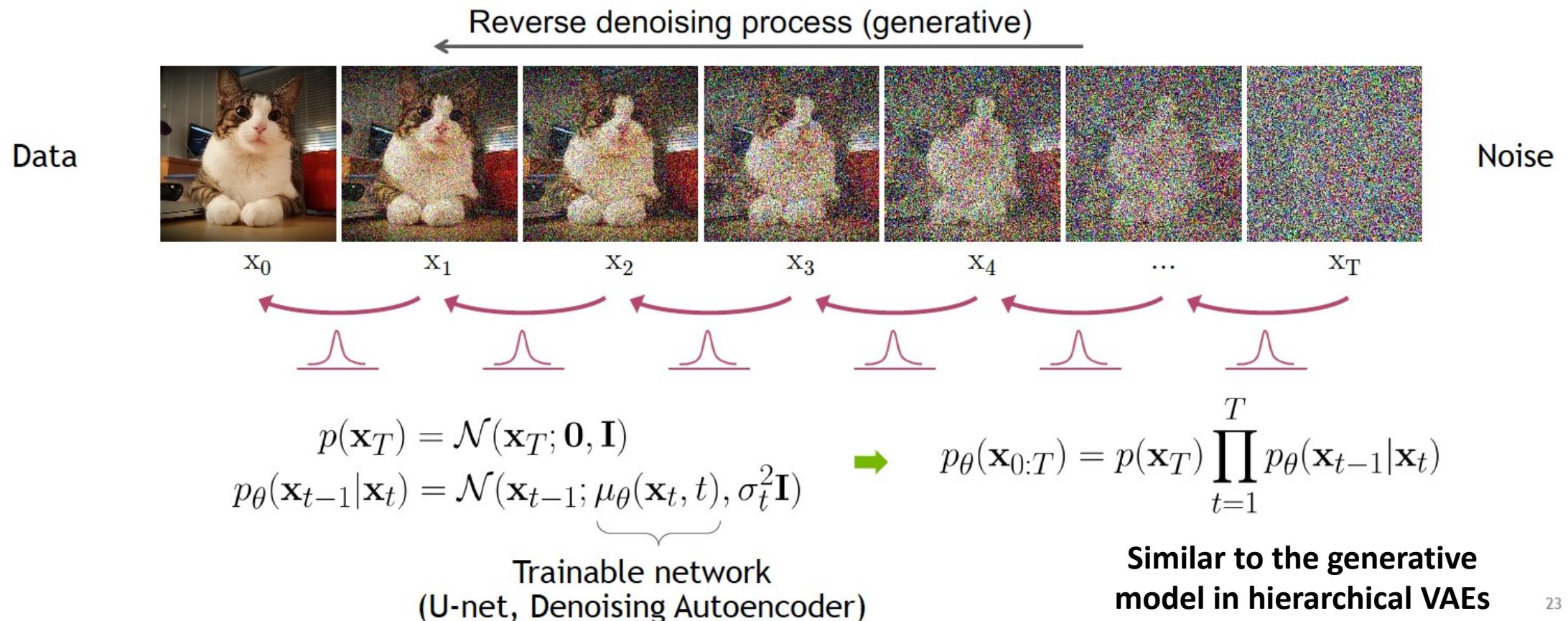
In general, $q(\mathbf{x}_{t-1} | \mathbf{x}_t) \propto q(\mathbf{x}_{t-1})q(\mathbf{x}_t | \mathbf{x}_{t-1})$ is intractable.

Can we approximate $q(\mathbf{x}_{t-1} | \mathbf{x}_t)$? Yes, we can use a **Normal distribution** if β_t is small in each forward diffusion step.



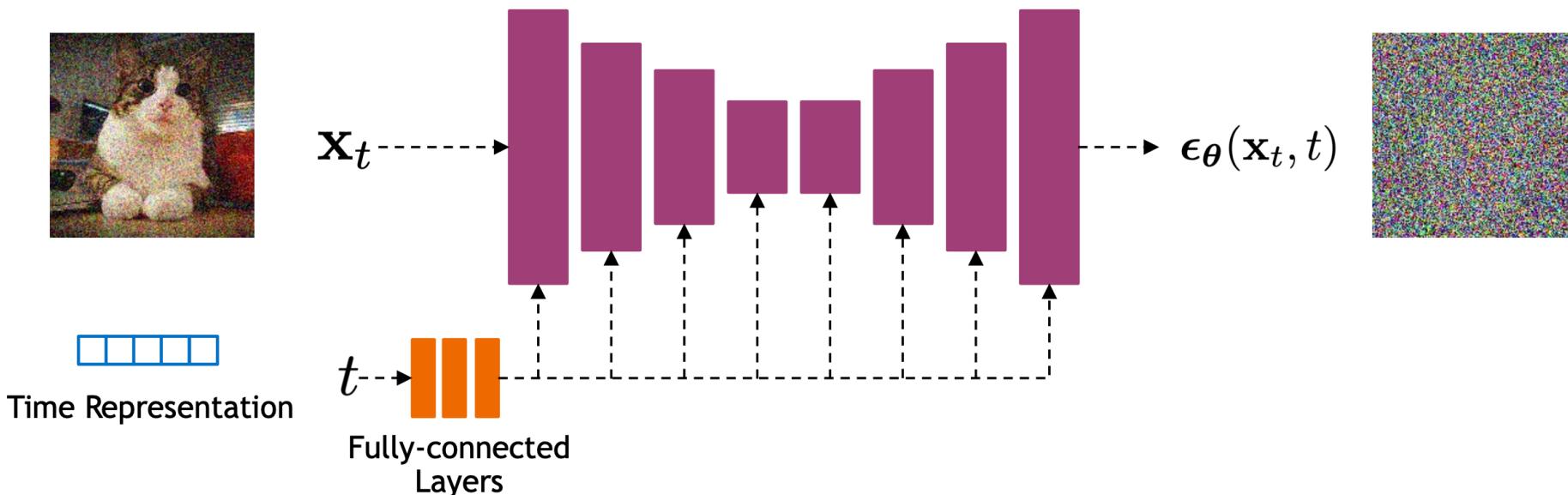
Reverse Denoising Process

Formal definition of forward and reverse processes in T steps:



Reverse Denoising Process

Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_\theta(\mathbf{x}_t, t)$



Time representation: sinusoidal positional embeddings or random Fourier features.

Time features are fed to the residual blocks using either simple spatial addition or using adaptive group normalization layers. (see [Dhariwal and Nichol NeurIPS 2021](#))

Training Loss (simplified)

Empirically, Ho et al. (2020) found that training the diffusion model works better with a simplified objective that ignores the weighting term:

$$\begin{aligned} L_t^{\text{simple}} &= \mathbb{E}_{t \sim [1, T], \mathbf{x}_0, \boldsymbol{\epsilon}_t} \left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\|^2 \right] \\ &= \mathbb{E}_{t \sim [1, T], \mathbf{x}_0, \boldsymbol{\epsilon}_t} \left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}_t, t)\|^2 \right] \end{aligned}$$

Denoising diffusion probabilistic models (DDPM)

Algorithm 1 Training

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

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:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
4:   
$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$$

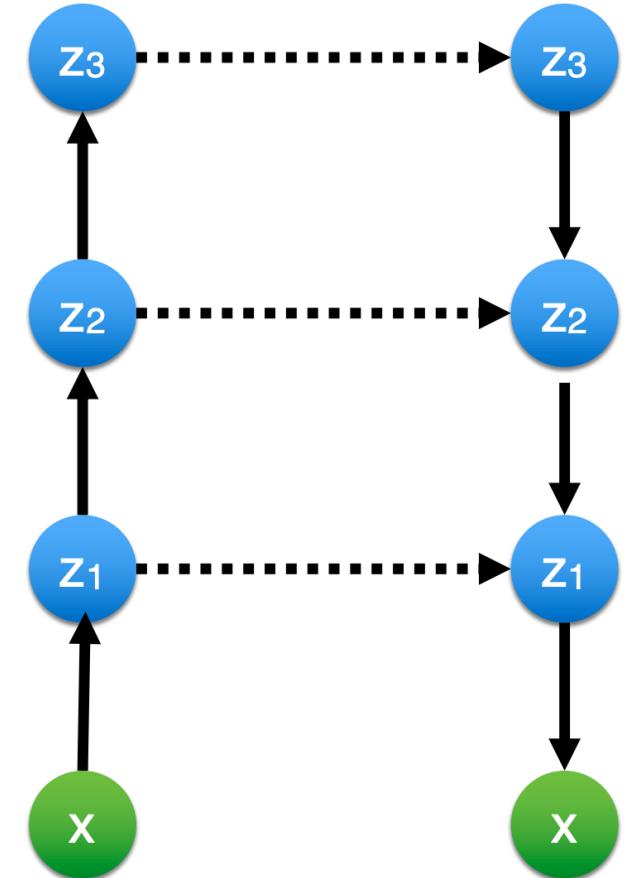
5: end for
6: return  $\mathbf{x}_0$ 
```

Connection to VAEs

Diffusion models can be considered as a special form of hierarchical VAEs.

However, in diffusion models:

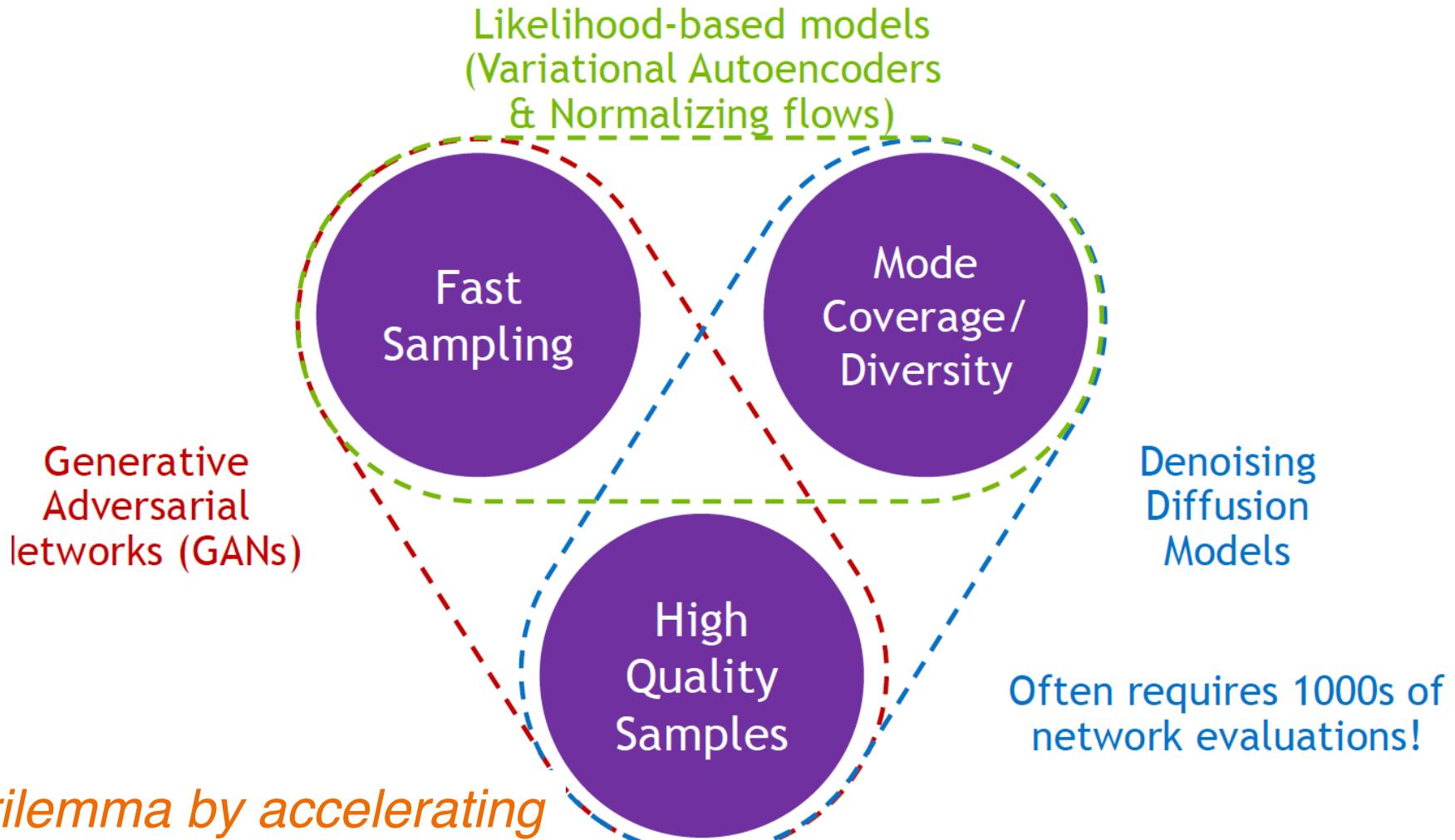
- The inference model is fixed: easier to optimize
- The latent variables have the same dimension as the data.
- The ELBO is decomposed to each time step: fast to train
 - Can be made extremely deep (even infinitely deep)
- The model is trained with some reweighting of the ELBO.



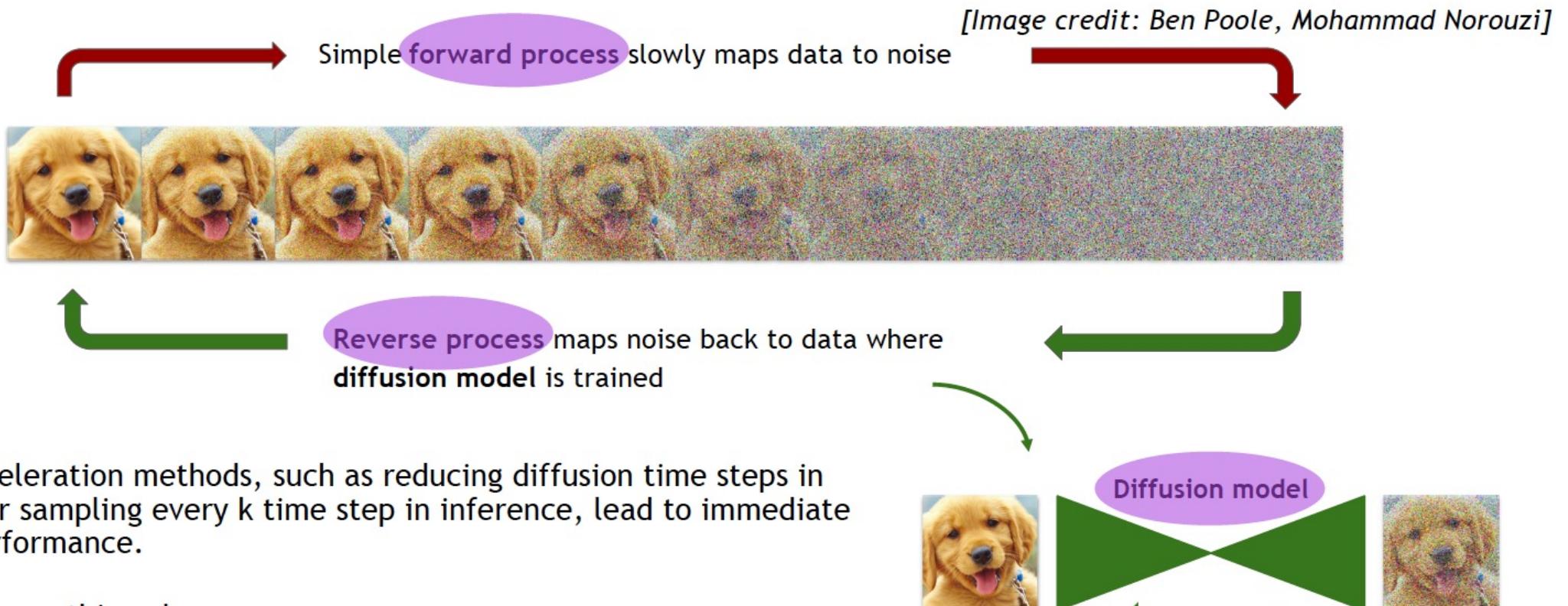
Inference model
 $q(z|x)$

Generative model
 $p(x,z)$

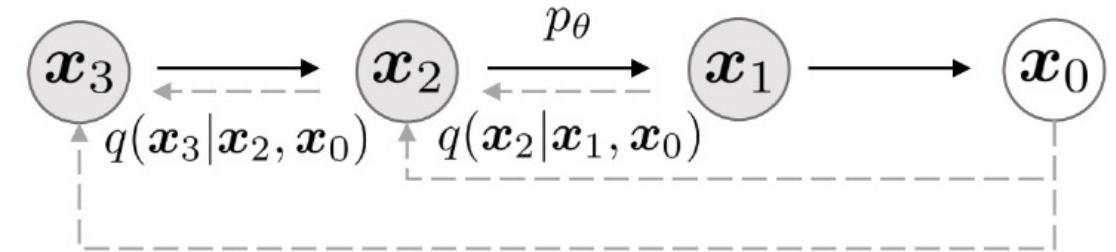
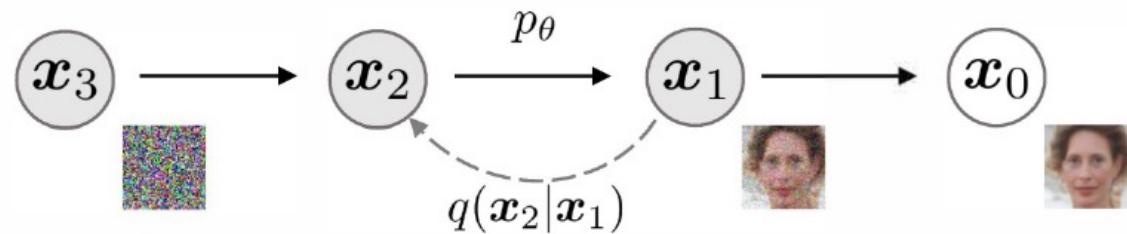
The generative learning trilemma



How to accelerate diffusion models?



From DDPM to DDIM: *Denoising diffusion implicit models*



Main Idea

Design a family of non-Markovian diffusion processes and corresponding reverse processes.

The process is designed such that the model can be optimized by the same surrogate objective as the original diffusion model.

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

Therefore, can take a pretrained diffusion model but with more choices of sampling procedure.

From DDPM to DDIM: *Denoising diffusion implicit models*

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \tilde{\sigma}_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0}{\sqrt{1 - \bar{\alpha}_t}}, \tilde{\sigma}_t^2 \mathbf{I}\right)$$

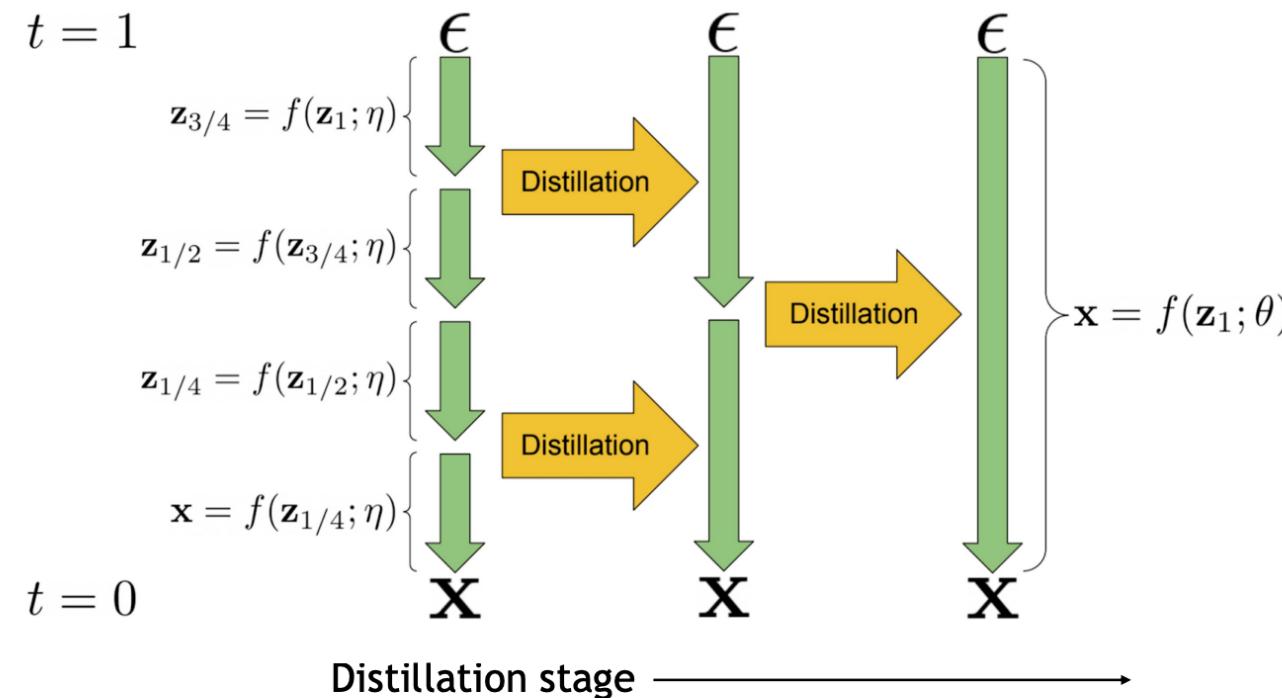
- ... often using its **deterministic form**: $\tilde{\sigma}_t^2 = 0, \forall t$
- With DDIM, it is possible to train the diffusion model up to any arbitrary number of forward steps but only **sample from a subset of steps in the generative process**

During generation, we only sample a subset of S diffusion steps $\{\tau_1, \dots, \tau_S\}$ and the inference process becomes:

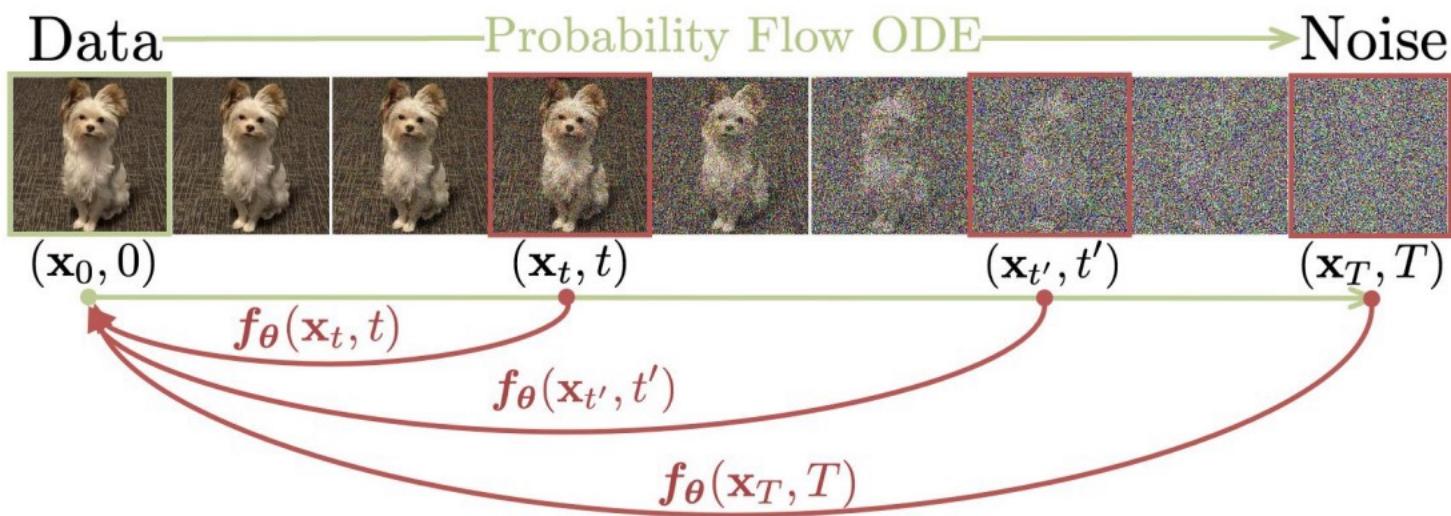
$$q_{\sigma,\tau}(\mathbf{x}_{\tau_{i-1}}|\mathbf{x}_{\tau_t}, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{\tau_{i-1}}; \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \frac{\mathbf{x}_{\tau_i} - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1 - \bar{\alpha}_t}}, \sigma_t^2 \mathbf{I})$$

Distill diffusion models into models using just 4-8 sampling steps!

- Distill a deterministic ODE sampler (i.e. DDIM sampler) to the same model architecture.
- At each stage, a “student” model is learned to distill two adjacent sampling steps of the “teacher” model to one sampling step.
- At next stage, the “student” model from previous stage will serve as the new “teacher” model.



Consistency Model (a special distillation)



- Given a Probability Flow ODE that smoothly converts data to noise, Consistency model learns to map any point on the ODE trajectory to its origin for generative modeling.
- Models of these mappings are called consistency models, as *their outputs are trained to be consistent for points on the same trajectory*.
- They support **fast one-step generation** by design, while still allowing multistep sampling to trade compute for sample quality

Conditional Generation

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

Variational upper bound: $L_\theta(\mathbf{x}_0|\mathbf{c}) = \mathbb{E}_q \left[L_T(\mathbf{x}_0) + \sum_{t>1} D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{c})) - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1, \mathbf{c}) \right].$

Incorporate conditions into U-Net

- Scalar conditioning: encode scalar as a vector embedding, simple spatial addition or adaptive group normalization layers.
- Image conditioning: channel-wise concatenation of the conditional image.
- Text conditioning: single vector embedding - spatial addition or adaptive group norm / a seq of vector embeddings - cross-attention.

Classifier guidance: Guiding Sampling usin the gradient of a trained classifier

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s .

Input: class label y , gradient scale s

$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$

for all t from T to 1 **do**

$\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$

$x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$

end for

return x_0

Main Idea

For class-conditional modeling of $p(\mathbf{x}_t|\mathbf{c})$, train an extra classifier $p(\mathbf{c}|\mathbf{x}_t)$

Mix its gradient with the diffusion/score model during sampling

Sample with a modified score: $\nabla_{\mathbf{x}_t} [\log p(\mathbf{x}_t|\mathbf{c}) + \omega \log p(\mathbf{c}|\mathbf{x}_t)]$

Classifier-free guidance: Implicit trick via Bayesian rule

- Instead of training an additional classifier, get an “implicit classifier” by jointly training a conditional and unconditional diffusion model:

$$p(\mathbf{c}|\mathbf{x}_t) \propto p(\mathbf{x}_t|\mathbf{c})/p(\mathbf{x}_t)$$

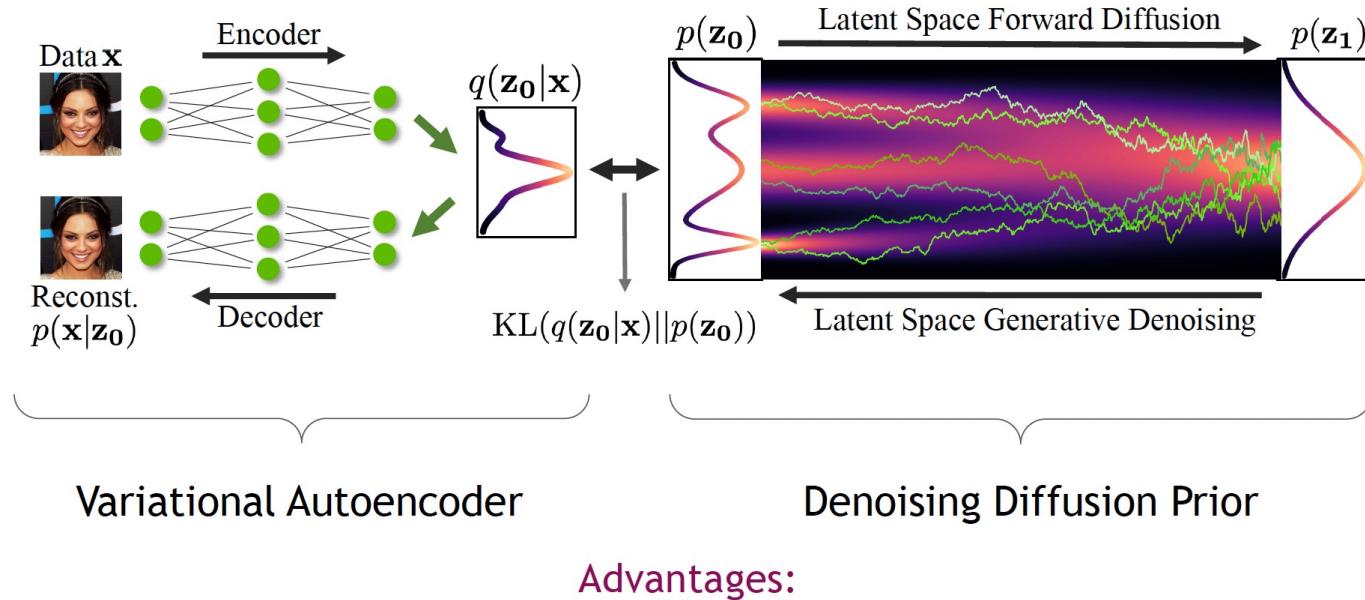
Conditional diffusion model Unconditional diffusion model

- In practice, $p(\mathbf{x}_t|\mathbf{c})$ and $p(\mathbf{x}_t)$ by randomly dropping the condition of the diffusion model at certain chance.
- The modified score with this implicit classifier included is:

$$\begin{aligned}\nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega \log p(\mathbf{c}|\mathbf{x}_t)] &= \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t|\mathbf{c}) + \omega(\log p(\mathbf{x}_t|\mathbf{c}) - \log p(\mathbf{x}_t))] \\ &= \nabla_{\mathbf{x}_t}[(1 + \omega) \log p(\mathbf{x}_t|\mathbf{c}) - \omega \log p(\mathbf{x}_t)]\end{aligned}$$

Latent Diffusion Models

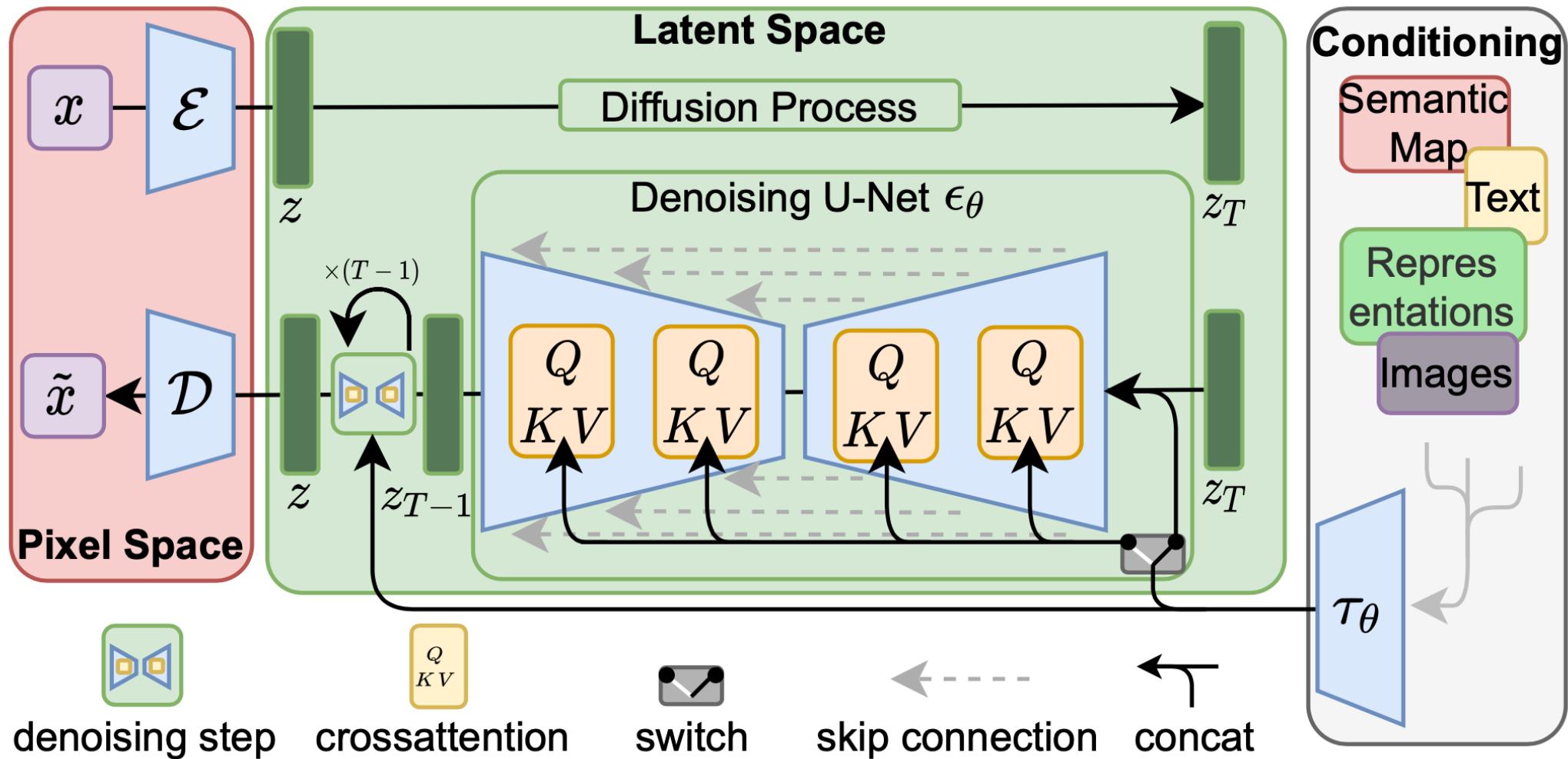
Variational autoencoder + score-based prior



- (1) The distribution of latent embeddings close to Normal distribution → *Simpler denoising, Faster synthesis!*
- (2) Latent space → *More expressivity and flexibility in design!*
- (3) Tailored Autoencoders → *More expressivity, Application to any data type (graphs, text, 3D data, etc.) !*

Latent Diffusion Model (CVPR'22): Important Jump toward High-Resolution!

*DDIM sampler
+ classifier-free
guidance +
many other
tweaks ...*



Latent Diffusion Model (CVPR'22): Important Jump toward High-Resolution!

- *The seminal work from Rombach et al. [CVPR 2022](#):*
 - *Two stage training: train autoencoder first, then train the diffusion prior*
 - *Focus on compression without of any loss in reconstruction quality*
 - *Demonstrated the expressivity of latent diffusion models on many conditional problems*
- *The efficiency and expressivity of latent diffusion models + open-source access fueled a large body of work in the community (e.g. **Stable Diffusion**!)*

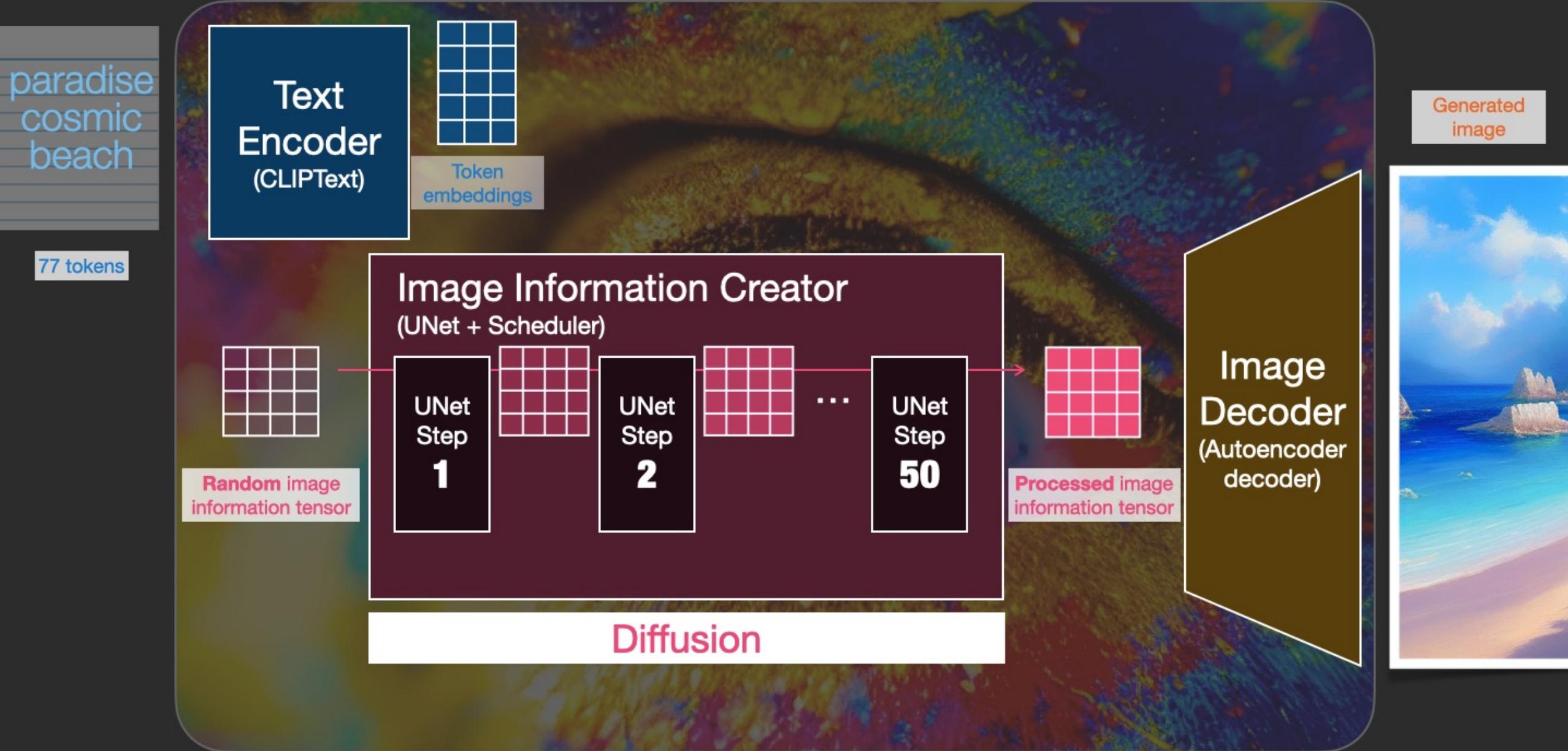
Latent Diffusion Model (CVPR'22): Important Jump toward High-Resolution!



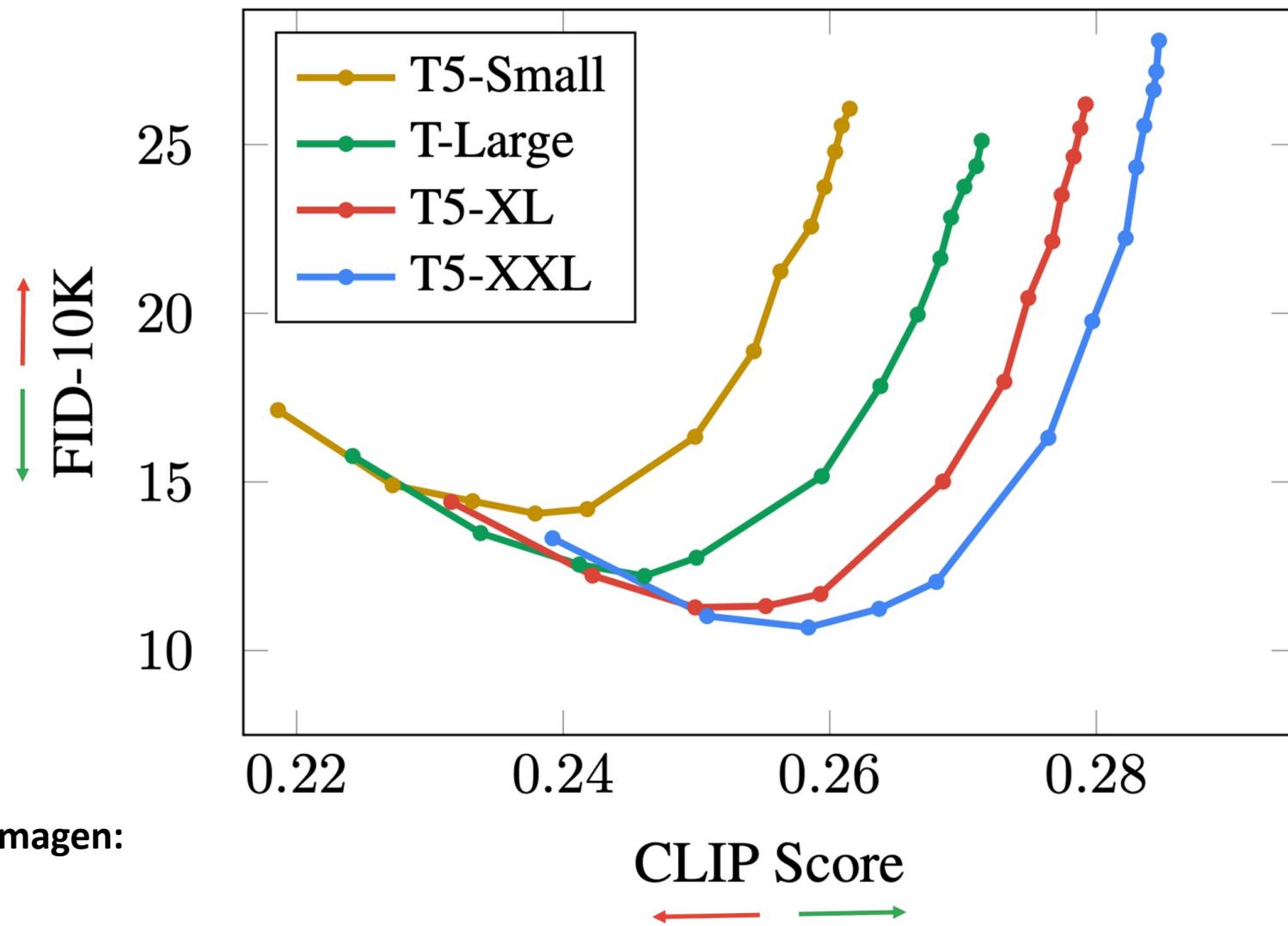
```
python scripts/txt2img.py --prompt "a sunset behind a mountain range, vector  
image" --ddim_eta 1.0 --n_samples 1 --n_iter 1 --H 384 --W 1024 --scale 5.0
```



Stable Diffusion



Larger Text Encoders → Better Alignment, Better Fidelity



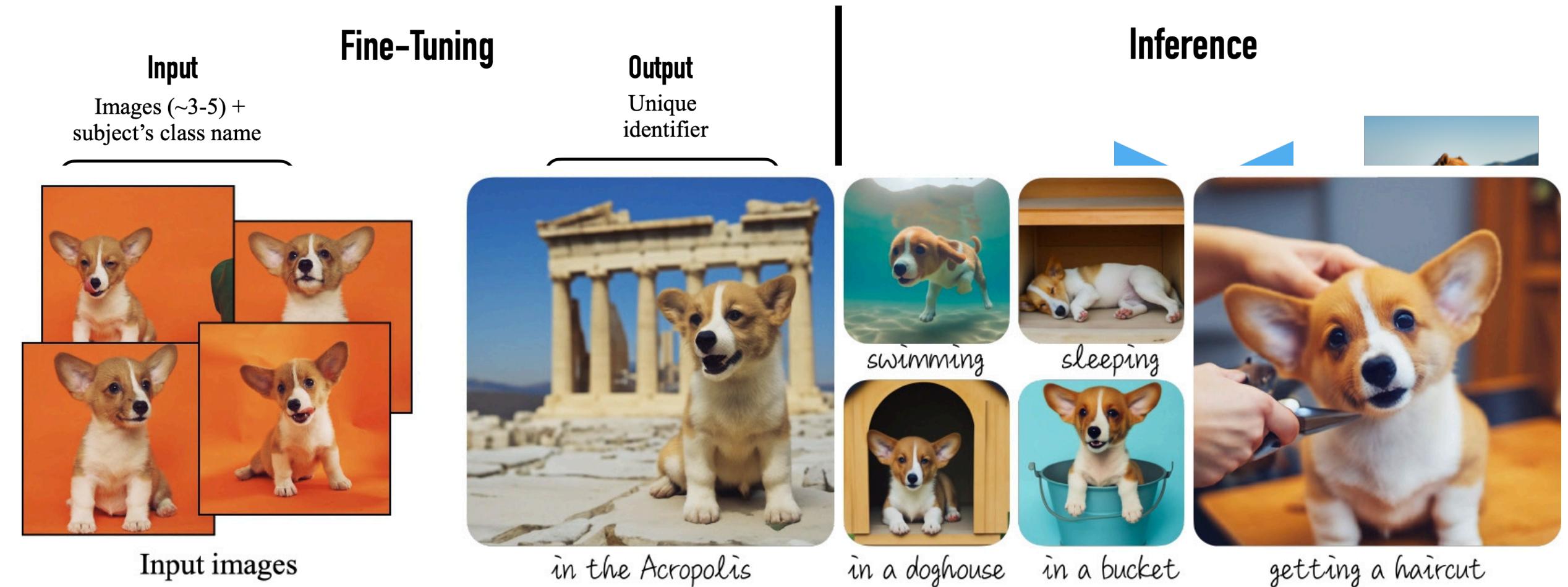
From Google Imagen:

A Few More Essentials...

- How to “Personalize”
- How to “Control”
- How to “Erase”



Personalizing Your Diffusion: DreamBooth



Personalizing Your Diffusion: DreamBooth



Input images



A [V] backpack in the
Grand Canyon



A wet [V] backpack
in water



A [V] backpack in Boston



A [V] backpack with the
night sky



Input images



A [V] teapot floating
in milk



A transparent [V] teapot
with milk inside

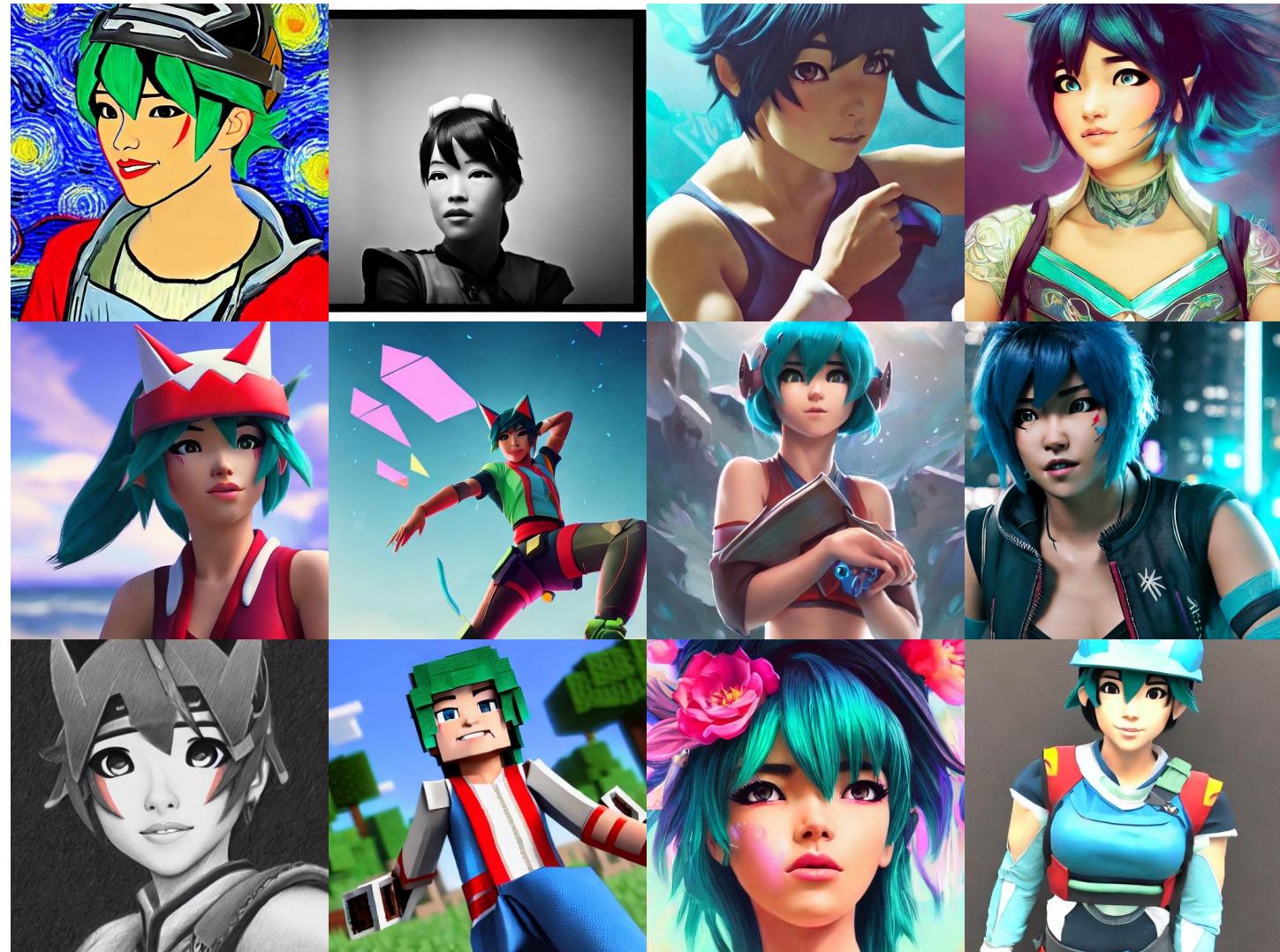


A [V] teapot
pouring tea

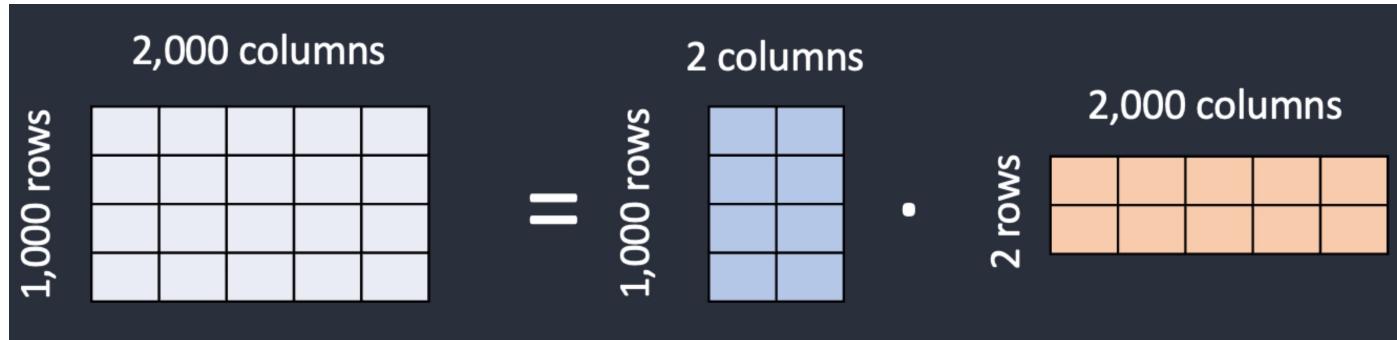


A [V] teapot floating
in the sea

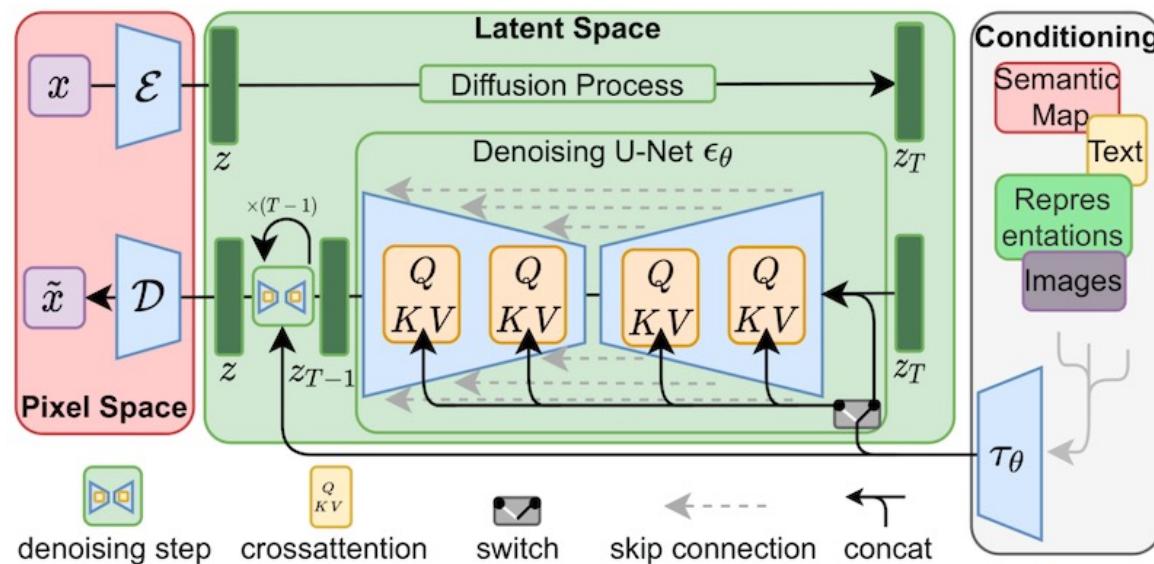
LoRA: Low-rank Adaptation for Fast Diffusion Fine-tuning



LoRA: Low-rank Adaptation for Fast Diffusion Fine-tuning



A LoRA model fine-tunes a model by adding its update weights to the pre-trained matrices, but using **low-rank compression**



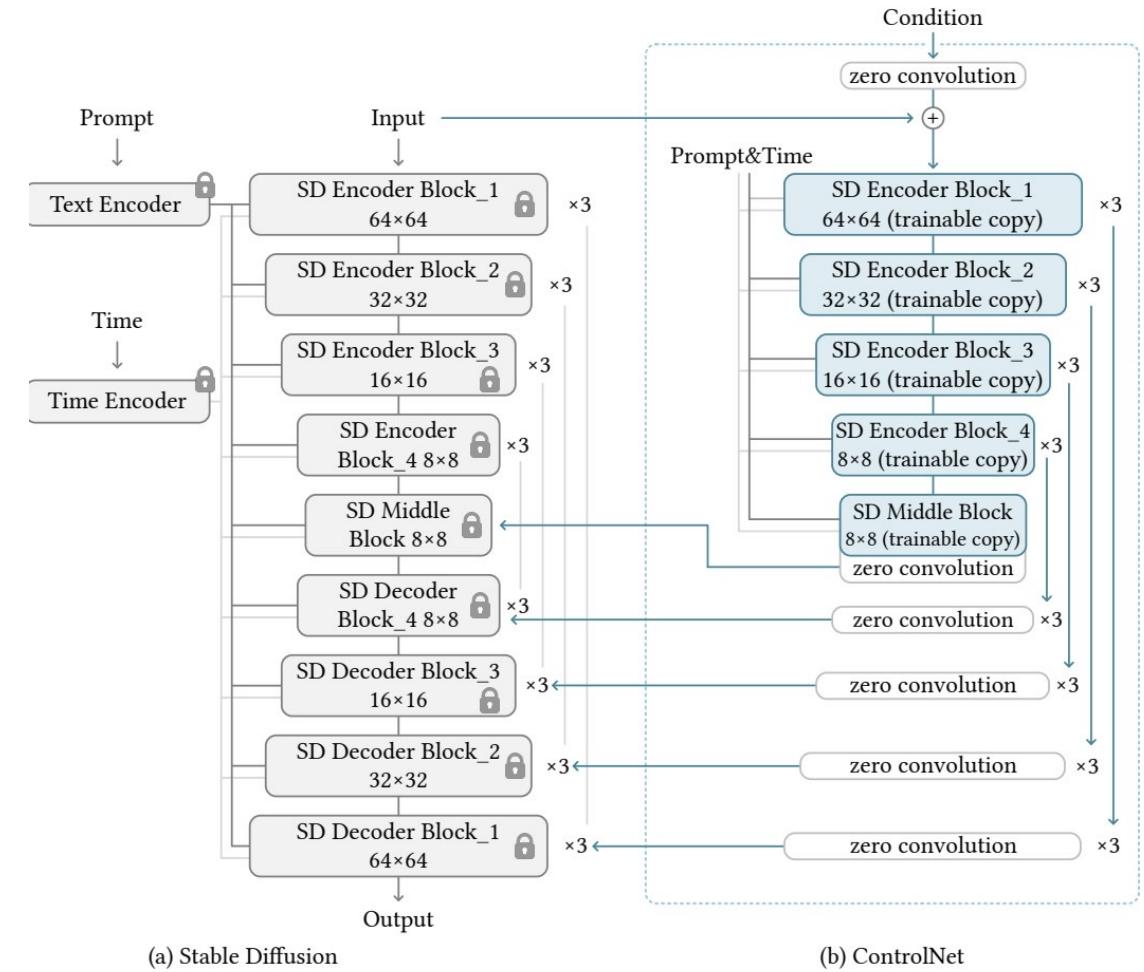
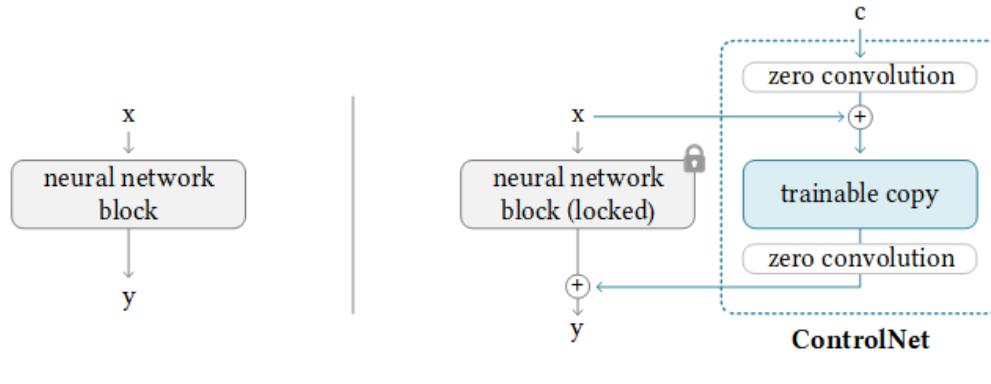
Original weights

$$W = W_0 + BA$$

Low-rank difference

In Practice now LORA fine-tunes the **cross-attention layers** (the QKV parts of the U-Net noise predictor)

ControlNet



ControlNet

Q: If the weight of a conv layer is zero, the gradient will also be zero, and the network will not learn anything. Why "zero convolution" works?

A: This is wrong. Let us consider a very simple

$$y = wx + b$$

and we have

$$\partial y / \partial w = x, \partial y / \partial x = w, \partial y / \partial b = 1$$

and if $w = 0$ and $x \neq 0$, then

$$\partial y / \partial w \neq 0, \partial y / \partial x = 0, \partial y / \partial b \neq 0$$

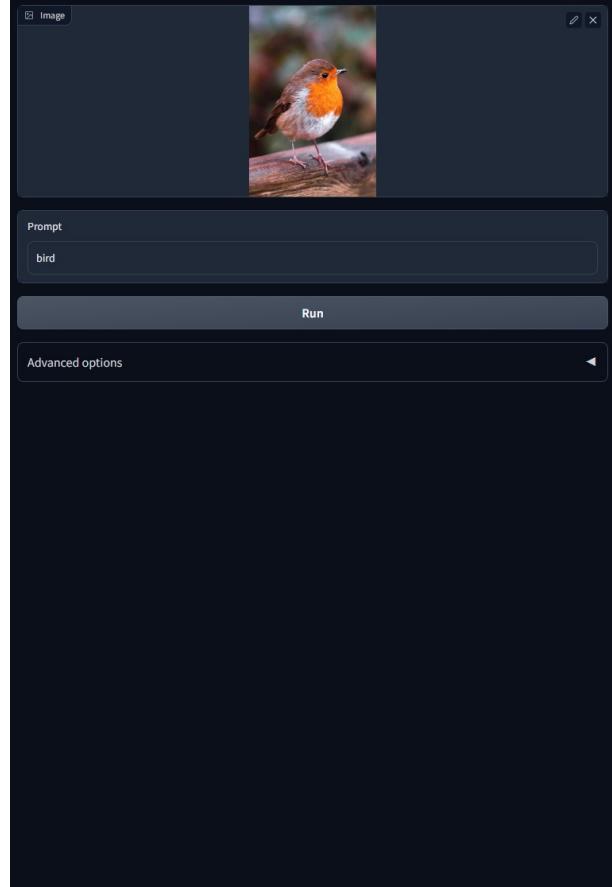
which means as long as $x \neq 0$, one gradient descent iteration will make w non-zero. Then

$$\partial y / \partial x \neq 0$$

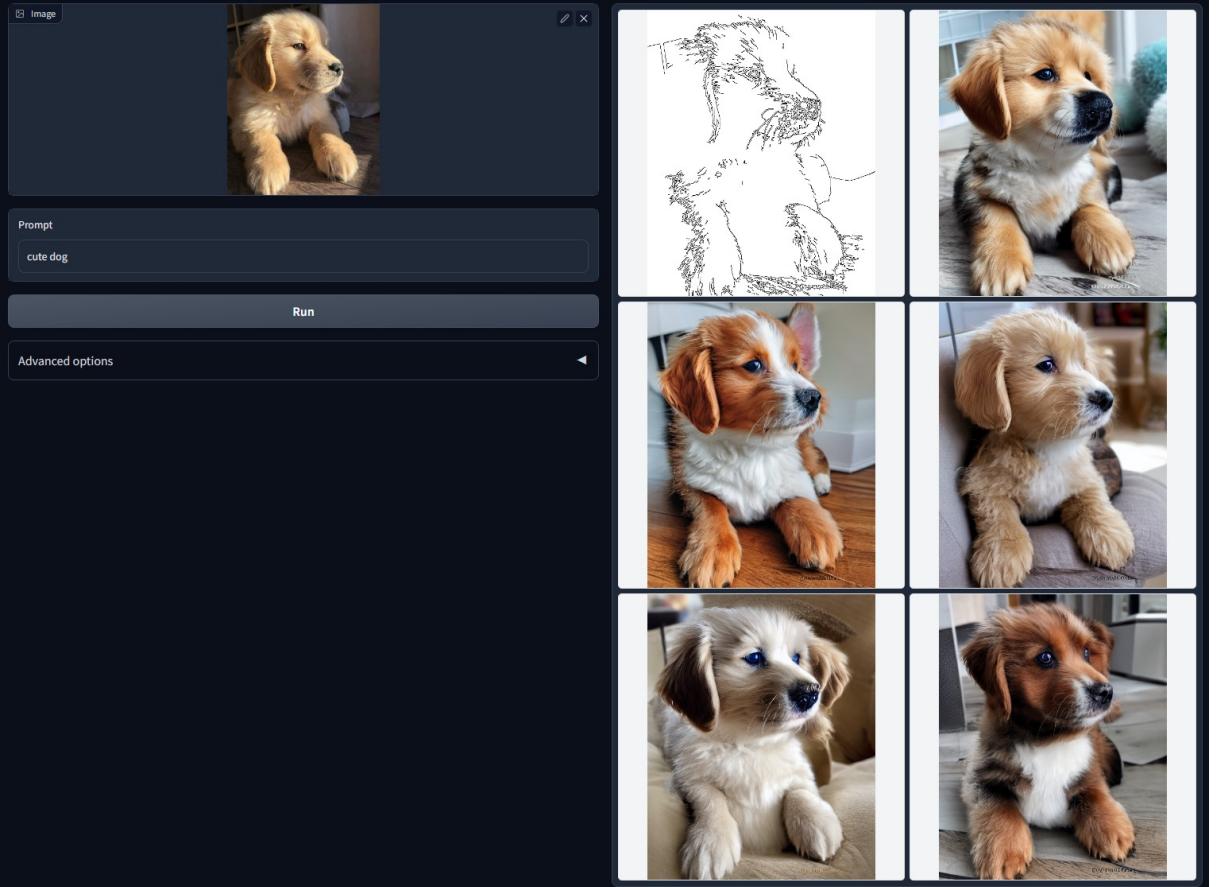
so that the zero convolutions will progressively become a common conv layer with non-zero weights.

ControlNet (Canny Edge)

Control Stable Diffusion with Canny Edge Maps



Control Stable Diffusion with Canny Edge Maps

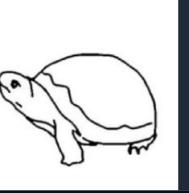


ControlNet (Sketch Lines)



ControlNet (User Scribbles)

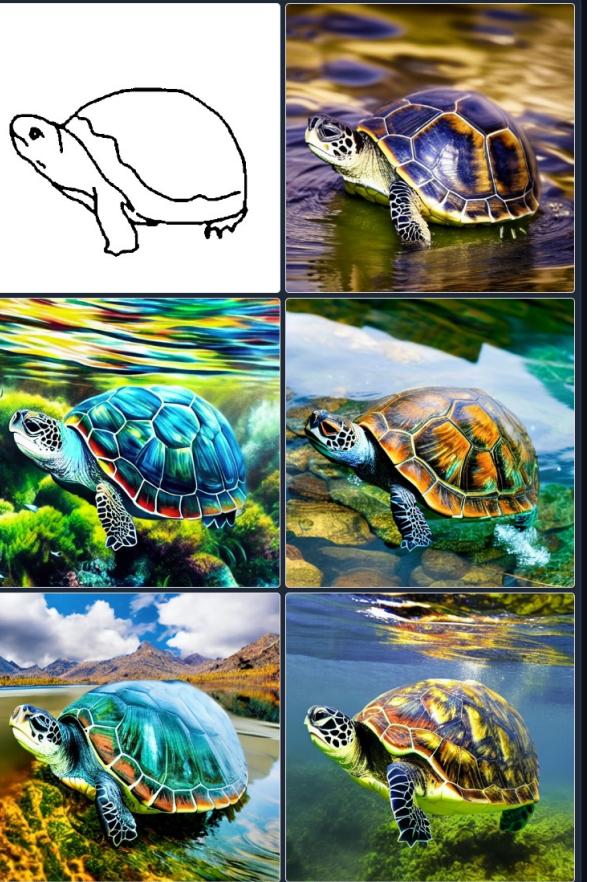
Control Stable Diffusion with Scribble Maps

Image 

Prompt
turtle

Run

Advanced options



Control Stable Diffusion with Scribble Maps

Image 

Prompt
hot air balloon

Run

Advanced options



ControlNet (Human Pose)

Control Stable Diffusion with Human Pose

Image

Prompt

Chef in the kitchen

Run

Advanced options

Control Stable Diffusion with Human Pose

Image

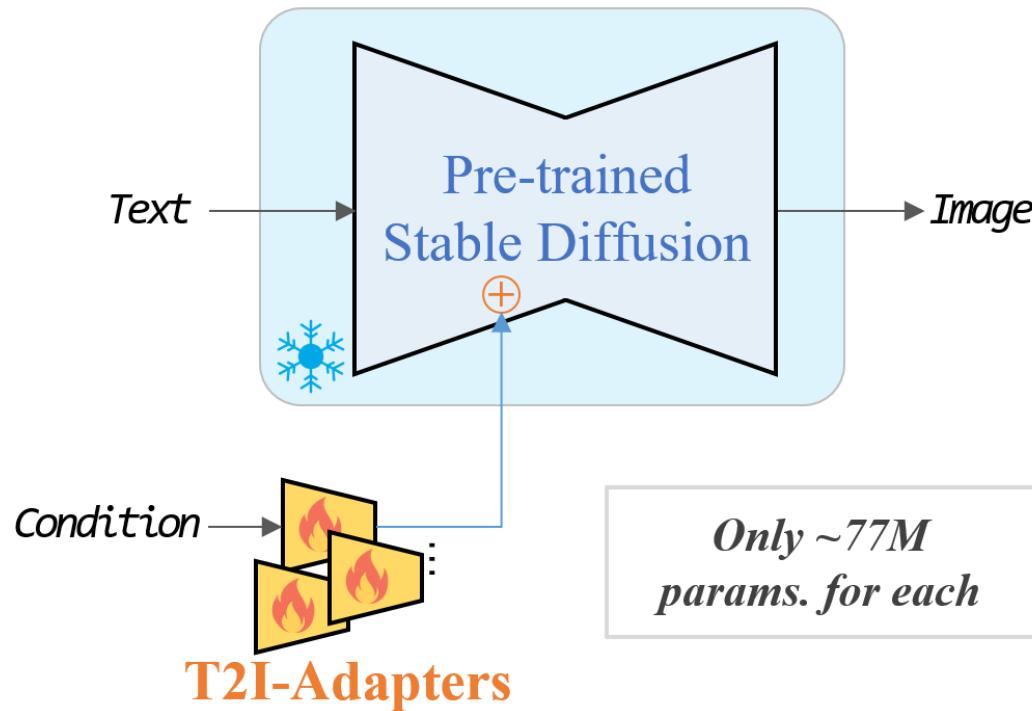
Prompt

An astronaut on the Moon

Run

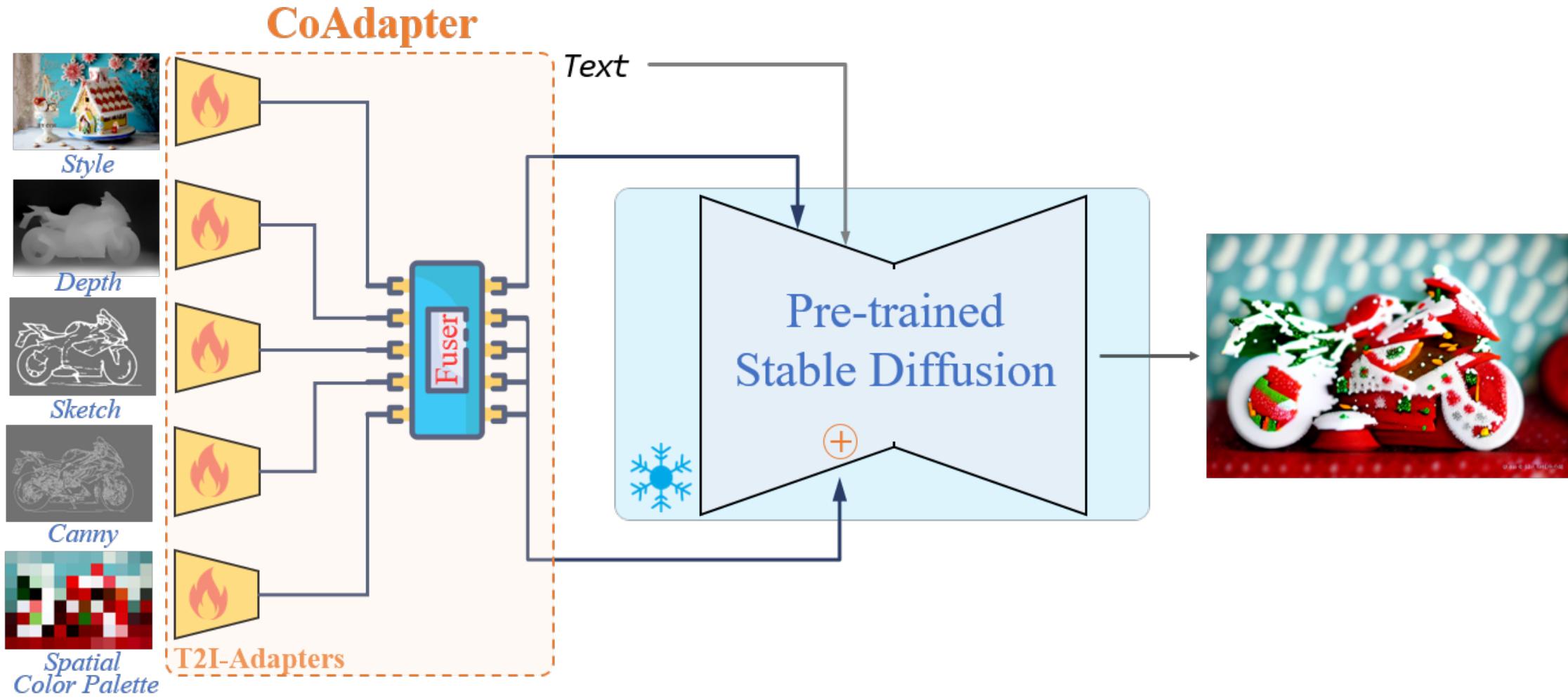
Advanced options

T2I Adapter



-
- ✓ **Plug-and-play.** Not affect original network topology and generation ability
 - ✓ **Simple and small.** ~77M parameters and ~300M storage
 - ✓ **Flexible.** Various adapters for different control conditions
 - ✓ **Composable.** More than one adapter can be easily composed to achieve multi-condition control
 - ✓ **Generalizable.** Can be directly used on customized models
-

T2I Adapter

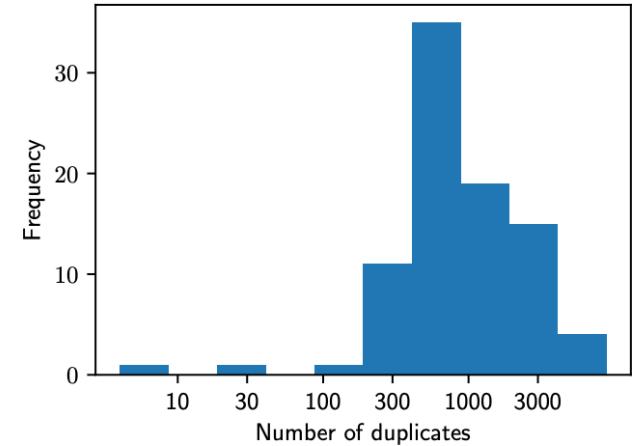


Data Memorization in Diffusion Models

- Due to the likelihood-base objective function, diffusion models can "memorize" data.
- And with a higher chance than GANs!
- Nevertheless, a lot of "memorized images" are highly-duplicated in the dataset.

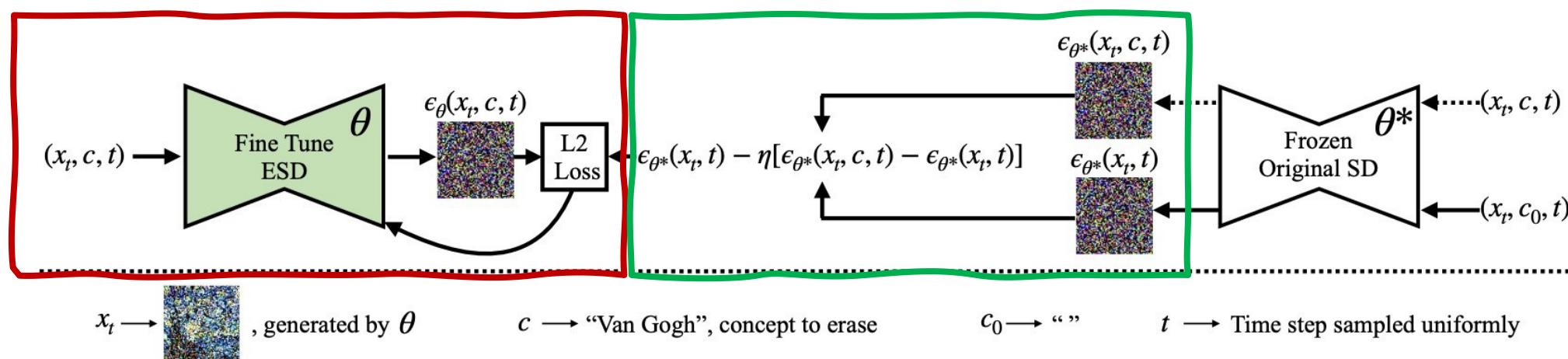
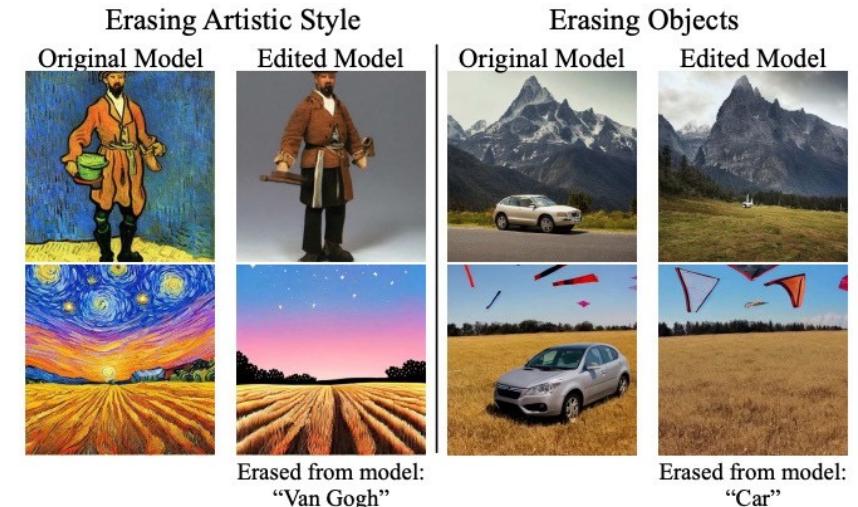


	Architecture	Images Extracted	FID
GANs	StyleGAN-ADA [43]	150	2.9
	DiffBigGAN [82]	57	4.6
	E2GAN [69]	95	11.3
	NDA [63]	70	12.6
	WGAN-ALP [68]	49	13.0
DDPMs	OpenAI-DDPM [52]	301	2.9
	DDPM [33]	232	3.2



Erasing Concepts in Diffusion Models

- Fine-tune a model to remove unwanted concepts.
- From original model, obtain score via negative CFG.
- A new model is fine-tuned from the new score function.



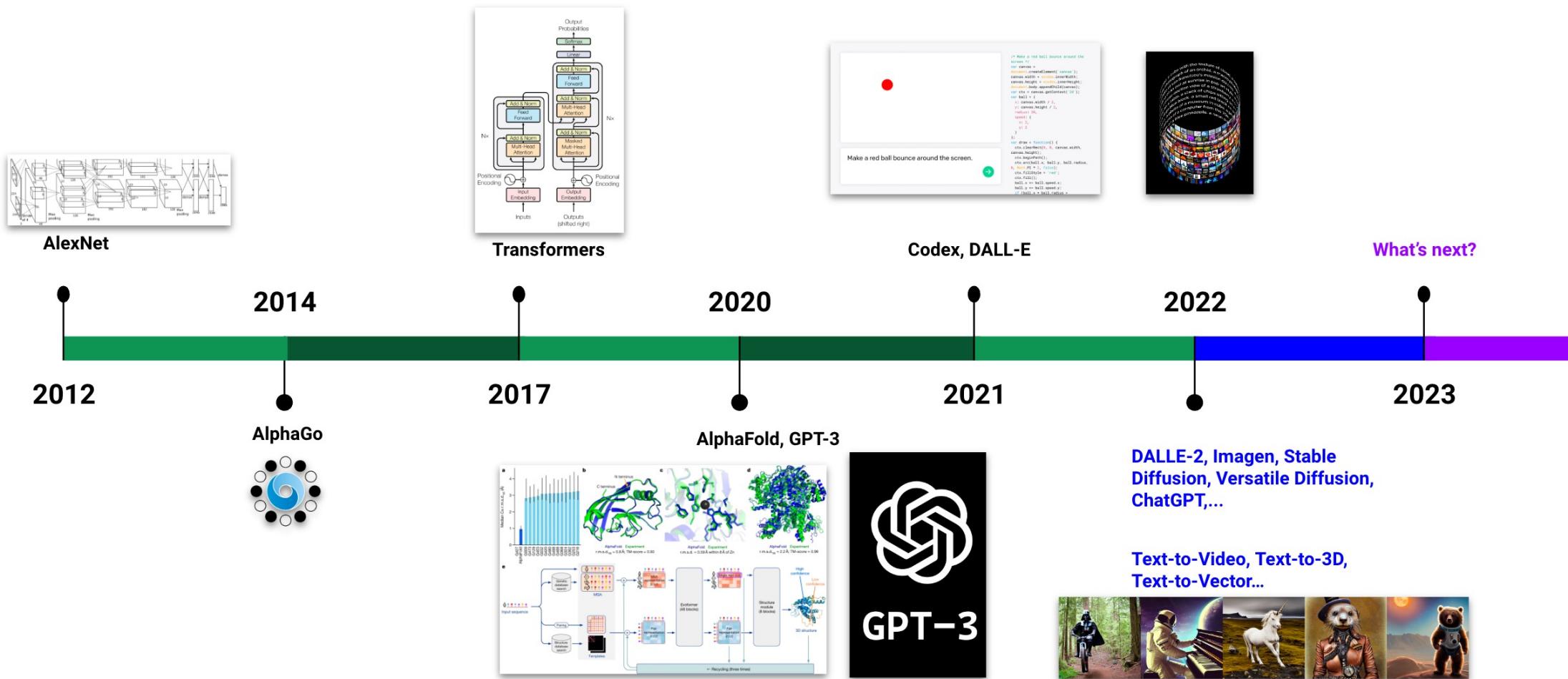
Is Diffusion Model Destined to be the Final Winner?



StyleGAN-T: Unlocking the Power of GANs for Fast Large-Scale Text-to-Image Synthesis

[Axel Sauer](#) [Tero Karras](#) [Samuli Laine](#) [Andreas Geiger](#) [Timo Aila](#)

Generative AI is revolutionizing the AI landscape EVERYDAY

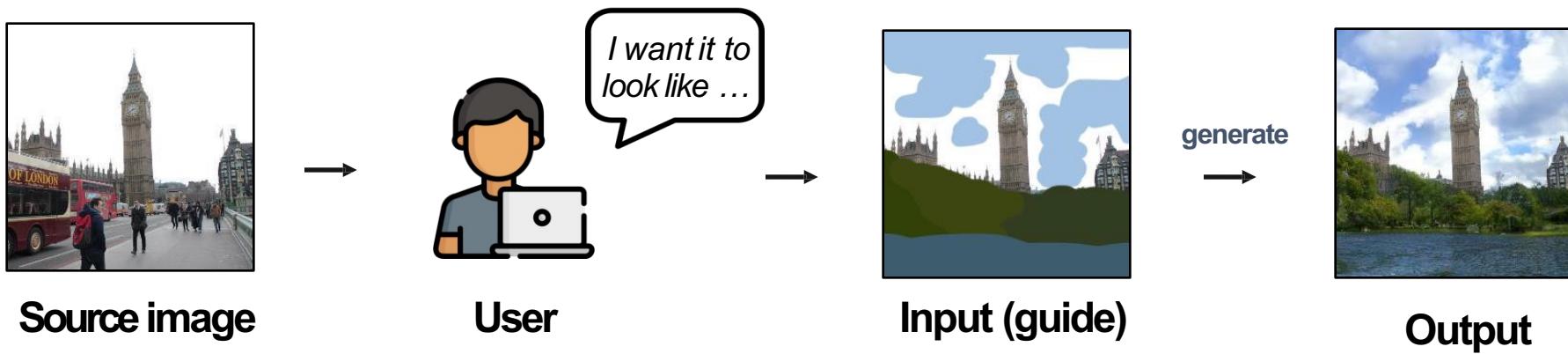


Application Aspects, Now!

- Image Editing
- Video Generation
- 3D Generation
- Ethic and Privacy Concerns



How to perform guided synthesis/editing?



Realistic

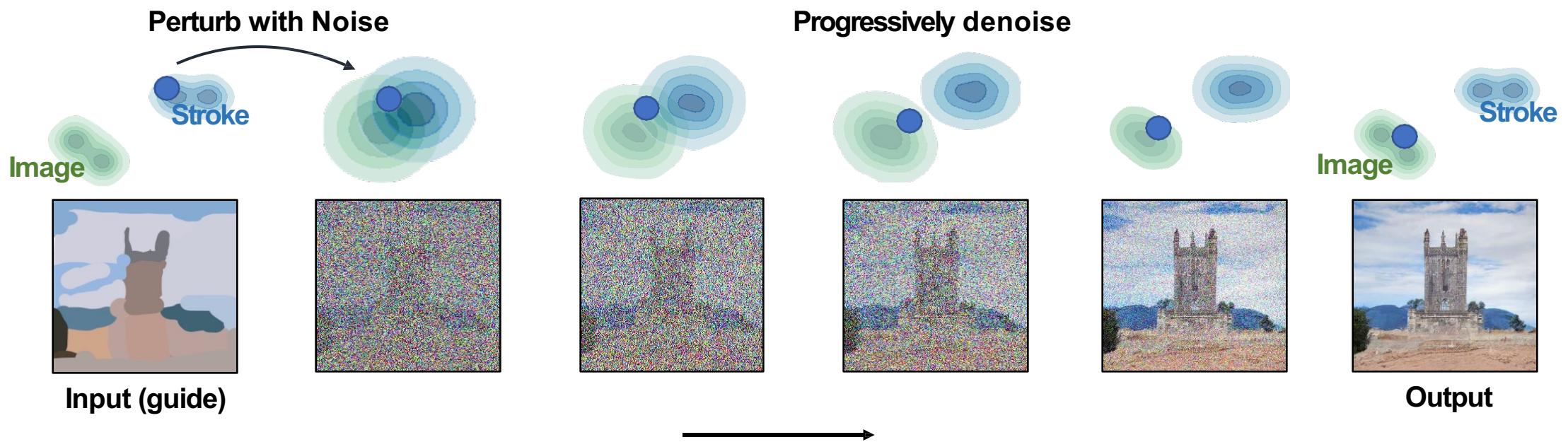


Faithful



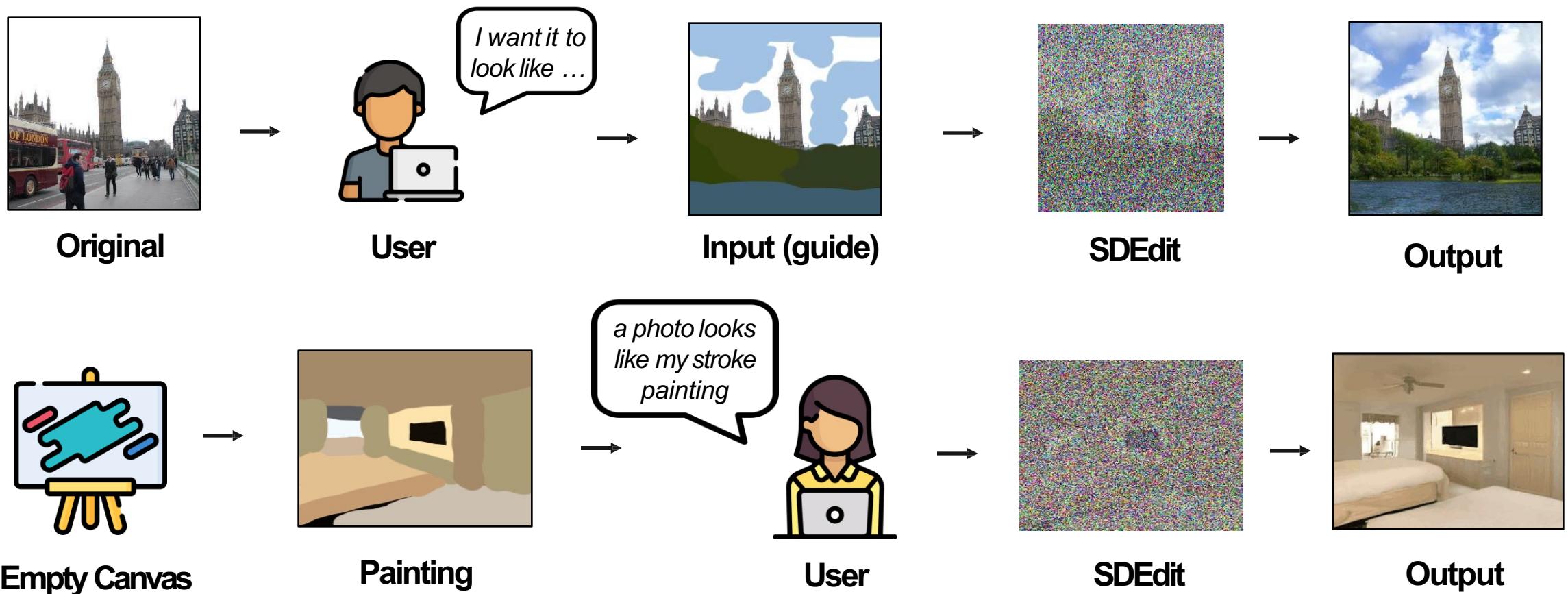
SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations

First perturb the input with **Gaussian noise** and then progressively remove the noise using a pretrained diffusion model.

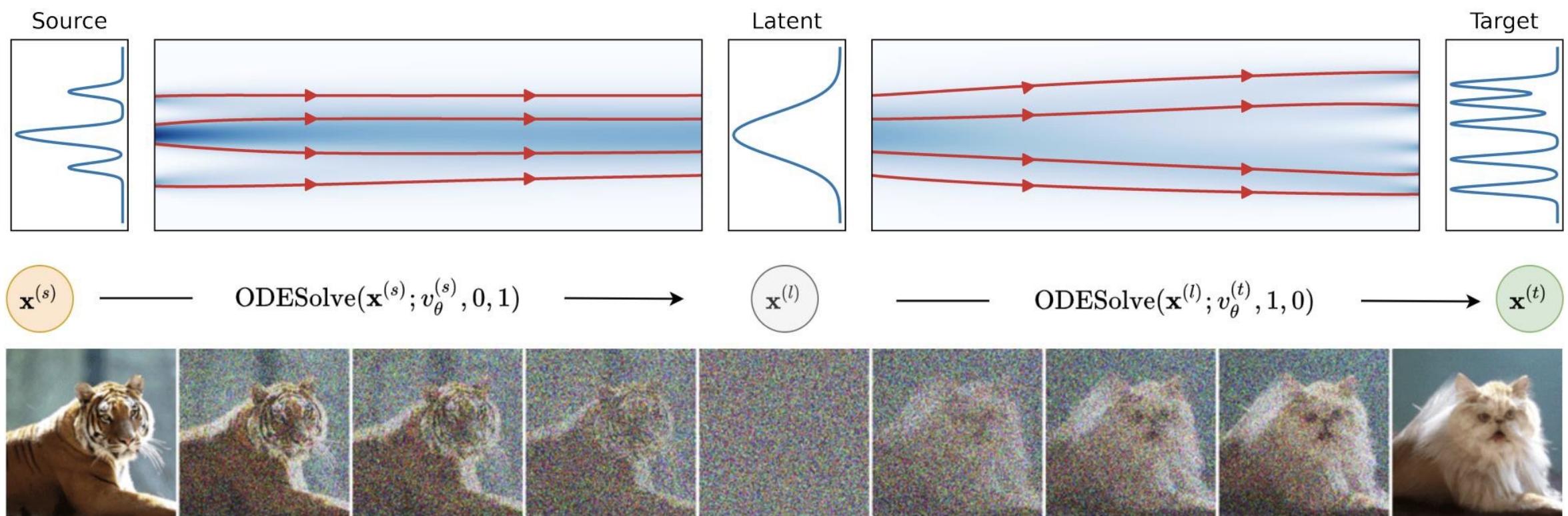


Gradually projects the input to the manifold of natural images.

Fine-grained control using strokes



Style transfer with DDIM inversion



Style transfer with DDIM inversion



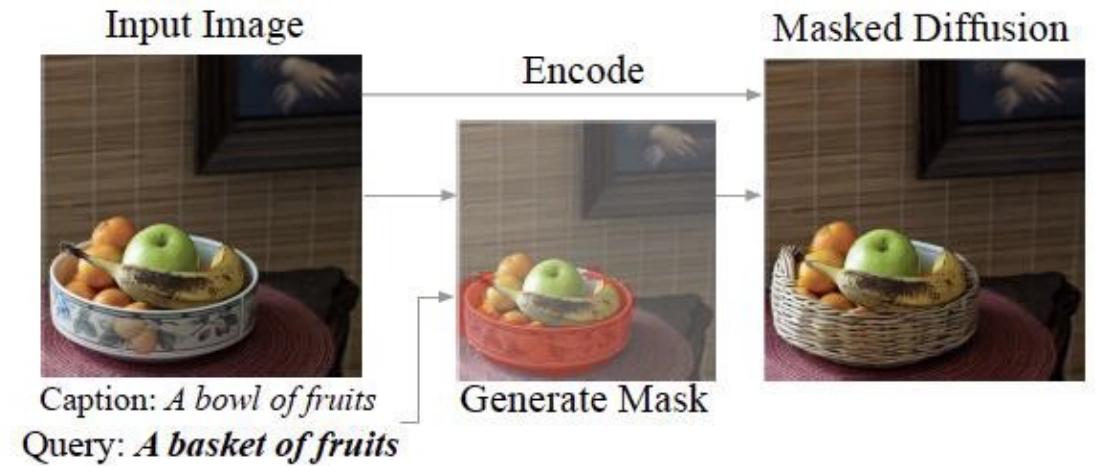
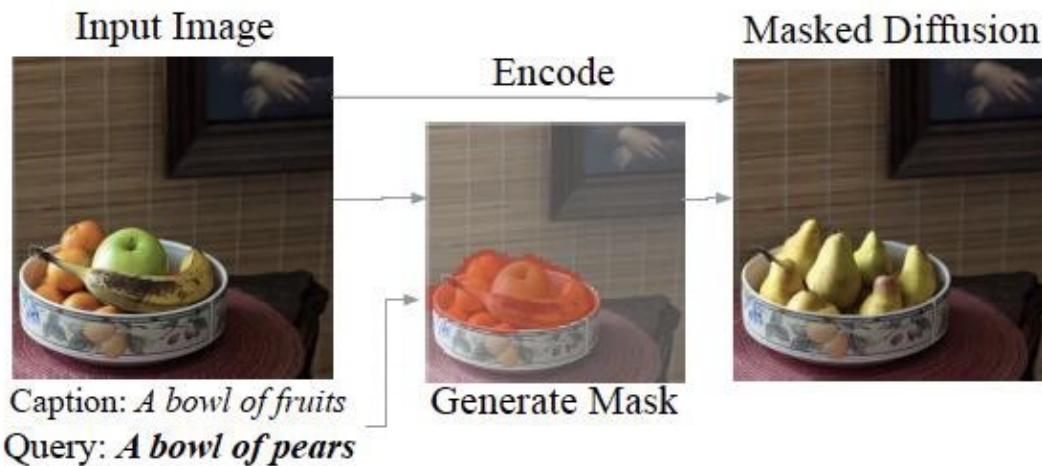
Multi-domain translation



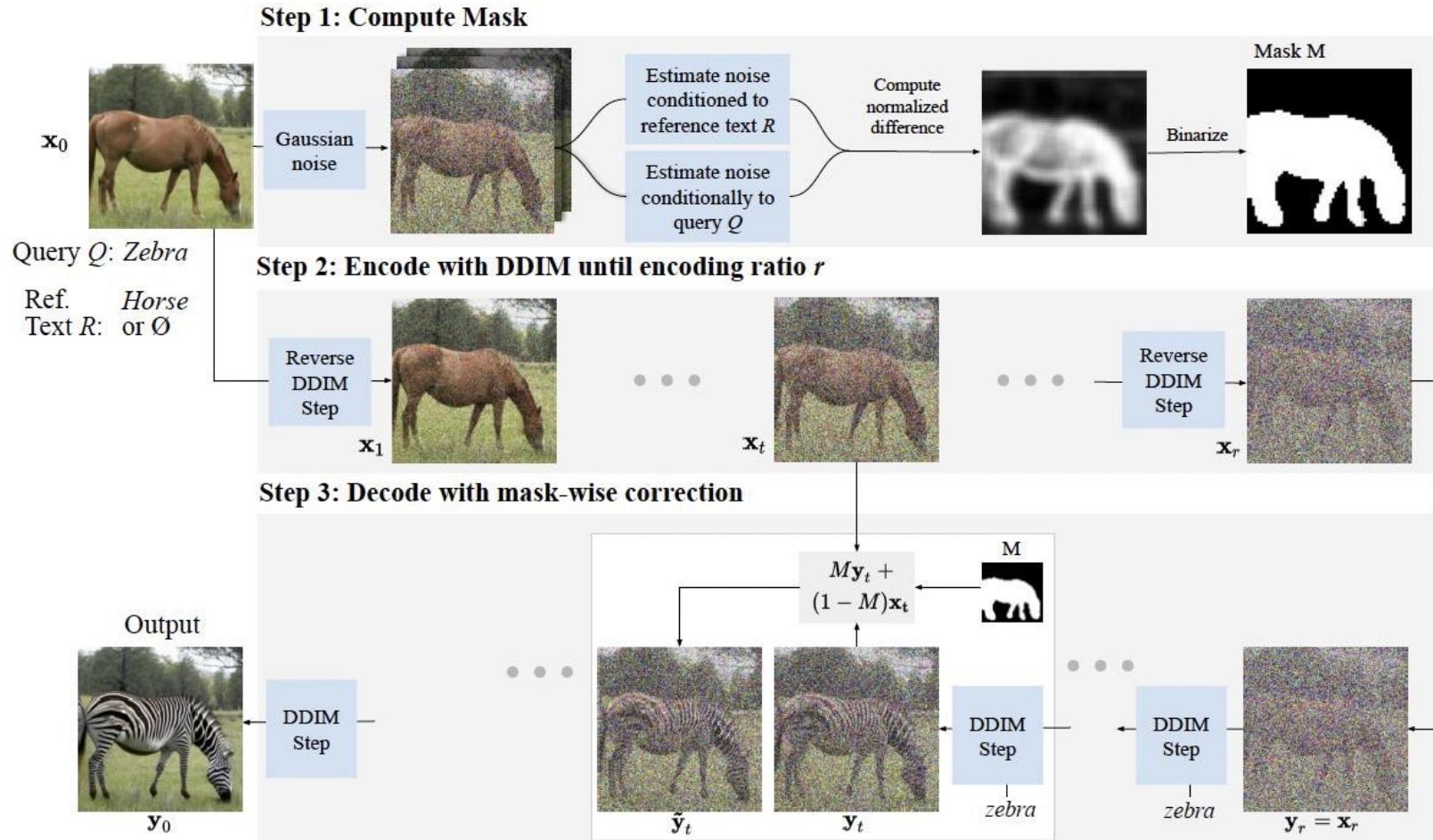
Example-Guided Color Transfer

DiffEdit: Diffusion-based semantic image editing with mask guidance

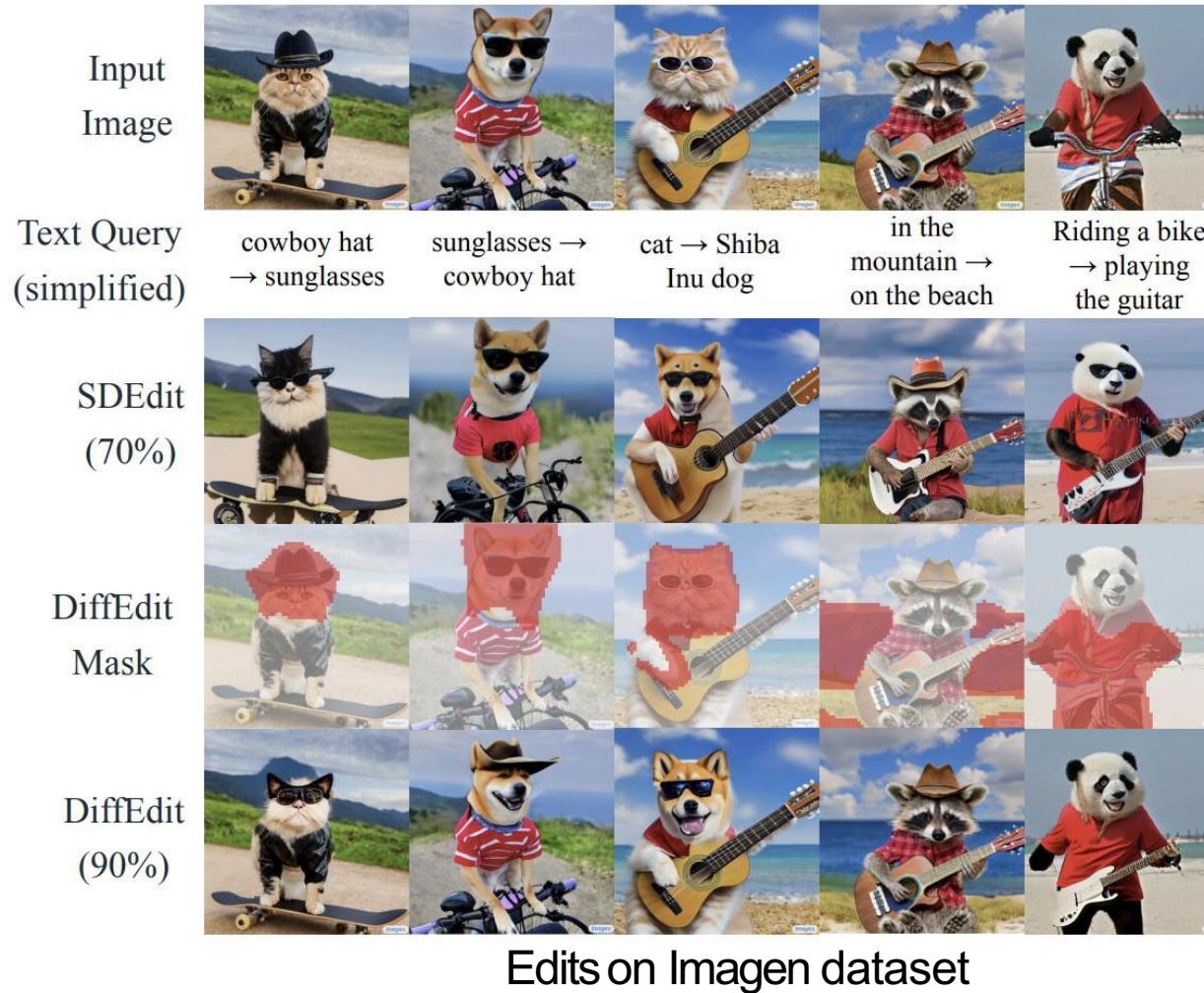
Instead of asking users to provide the mask, the model will generate the mask itself based on the caption and query.



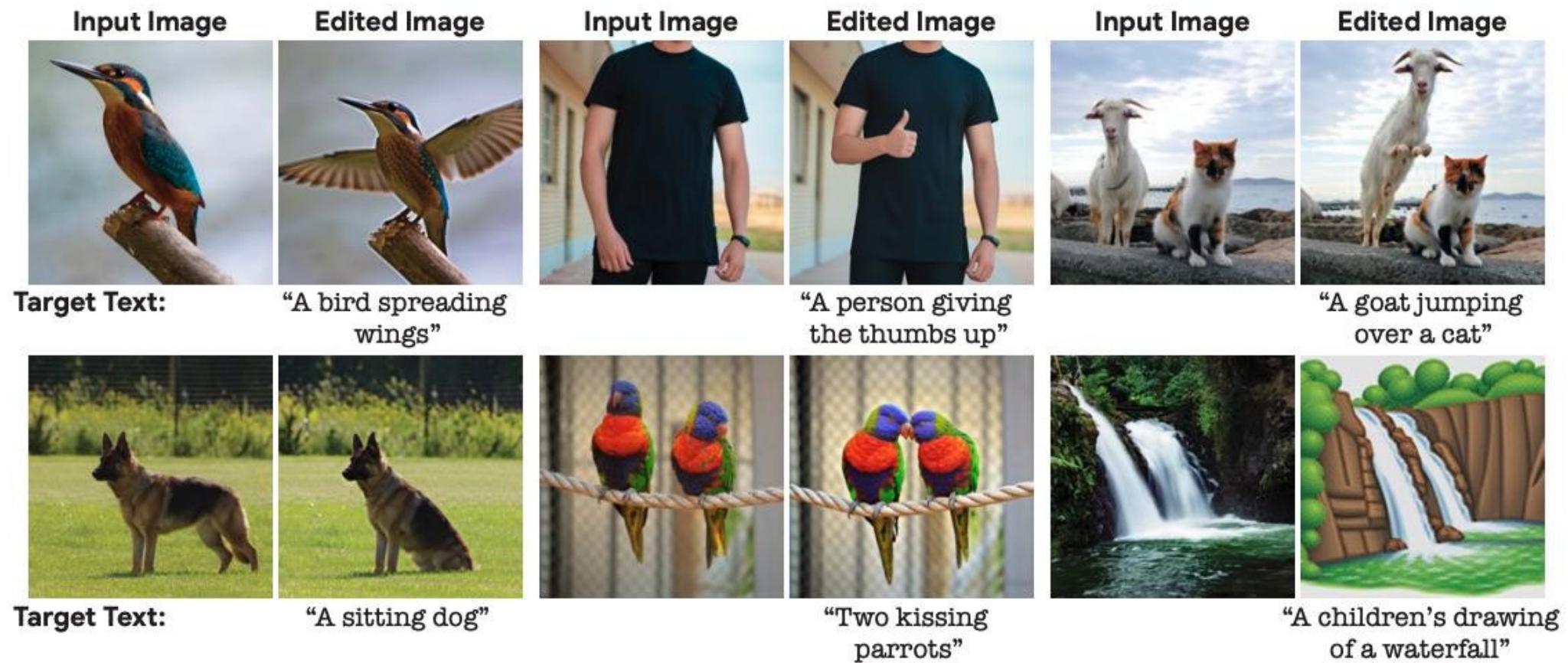
DiffEdit: Diffusion-based semantic image editing with mask guidance



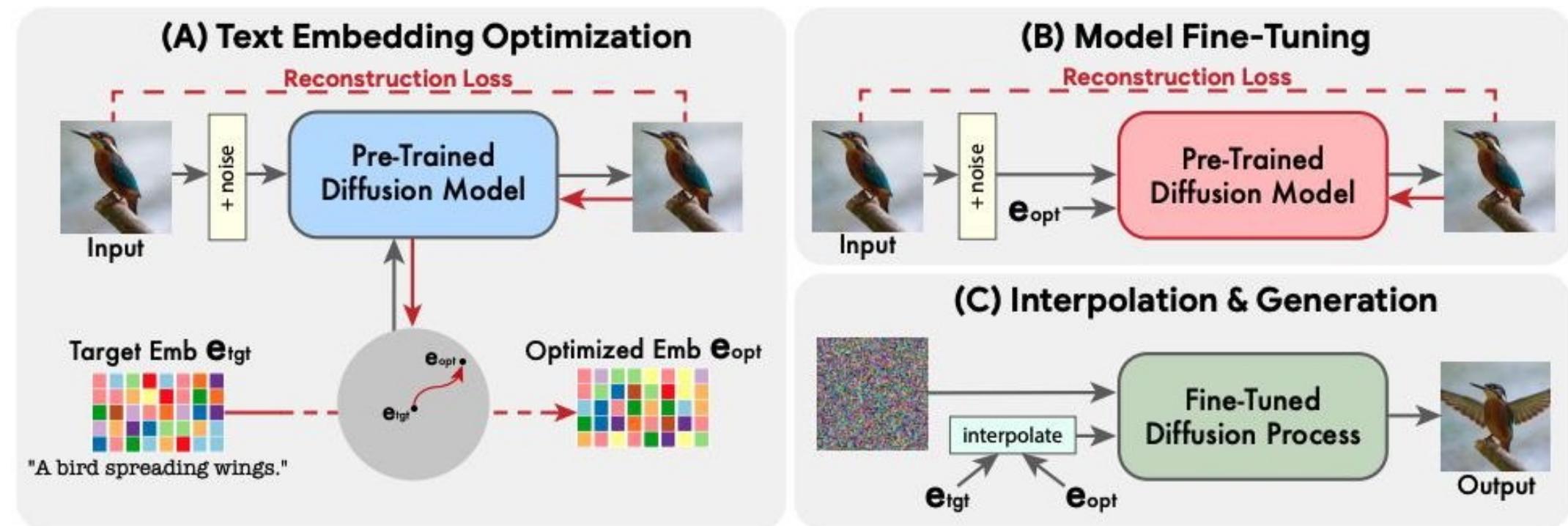
DiffEdit: Diffusion-based semantic image editing with mask guidance



Imagic: Text-Based Real Image Editing with Diffusion Models



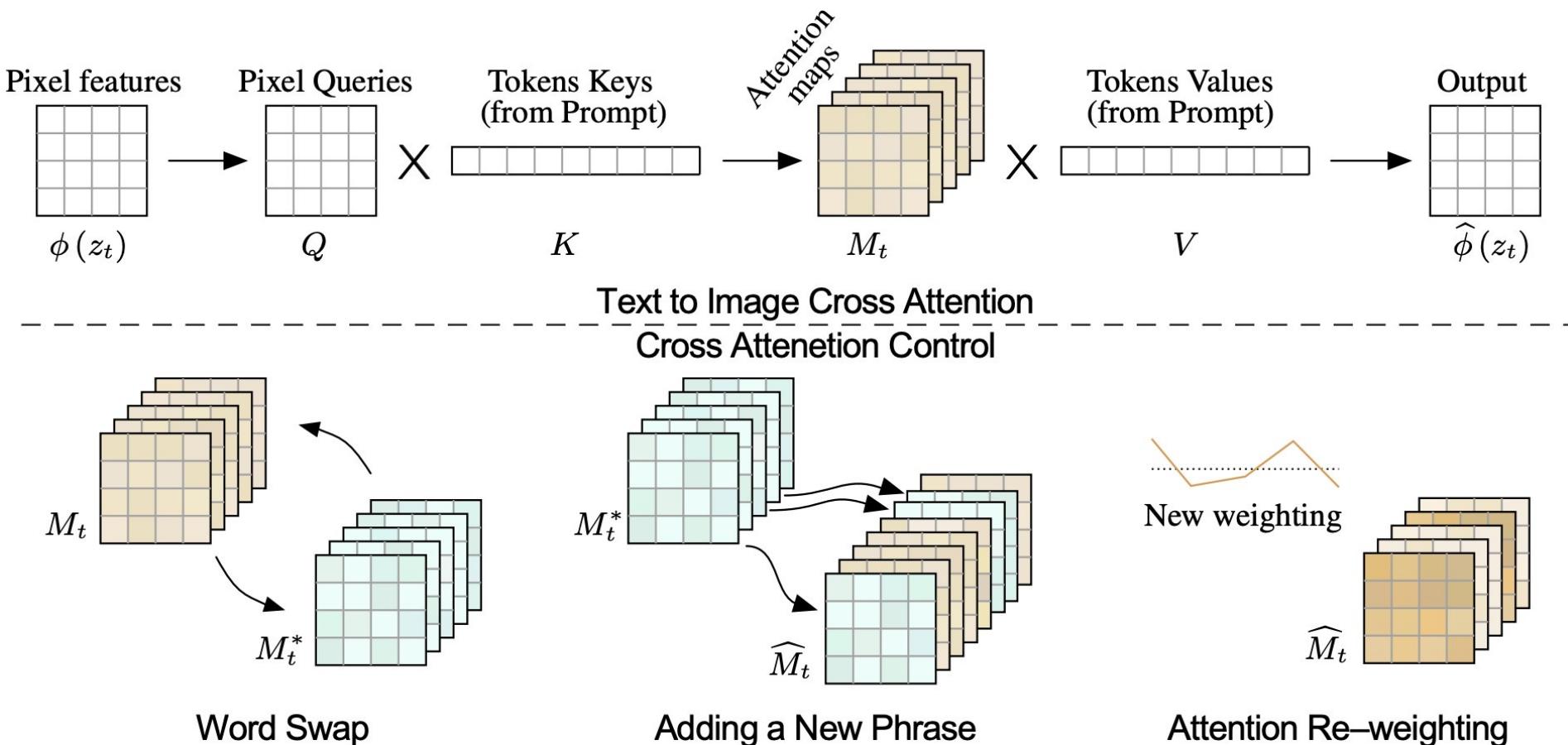
Imagic: Text-Based Real Image Editing with Diffusion Models



Prompt-to-Prompt Image Editing with Cross-Attention Control



Prompt-to-Prompt Image Editing with Cross-Attention Control



InstructPix2Pix: Learning to Follow Image Editing Instructions

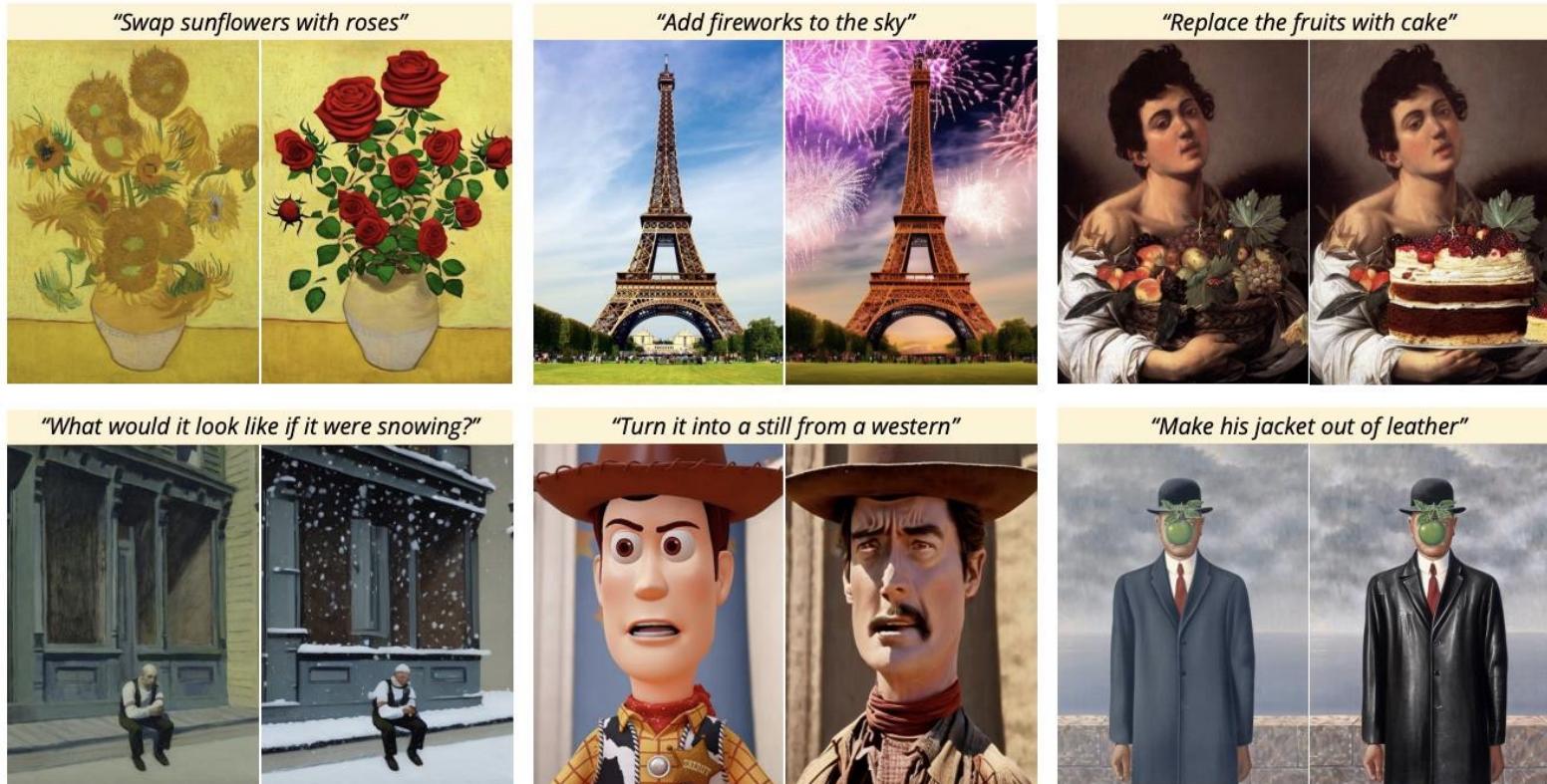
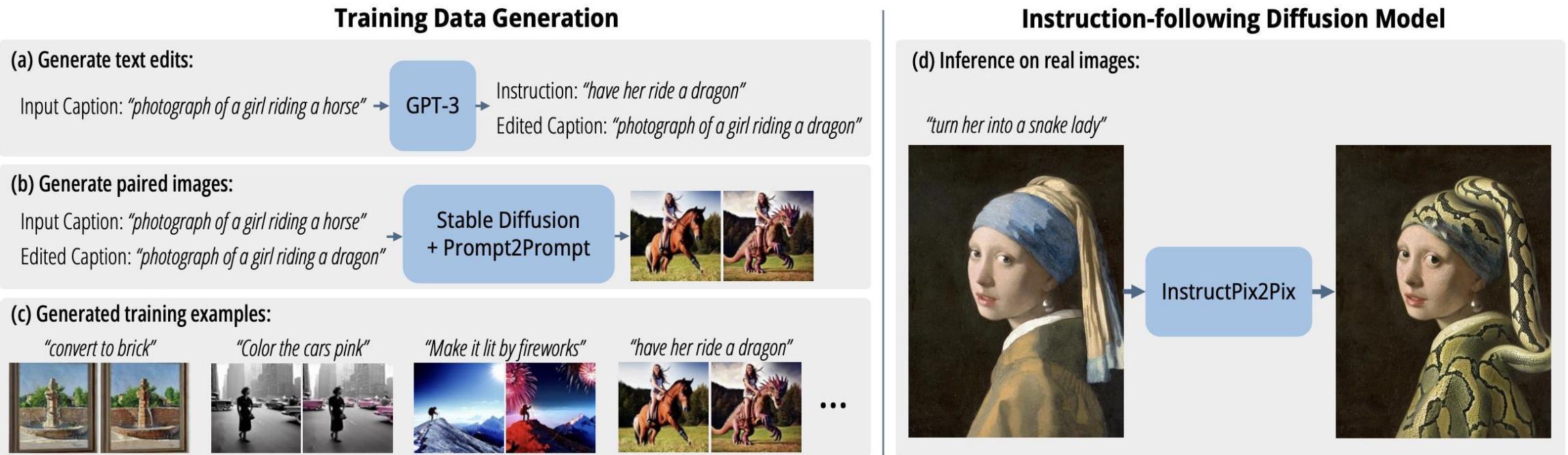
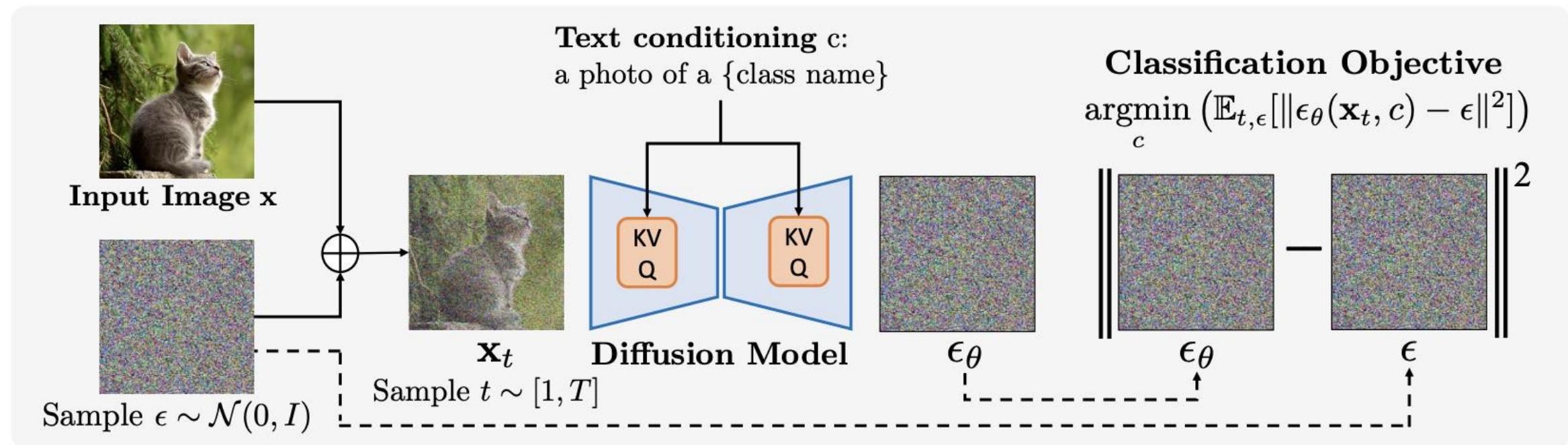


Figure 1. Given **an image** and **an instruction** for how to edit that image, our model performs the appropriate edit. Our model does not require full descriptions for the input or output image, and edits images in the forward pass without per-example inversion or fine-tuning.

InstructPix2Pix: Learning to Follow Image Editing Instructions



Your Diffusion Model is Secretly a Zero-Shot Classifier



Versatile Diffusion: All in One

A dream of a
village in China, by
Caspar David
Friedrich, matte
painting trending
on artstation-HQ.

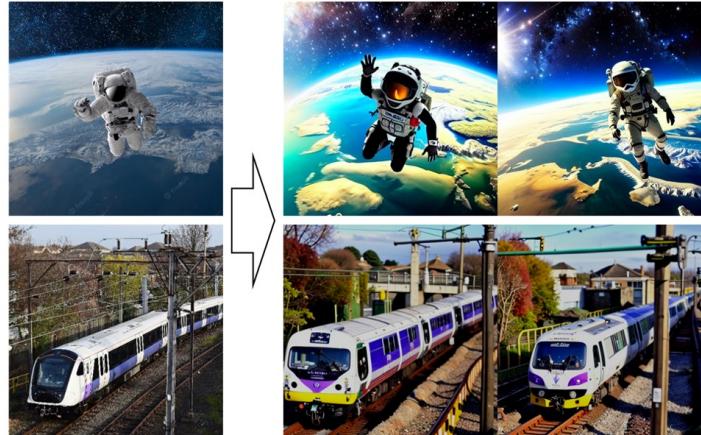
Grand nebula in the universe.



(a) Text-to-Image

- There are stars that a child is watching about.
 - Two young girls and a boy standing near a star.
 - Two young girls are watching a star.
 - Kids standing for their stars.

- Houses on the lake with boats and trees beside there with the mountains on the background.
 - House, mountain, boat, somewhere near lake
 - House on the cliff near the lake.
 - Houses on the lake with the trees.



(b) Image-Variation



(c) Image-to-Text



Semantic

Style

(d) Disentanglement



A picture in oil painting style.

(e) Dual-Guided Generation



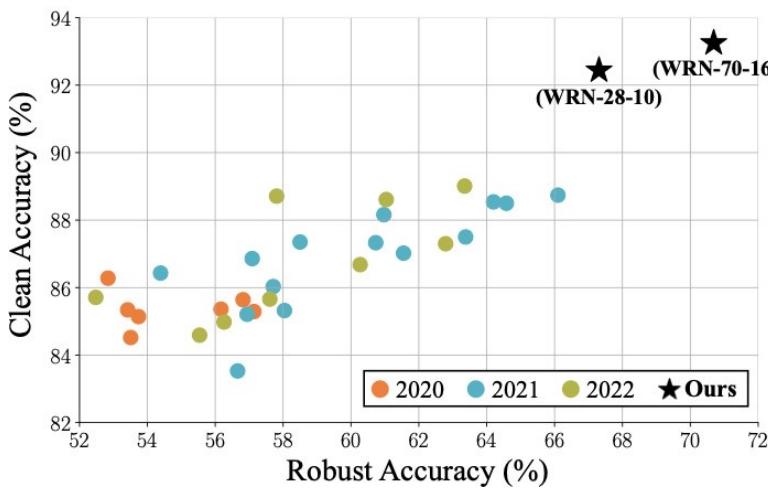
A house on
lake.

A **house** on a lake.
tall castle

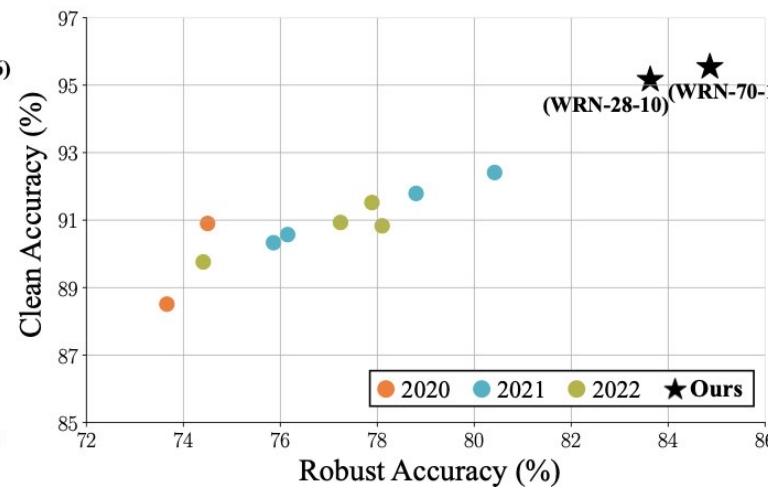
(f) Editable I2T2I

Better Diffusion Models Improve Adversarial Training

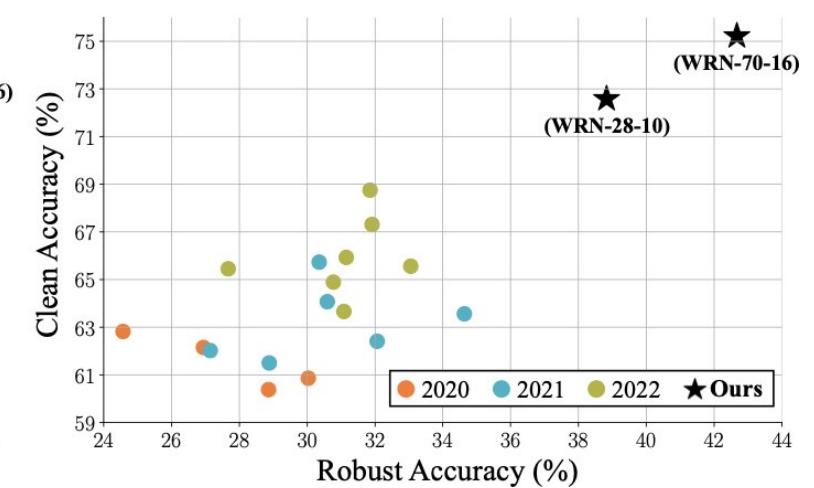
CIFAR-10($\ell_\infty, \epsilon = 8/255$)



CIFAR-10($\ell_2, \epsilon = 128/255$)



CIFAR-100($\ell_\infty, \epsilon = 8/255$)



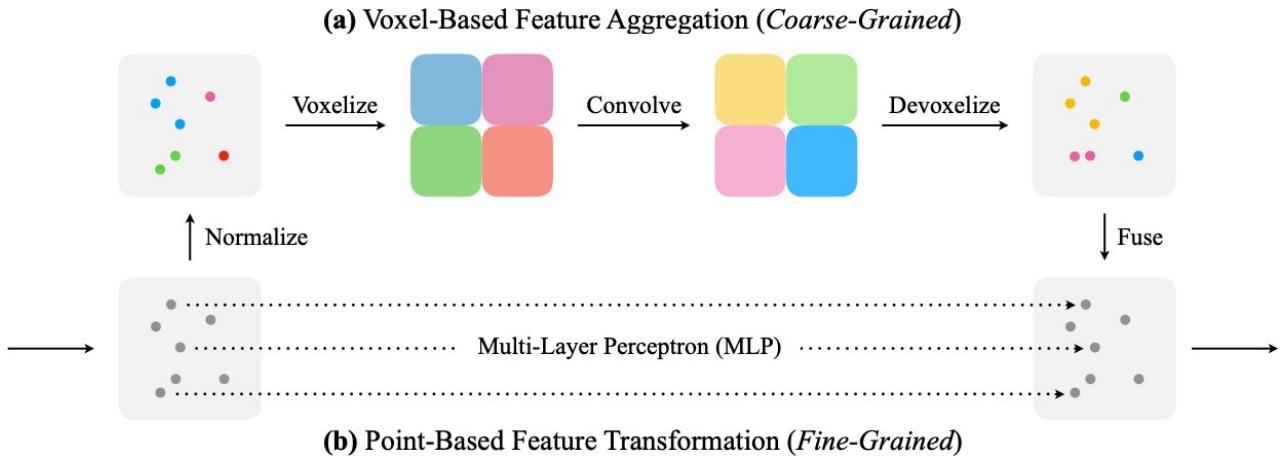
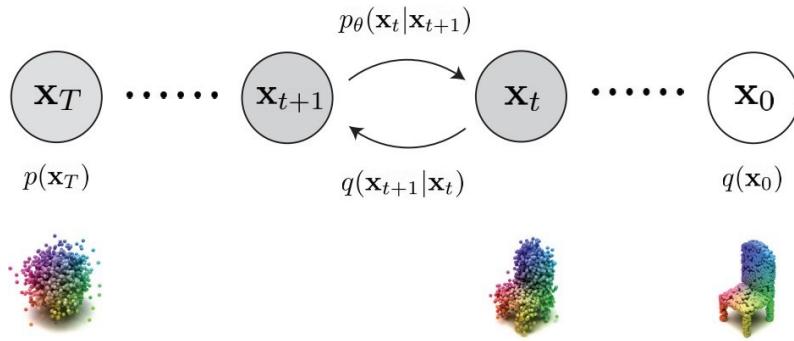
From 2D to 3D: A “natural” idea?



- Train a 3D diffusion model just like image diffusion model
 - Design proper diffusion architecture (still UNet?)
 - Choose proper 3D representation
 - Collect large 3D training corpus

Diffusion Models for Point Clouds

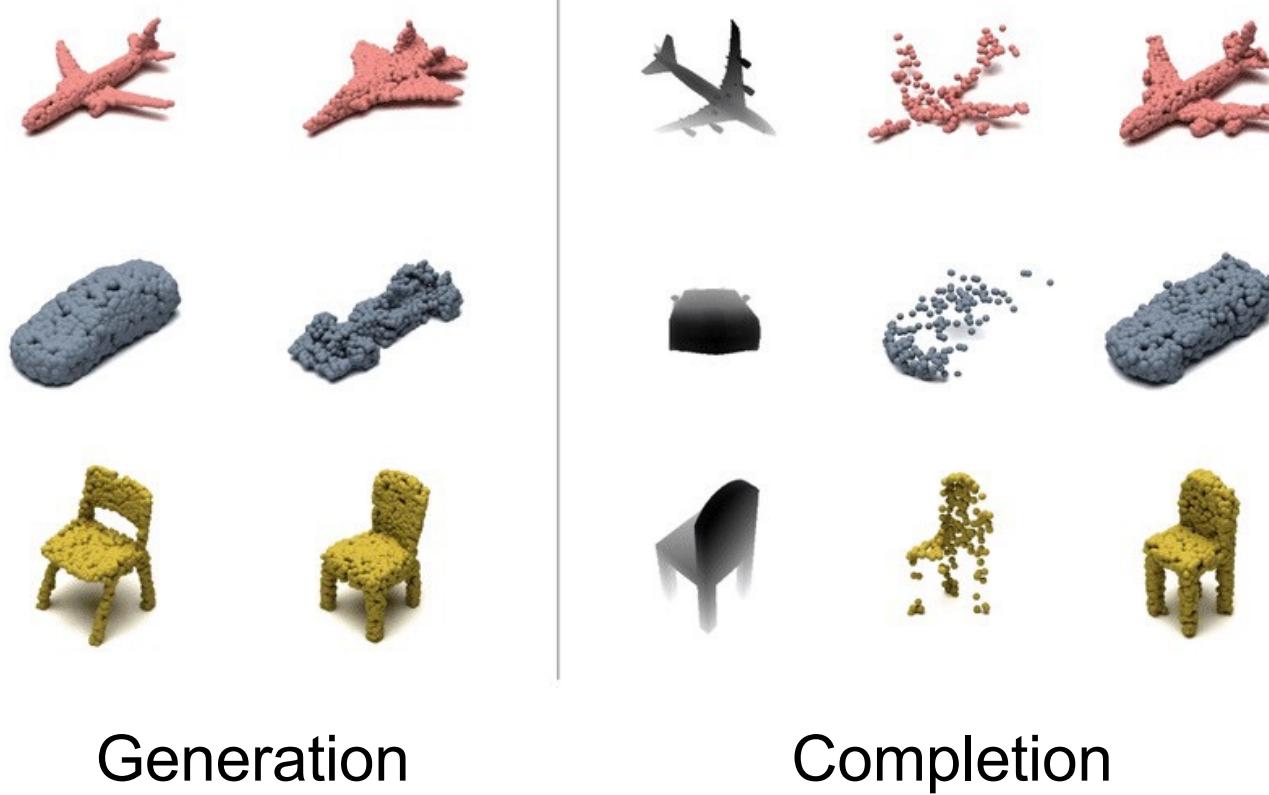
A set of points with location information.



Procedure

Point-Voxel CNN architecture

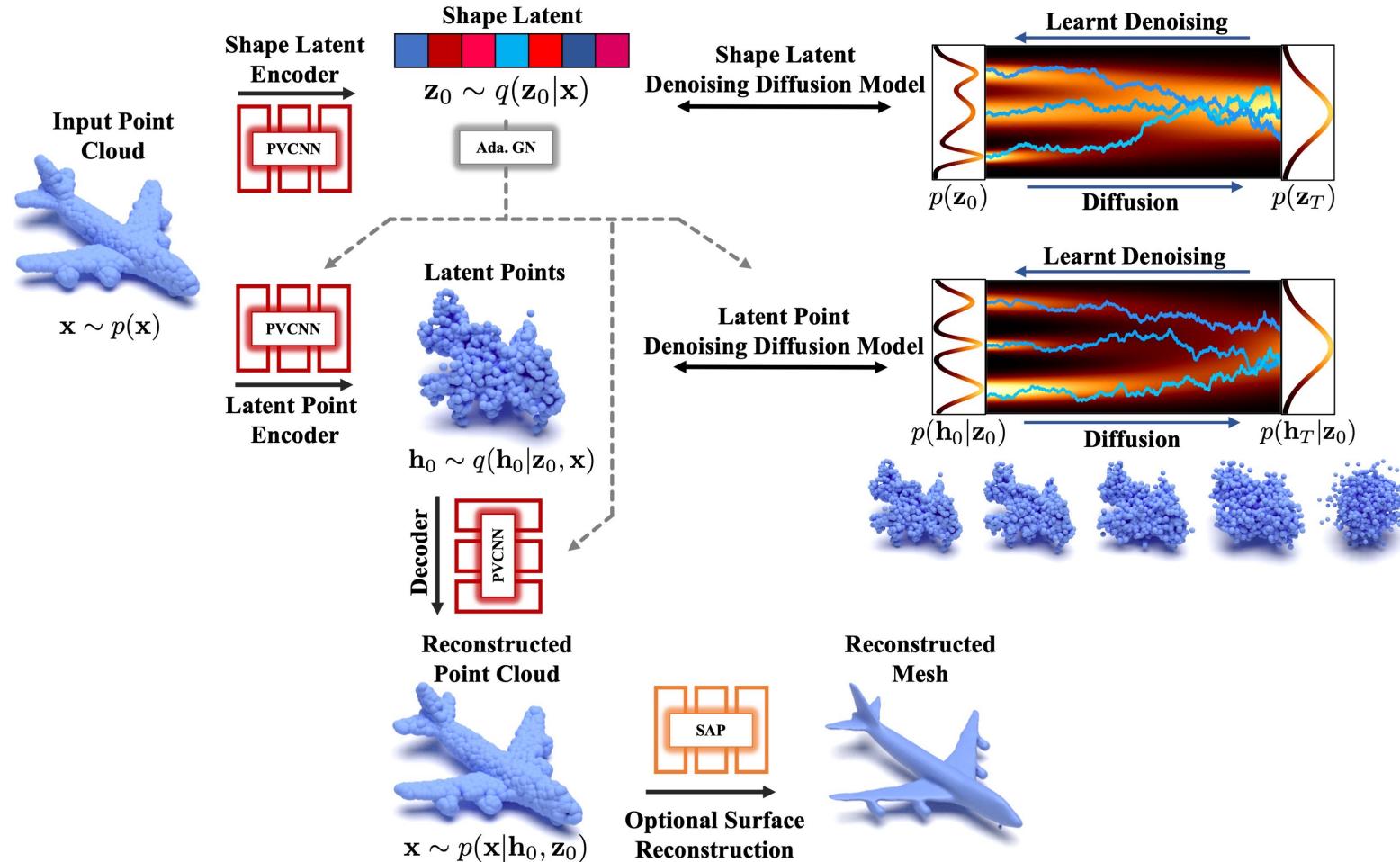
Diffusion Models for Point Clouds



Generation

Completion

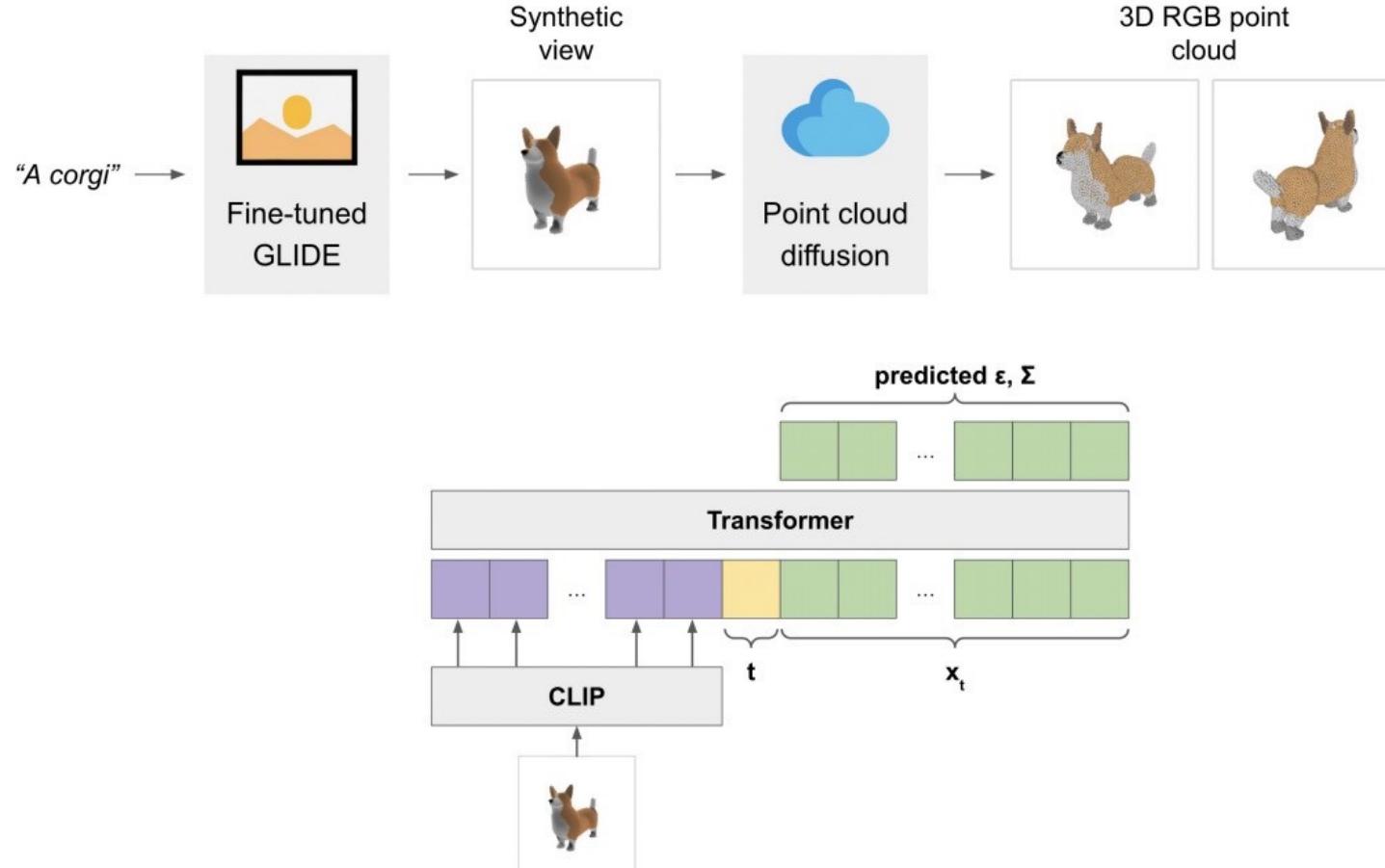
Diffusion Models for Point Clouds



Point cloud diffusion in the latent space

Diffusion Models for Point Clouds

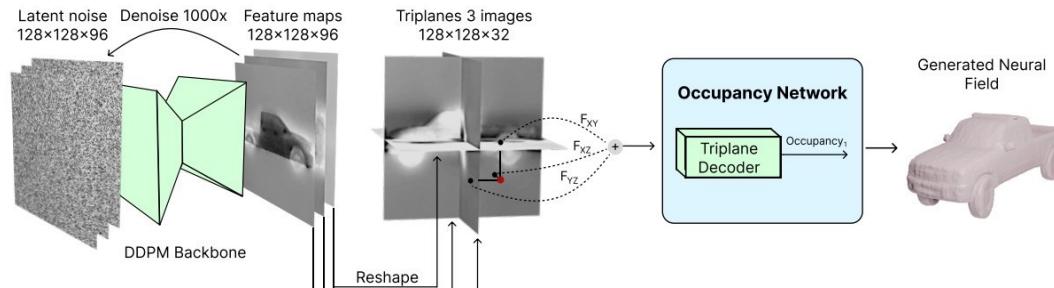
Point-E uses a synthetic view from fine-tuned GLIDE, and then "lifts" the image to a 3d point cloud.



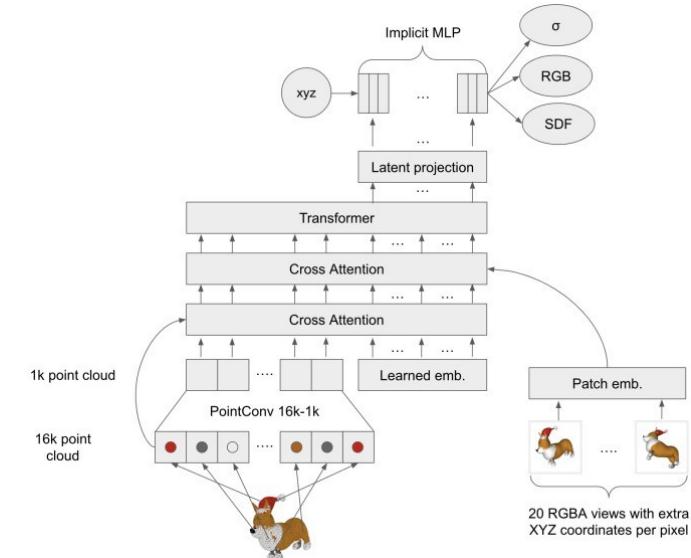
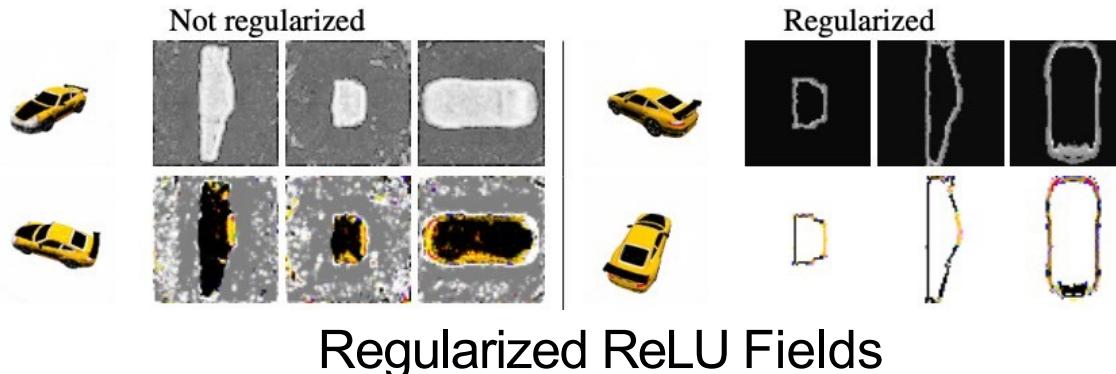
A transformer-based architecture

Diffusion Models for Other 3D Representations

- Triplanes, regularized ReLU Fields, the MLP of NeRFs...
- A good representation is important!



Triplane diffusion



Implicit MLP of NeRFs

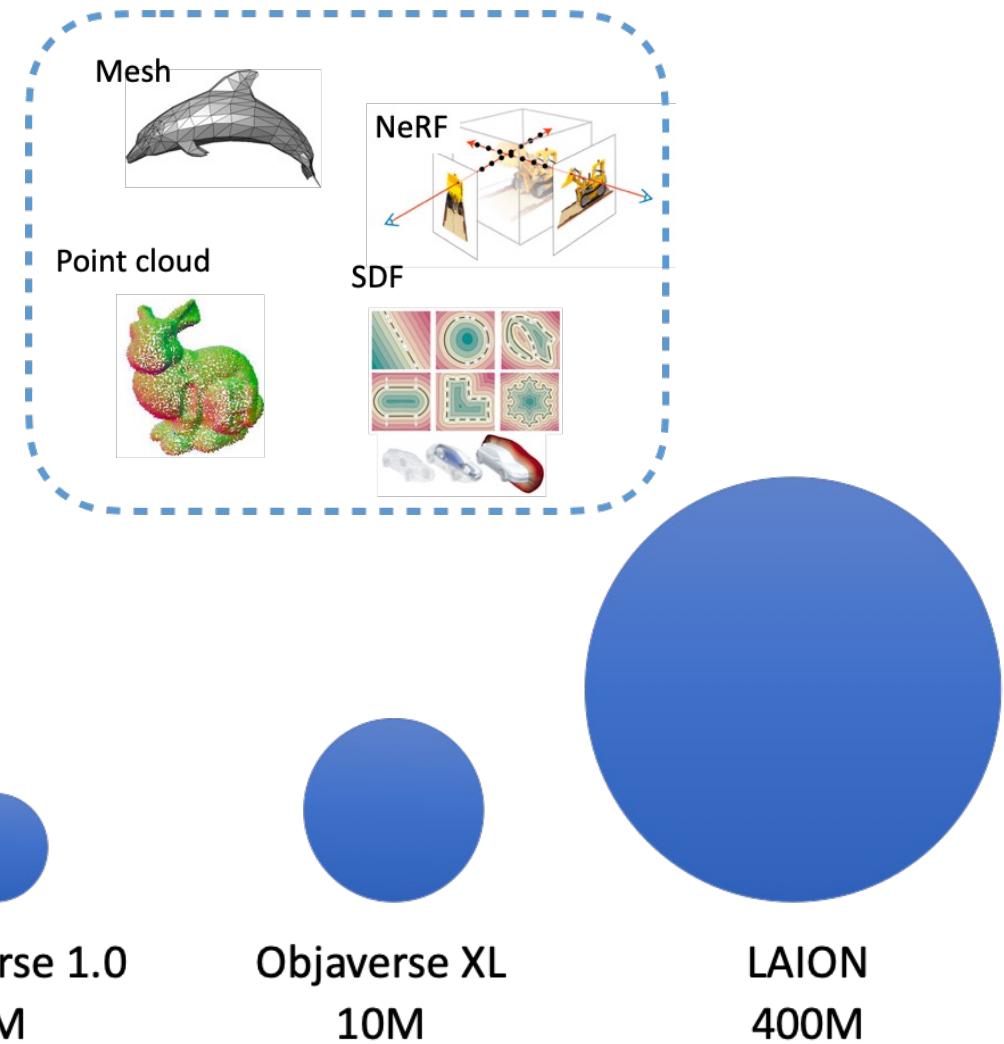
Shue et al., "[3D Neural Field Generation using Triplane Diffusion](#)", arXiv 2022

Yang et al., "[Learning a Diffusion Prior for NeRFs](#)", ICLR Workshop 2023

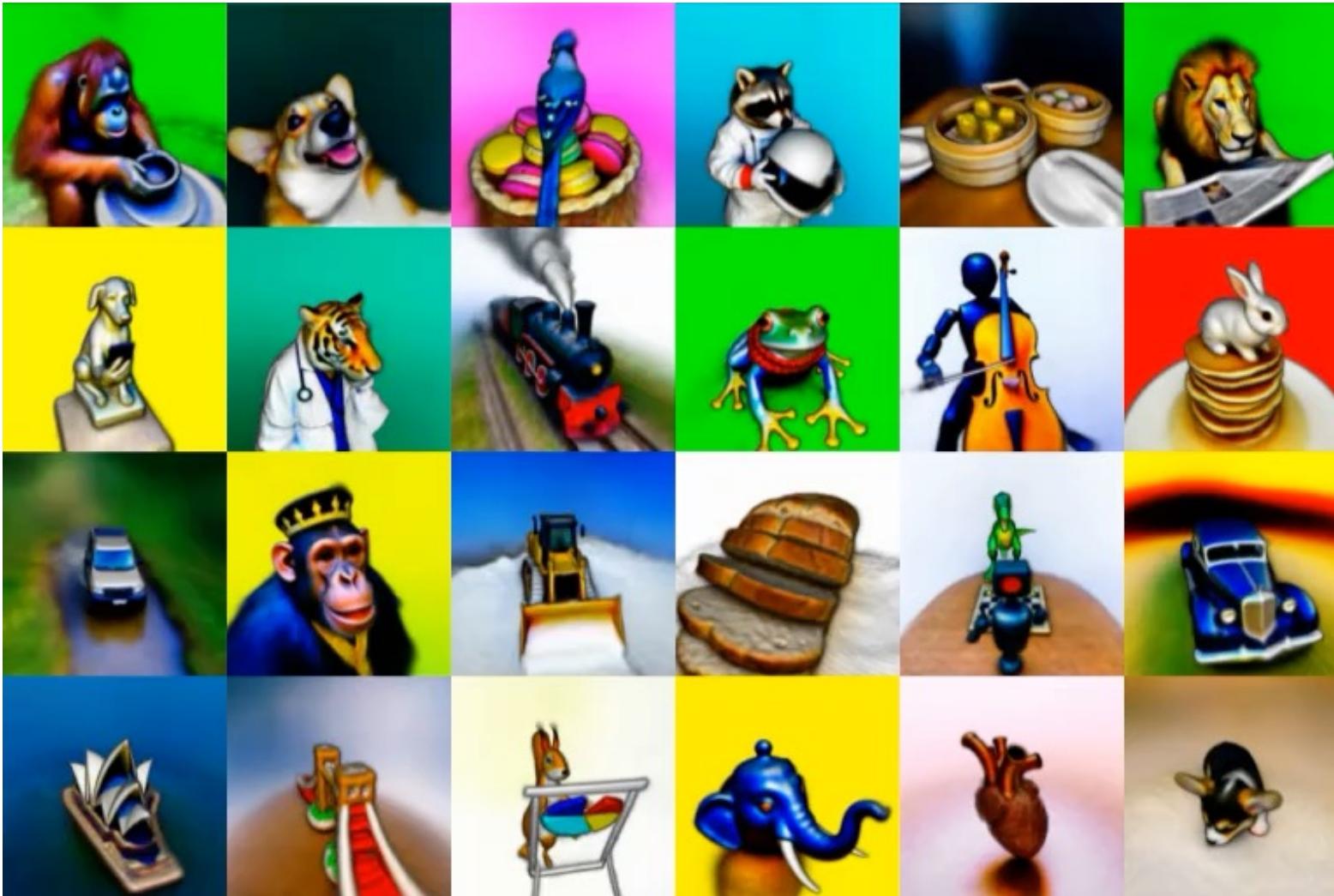
Jun and Nichol, "[Shap-E: Generating Conditional 3D Implicit Functions](#)", arXiv 2023

2D Diffusion Models for 3D Generation

- We just discussed diffusion models directly on 3d
- However:
- Design neural architecture for 3D domain is harder
 - 3D data are way more flexible in representation and data preprocessing is heavily demanded
 - A sufficiently large 3D dataset is less realistic at present (too many experiments on ShapeNet!)
- Can we use 2d diffusion models as a “prior” for 3d?



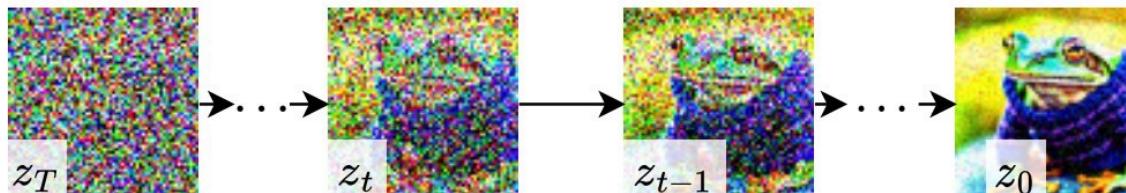
DreamFusion: where it all started



DreamFusion: Setup

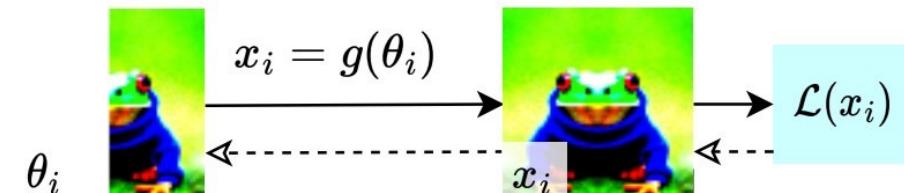
- Suppose there is a text-to-image diffusion model.
- Goal: optimize NeRF parameter such that each angle “looks good” from the text-to-image model.
- Unlike ancestral sampling (e.g., DDIM), the underlying parameters are being optimized over some loss function.

Ancestral Sampling



Updates sample in **pixel space**: $z_{t-1} = \text{ddpm_update}(z_t)$

Score Distillation Sampling



Updates **parameters** with SGD: $\theta_{i+1} = \text{opt.step}(\theta_i, \nabla_\theta \mathcal{L}(x_i))$

DreamFusion: Score Distillation Sampling

- Consider the diffusion model objective for a sample \mathbf{x} :

$$\mathcal{L}_{\text{Diff}}(\phi, \mathbf{x}) = \mathbb{E}_{t \sim \mathcal{U}(0,1), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} [w(t) \|\epsilon_\phi(\alpha_t \mathbf{x} + \sigma_t \epsilon; t) - \epsilon\|_2^2] ,$$

- Directly computing the gradient leads to a Jacobian term over the U-Net:

$$\nabla_\theta \mathcal{L}_{\text{Diff}}(\phi, \mathbf{x} = g(\theta)) = \mathbb{E}_{t, \epsilon} \left[w(t) \underbrace{(\hat{\epsilon}_\phi(\mathbf{z}_t; y, t) - \epsilon)}_{\text{Noise Residual}} \underbrace{\frac{\partial \hat{\epsilon}_\phi(\mathbf{z}_t; y, t)}{\mathbf{z}_t}}_{\text{U-Net Jacobian}} \underbrace{\frac{\partial \mathbf{x}}{\partial \theta}}_{\text{Generator Jacobian}} \right] \quad \mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \mathbf{c}$$

- However, it turns out we can consider removing the U-Net Jacobian!

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, \mathbf{x} = g(\theta)) \triangleq \mathbb{E}_{t, \epsilon} \left[w(t) (\hat{\epsilon}_\phi(\mathbf{z}_t; y, t) - \epsilon) \frac{\partial \mathbf{x}}{\partial \theta} \right]$$

DreamFusion: Score Distillation Sampling

- Given image distribution as diffusion model: $\epsilon_t(x|y) \approx \nabla \log p_t(x|y)$
- Maximal log-likelihood estimation: $\min_{\theta} \mathbb{E}_{t,c}[-\log p_t(g(\theta, c)|y)]$

$$L_{MLE} = -\log p(x|y) \leq \mathbb{E}_{t,\varepsilon} [w(t) \|\epsilon_t(\alpha_t x + \sigma_t \varepsilon | y) - \varepsilon\|_2^2] = L_{DSM}(x) \quad (\text{ELBO})$$

$$\nabla_{\theta} L_{DSM}(x = g(\theta, c)) = \mathbb{E}_{t,\varepsilon} \left[w(t) (\epsilon_t(\tilde{x}|y) - \varepsilon) \frac{\partial \epsilon_t(\tilde{x}|y)}{\partial \tilde{x}} \frac{\partial g(\theta, c)}{\partial \theta} \right] \quad (\text{Grad.})$$

$$\nabla_{\theta} L_{SDS}(x = g(\theta, c)) = \mathbb{E}_{t,\varepsilon} \left[w(t) (\epsilon_t(\tilde{x}|y) - \varepsilon) \frac{\partial g(\theta, c)}{\partial \theta} \right] \quad (\text{Drop UNet Jacobian})$$

DreamFusion in Text-to-3D

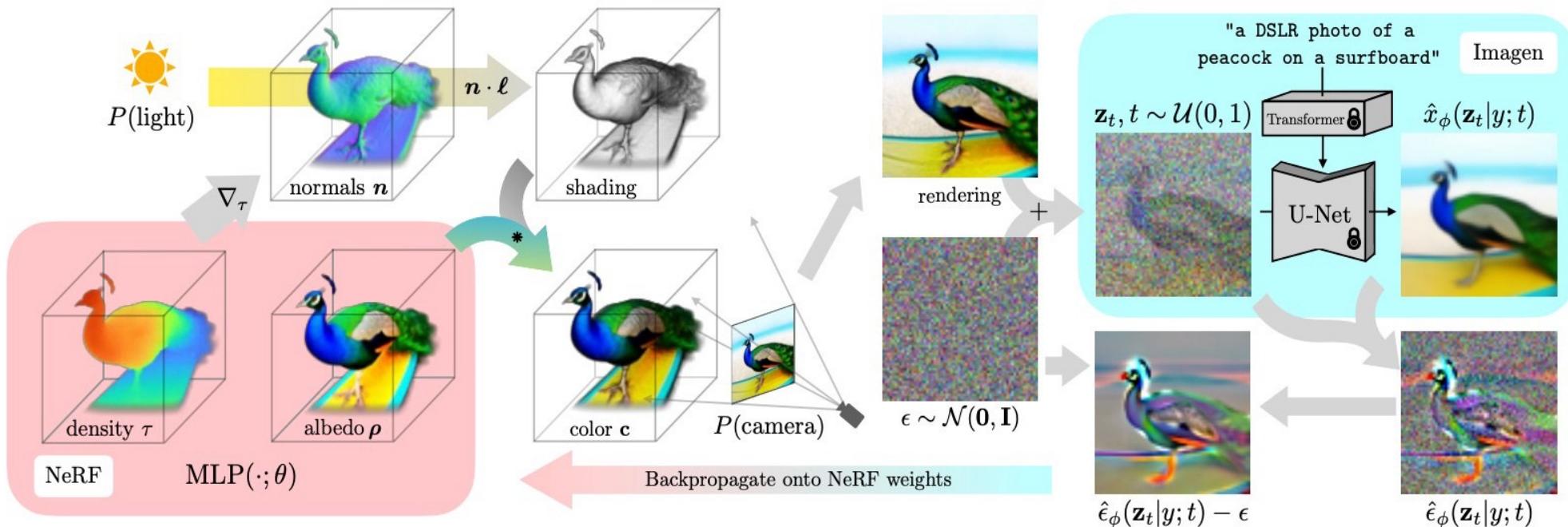
- SDS can be used to optimize a 3D representation, like NeRF.

- Random sample a camera pose and render an image: $x = g(\theta, c)$
- Sample t and $\epsilon \sim \mathcal{N}(0, I)$.
- Let $\tilde{x} = \alpha_t x + \sigma_t \epsilon$.
- Update θ via SDS gradient:

$$\nabla_{\theta} L_{SDS} = \mathbb{E}_{t,c,\epsilon} \left[w(t)(\epsilon_t(\tilde{x}|y) - \epsilon) \frac{\partial \epsilon(\tilde{x}|y)}{\partial \theta} \right]$$

- Stop gradient trick:

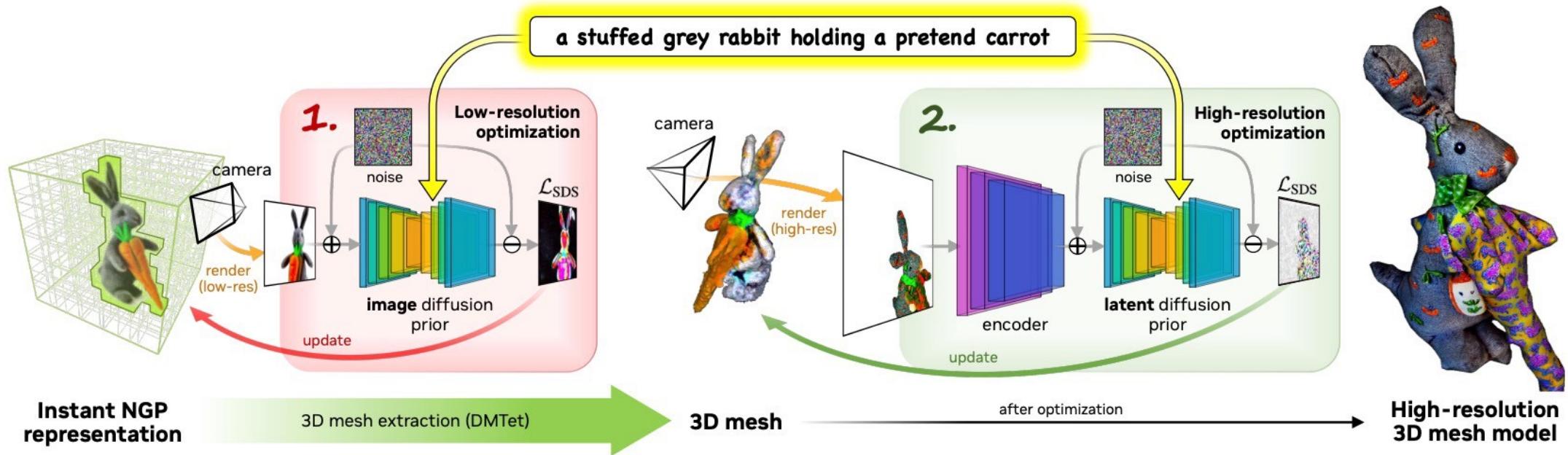
$$\nabla_{\theta} L_{SDS} = \nabla_{\theta} \mathbb{E}_{t,c,\epsilon} [\text{stopgrad}[\epsilon_t(\tilde{x}|y) - \epsilon]^T g(\theta, c)]$$



Extensions to SDS: Magic3D

2x speed and higher resolution

- Accelerate NeRF with Instant-NGP, for coarse representations.
- Optimize a fine mesh model with differentiable renderer.



Alternative to SDS: Score Jacobian Chaining

A different formulation, motivated from approximating 3D score.

$$\nabla_{\theta} \log \tilde{p}_{\sigma}(\theta) = \mathbb{E}_{\pi} [\nabla_{\theta} \log p_{\sigma}(x_{\pi})]$$

$$\frac{\partial \log \tilde{p}_{\sigma}(\theta)}{\partial \theta} = \mathbb{E}_{\pi} \left[\frac{\partial \log p_{\sigma}(x_{\pi})}{\partial x_{\pi}} \cdot \frac{\partial x_{\pi}}{\partial \theta} \right]$$

$$\underbrace{\nabla_{\theta} \log \tilde{p}_{\sigma}(\theta)}_{\text{3D score}} = \mathbb{E}_{\pi} [\underbrace{\nabla_{x_{\pi}} \log p_{\sigma}(x_{\pi})}_{\text{2D score; pretrained}} \cdot \underbrace{J_{\pi}}_{\text{renderer Jacobian}}].$$

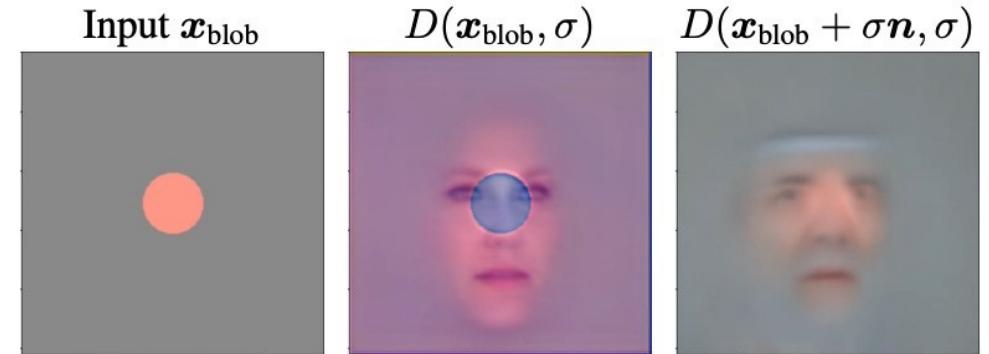
In principle, the diffusion model is the noisy 2D score (over clean images),
but in practice, the diffusion model suffers from out-of-distribution (OOD) issues!

For diffusion model on noisy images, the non-noisy images are OOD!

Score Jacobian Chaining

- SJC approximates noisy score with “Perturb-and-Average Scoring”, which is not present in SDS.
- Use score model on multiple noise-perturbed data, then average it.

$$\begin{aligned} \text{PAAS}(\mathbf{x}_\pi, \sqrt{2}\sigma) &\triangleq \mathbb{E}_{\mathbf{n} \sim \mathcal{N}(0, \mathbf{I})} [\text{score}(\mathbf{x}_\pi + \sigma\mathbf{n}, \sigma)] \\ &= \mathbb{E}_{\mathbf{n}} \left[\frac{D(\mathbf{x}_\pi + \sigma\mathbf{n}, \sigma) - (\mathbf{x}_\pi + \sigma\mathbf{n})}{\sigma^2} \right] \\ &= \mathbb{E}_{\mathbf{n}} \left[\frac{D(\mathbf{x}_\pi + \sigma\mathbf{n}, \sigma) - \mathbf{x}_\pi}{\sigma^2} \right] - \underbrace{\mathbb{E}_{\mathbf{n}} \left[\frac{\mathbf{n}}{\sigma} \right]}_{=0}. \\ \text{PAAS}(\mathbf{x}_\pi, \sqrt{2}\sigma) &\approx \nabla_{\mathbf{x}_\pi} \log p_{\sqrt{2}\sigma}(\mathbf{x}_\pi). \end{aligned}$$



PAAS helps guide updates with a better score.

Advancements on score distillation

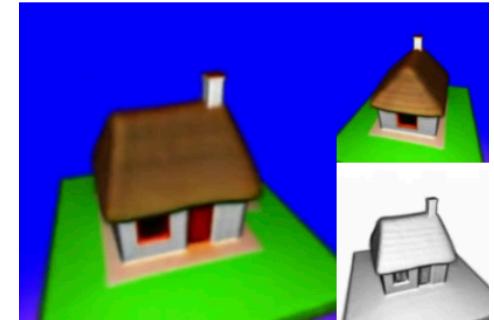
- ProlificDreamer - Variational Score Distillation (VSD):

$$\bullet \quad \nabla_{\theta} L_{VSD} = \mathbb{E}_{t,c,\epsilon} \left[w(t)(\epsilon_t(\tilde{x} | y) - \epsilon_t^{LoRA}(\tilde{x} | c, y)) \frac{\partial g(\theta, c)}{\partial \theta} \right]$$

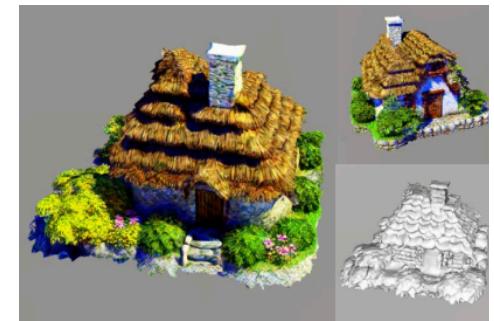
- where $\epsilon_t^{LoRA}(\tilde{x} | c, y)$ is camera pose conditioned and fine-tuned from pre-trained SD using LoRA.

- Wasserstein gradient flow to minimize KL divergence between noisy rendered image distribution and perturbed 2D image distribution.

DreamFusion



ProlificDreamer



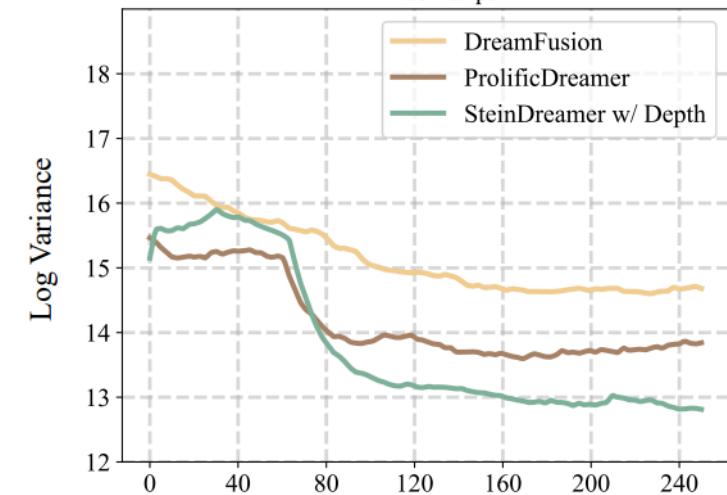
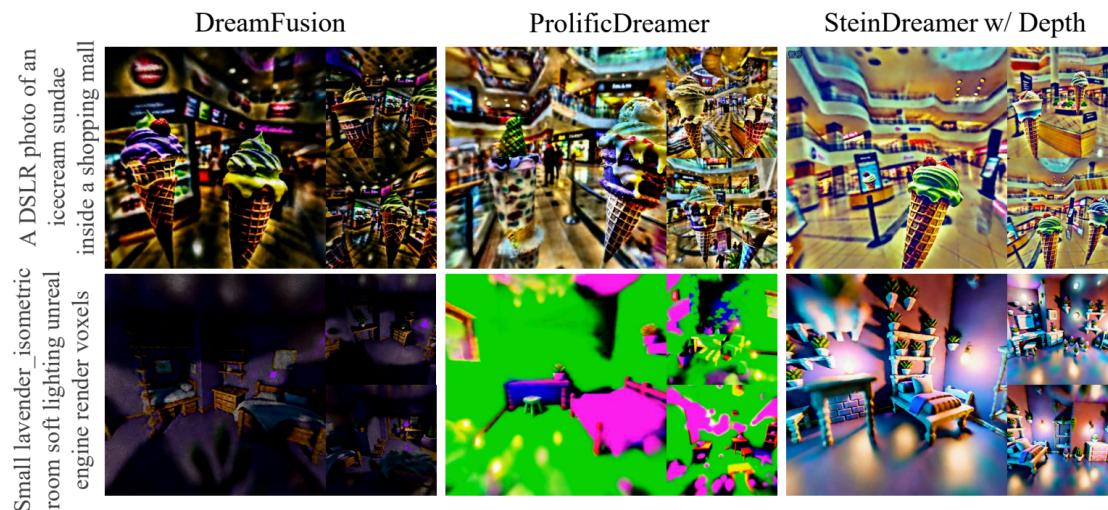
ProlificDreamer produces sharper and more photorealistic textures.

Advancements on score distillation

- SteinDreamer - Stein Score Distillation (SSD)

$$\bullet \nabla_{\theta} L_{SSD} = \mathbb{E}_{t,c,\varepsilon} \left[w(t)(\epsilon_t(\tilde{x} - y) + \varepsilon \phi(\tilde{x}, \theta, c) + \nabla \phi(\tilde{x}, \theta, c)) \frac{\partial g(\theta, c)}{\partial \theta} \right]$$

- Control variate method via Stein identity for variance reduction on gradient estimation.

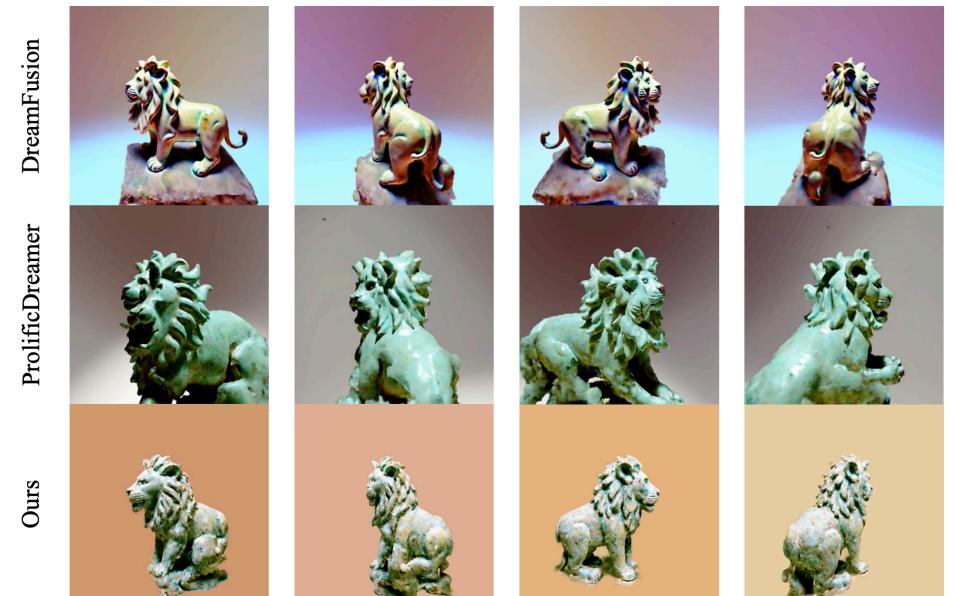
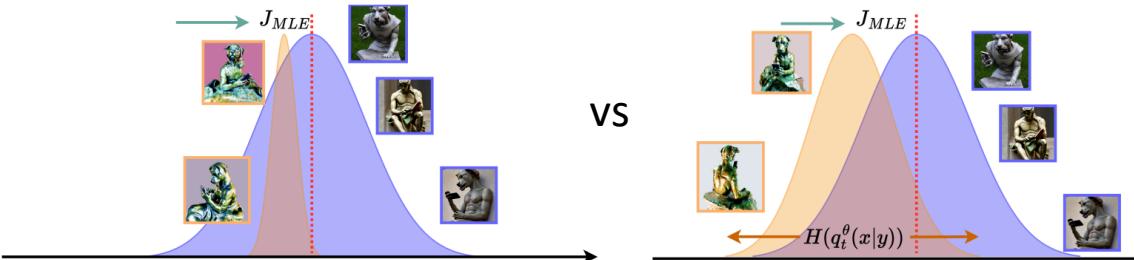


Advancements on score distillation

- Entropic Score Distillation (ESD)

- $\nabla_{\theta} L_{ESD} = \mathbb{E}_{t,c,\epsilon} \left[w(t)(\epsilon_t(\tilde{x} | y) - \lambda \epsilon_t^{Lora}(\tilde{x} | c, y) - (1 - \lambda) \epsilon_t^{Lora}(\tilde{x} | y)) \frac{\partial g(\theta, c)}{\partial \theta} \right]$

- Recover the entropy maximization term for VSD when minimizing the KL divergence to alleviate Janus problem.



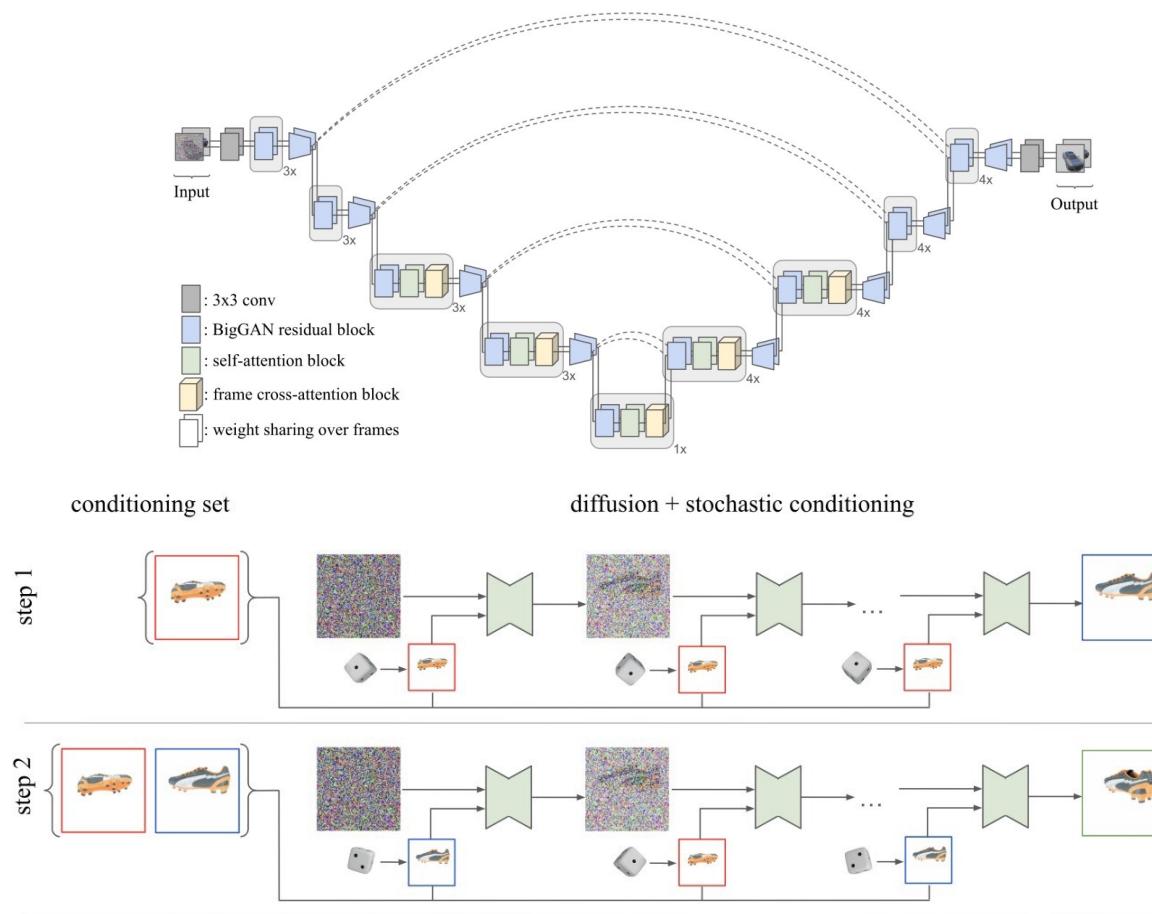
Novel-view Synthesis with Diffusion Models

- These do not produce 3D as output, but synthesis the view at different angles.



3DiM

- Condition on a frame and two poses, predict another frame.

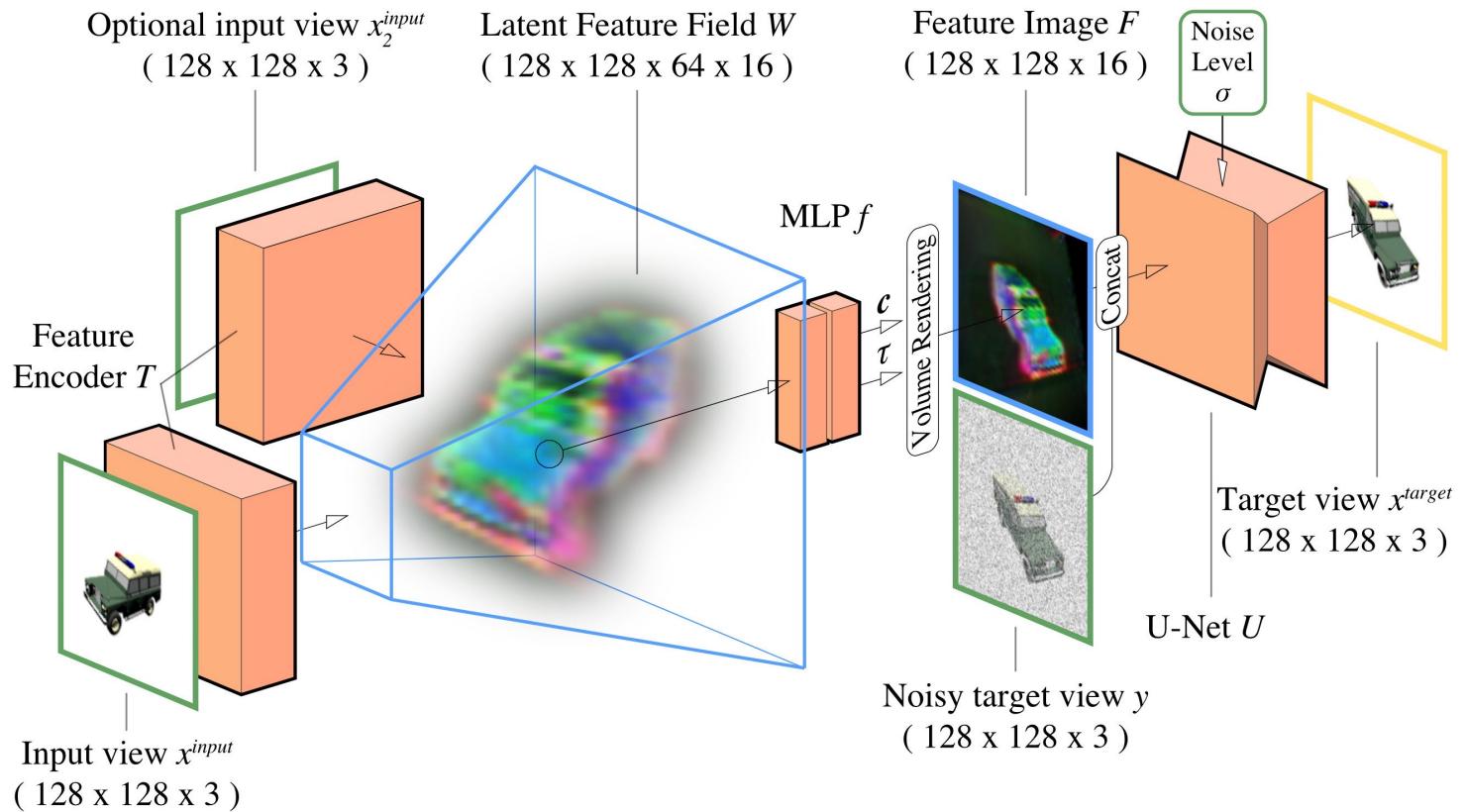


UNet with frame cross-attention

Sample based on stochastic conditions,
allowing the use of multiple conditional frames.

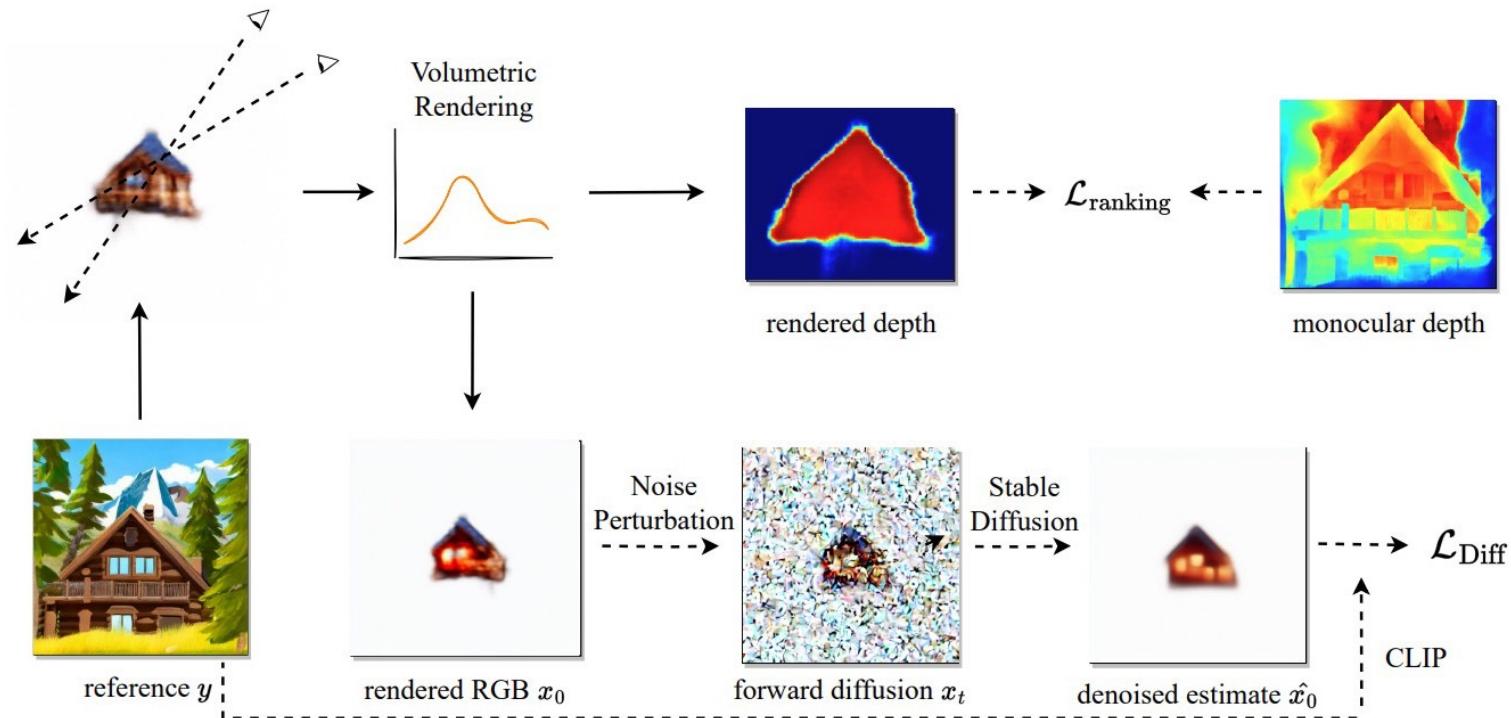
GenVS

- 3D-aware architecture with latent feature field.
- Use diffusion model to improve render quality based on structure.



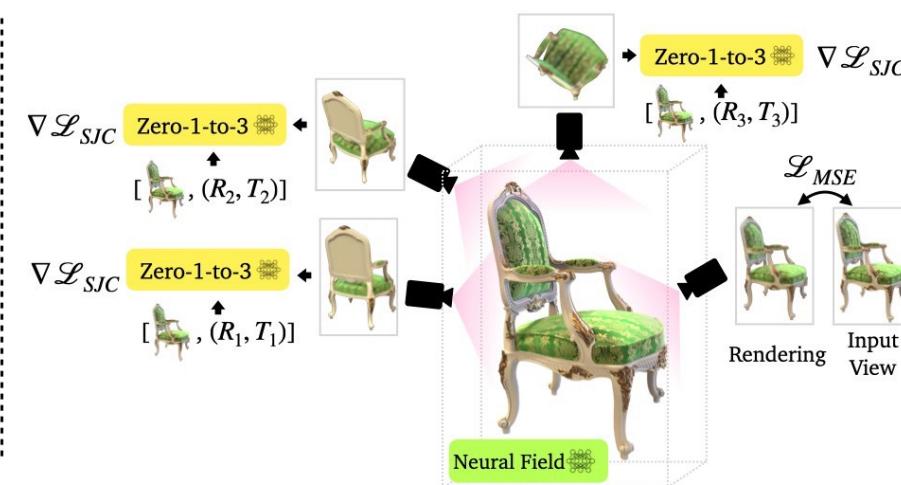
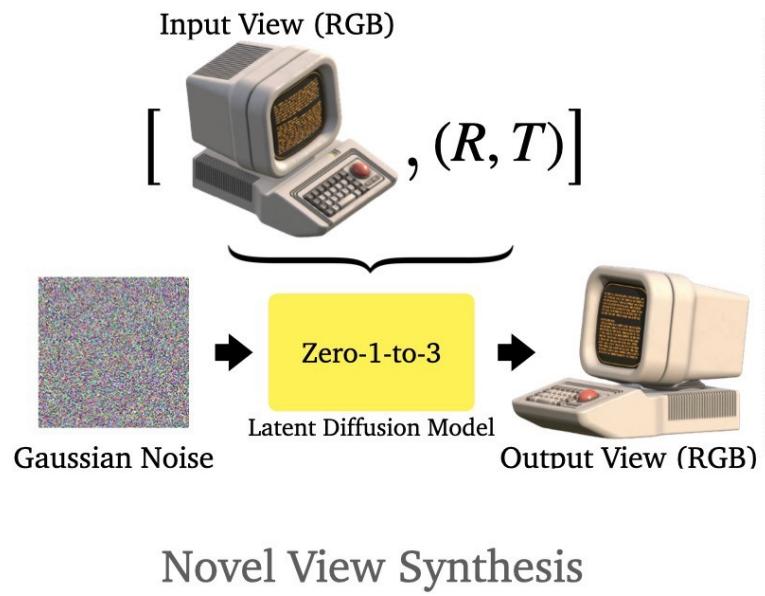
NeuralLift-360 for 3D reconstruction

- SDS + Fine-tuned CLIP text embedding + Depth supervision



Zero 1-to-3

- Generate novel view from 1 view and pose, with 2d model.
- Then, run SJC / SDS-like optimizations with camera view-conditioned model
- Training using Objaverse dataset from pre-trained SD
- Follow-up: Zero123++, MVDreamer ...



3D Reconstruction

- [1] Liu et al., Zero-1-to-3: Zero-shot One Image to 3D Object
- [2] Shi et al., Zero123++: a Single Image to Consistent Multi-view Diffusion Base Model
- [3] Shi et al., MVDream: Multi-view Diffusion for 3D Generation

Instruct NeRF2NeRF

Edit a 3D scene with text instructions



Original NeRF

*“Turn him into the
Tolkien Elf”*

*“Make it look like a
Fauvism painting”*

*“Make it look like an
Edward Munch Painting”*

*“Turn him into Lord
Voldemort”*

*“Make him look like
Vincent Van Gogh”*

Instruct NeRF2NeRF

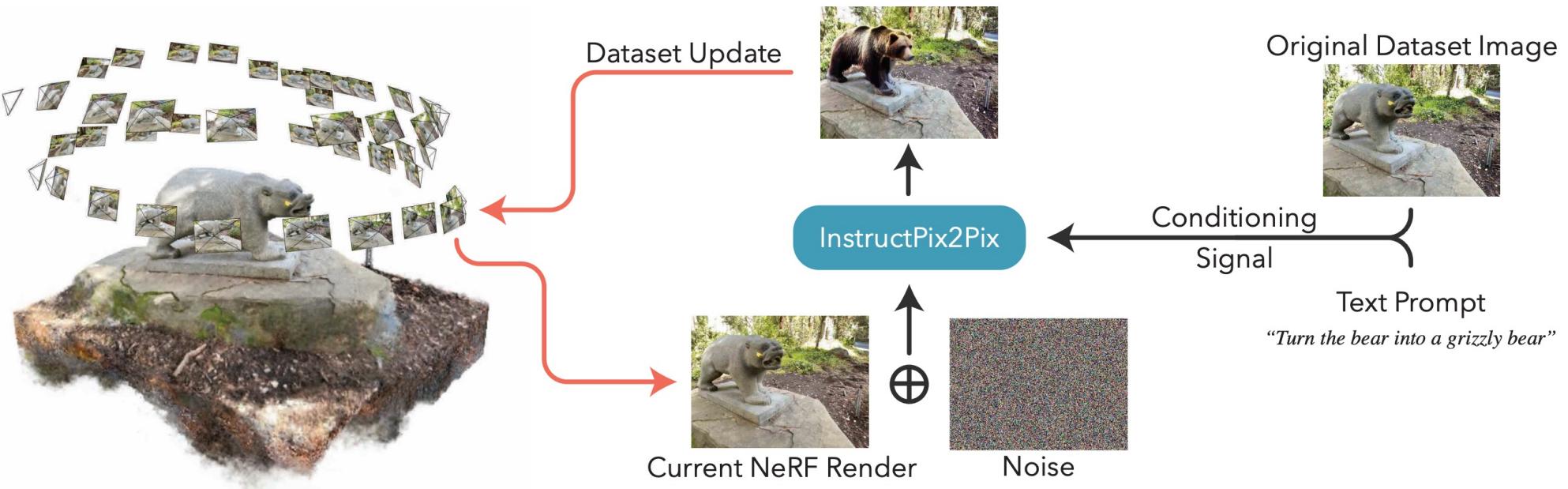
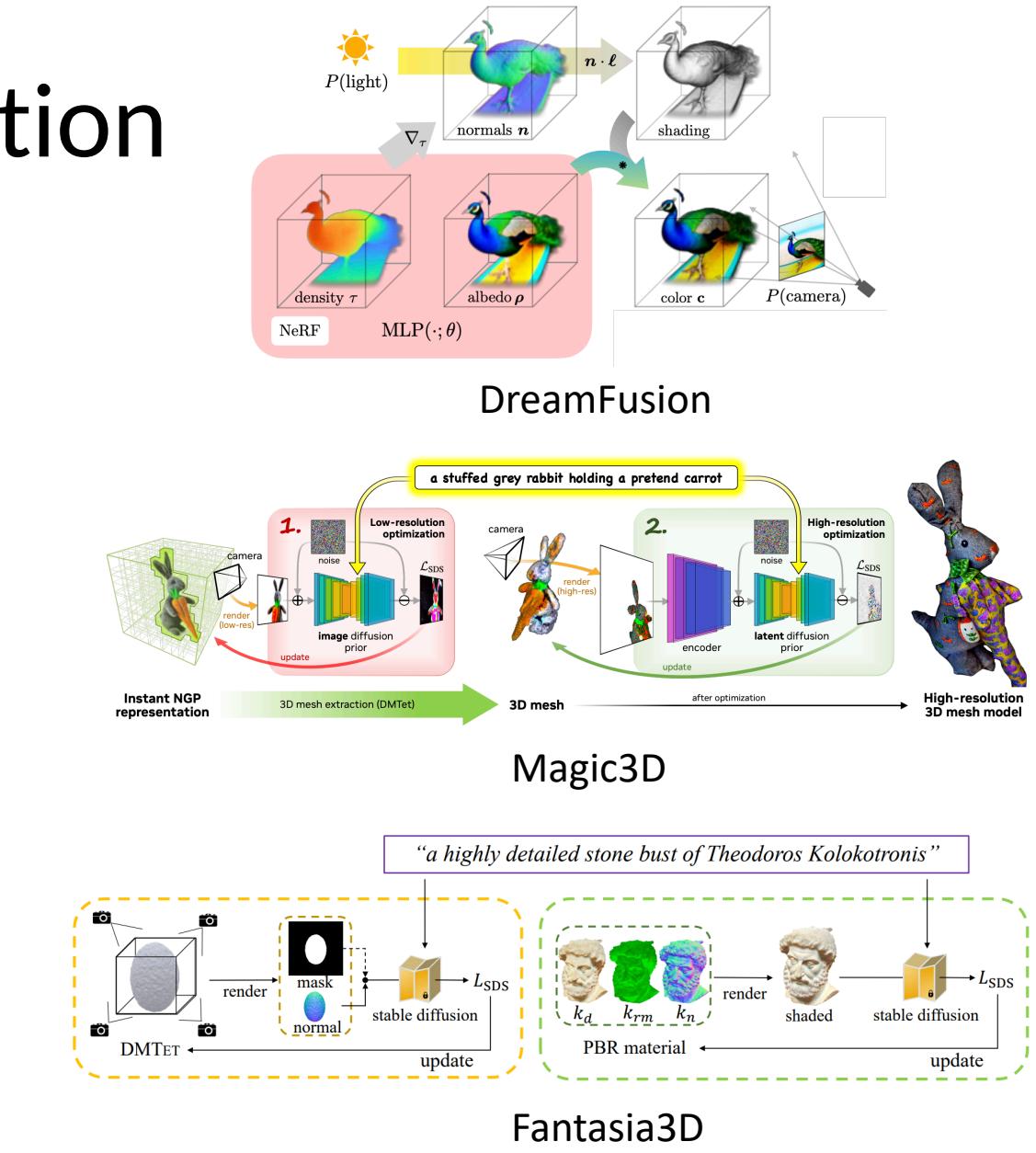


Figure 2: **Overview:** Our method gradually updates a reconstructed NeRF scene by iteratively updating the dataset images while training the NeRF: (1) an image is rendered from the scene at a training viewpoint, (2) it is edited by InstructPix2Pix given a global text instruction, (3) the training dataset image is replaced with the edited image, and (4) the NeRF continues training as usual.

Choosing 3D representation

- DreamFusion: Ref-NeRF as base representation and apply lighting augmentation.
- Magic3D [1]: coarse generation via volumetric representation (NeRF) -> refinement with differentiable mesh representation (DMTet).
- Fantasia3D [2]: disentangle 3D representation to geometry and appearance (material) properties.



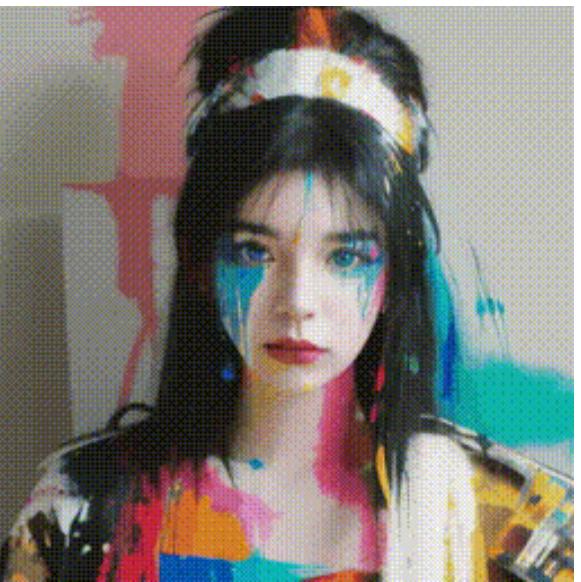
[1] Lin et al., Magic3D: High-Resolution Text-to-3D Content Creation

[2] Chen et al., Fantasia3D: Disentangling Geometry and Appearance for High-quality Text-to-3D Content Creation



Text-to-Video Generation

From Text-to-Image Diffusion Models to **Text-to-Video** Diffusion Models



Video Diffusion Models

3D UNet from a 2D UNet.

- 3x3 2d conv to 1x3x3 3d conv.
- Factorized spatial and temporal attentions.

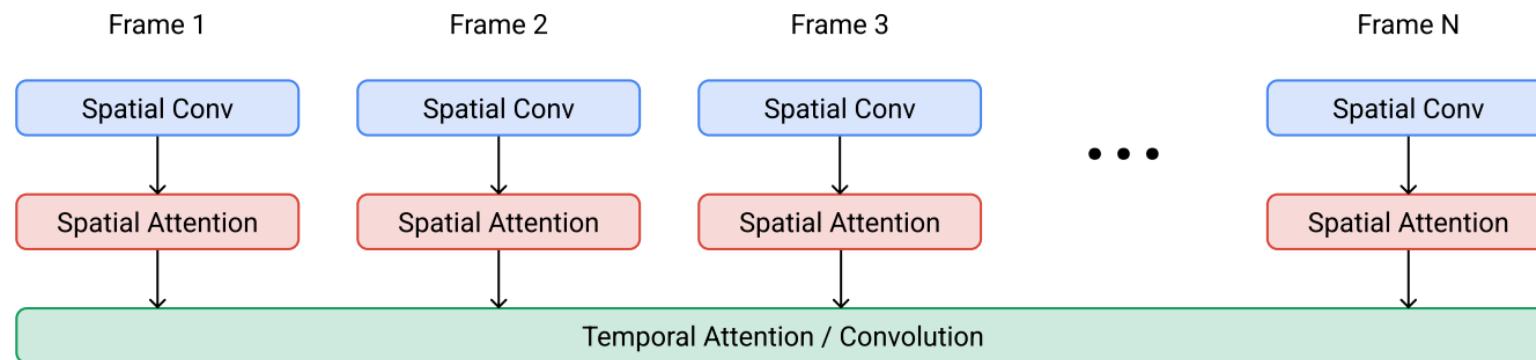
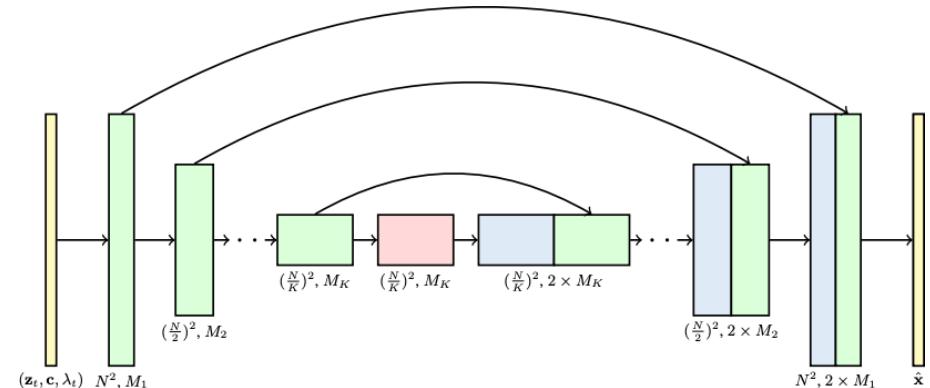
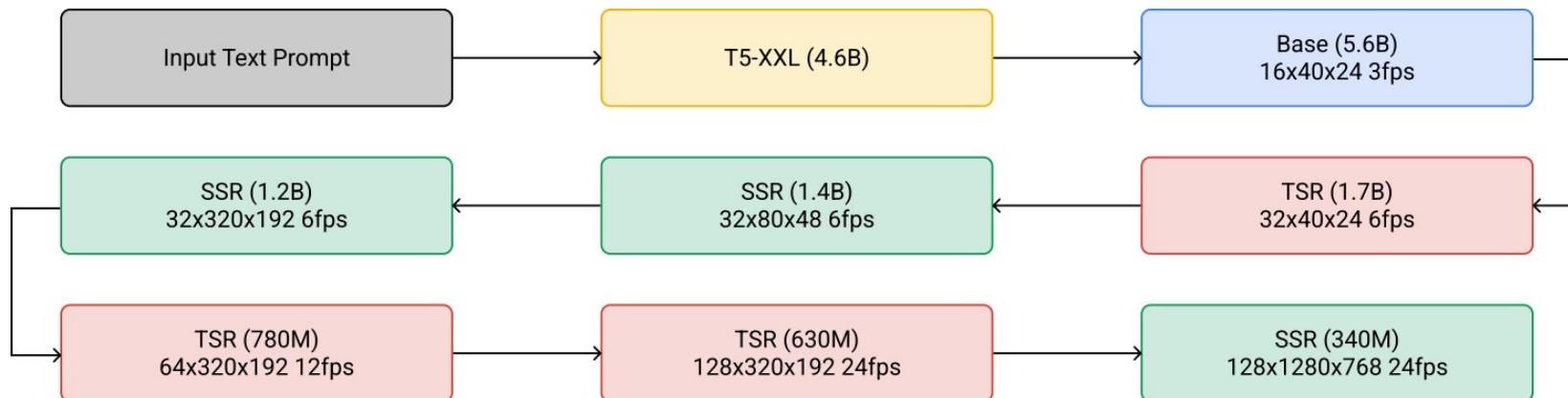


Illustration on how the 3d attention is factorized (from Imagen video)

Imagen Video: Large Scale Text-to-Video

- 7 cascade models in total.
- 1 Base model (16x40x24)
- 3 Temporal super-resolution models.
- 3 Spatial super-resolution models.



Make-A-Video: Text-to-Video Generation without Text-Video Data

Convert text into image embedding and train a video generator conditioned on image

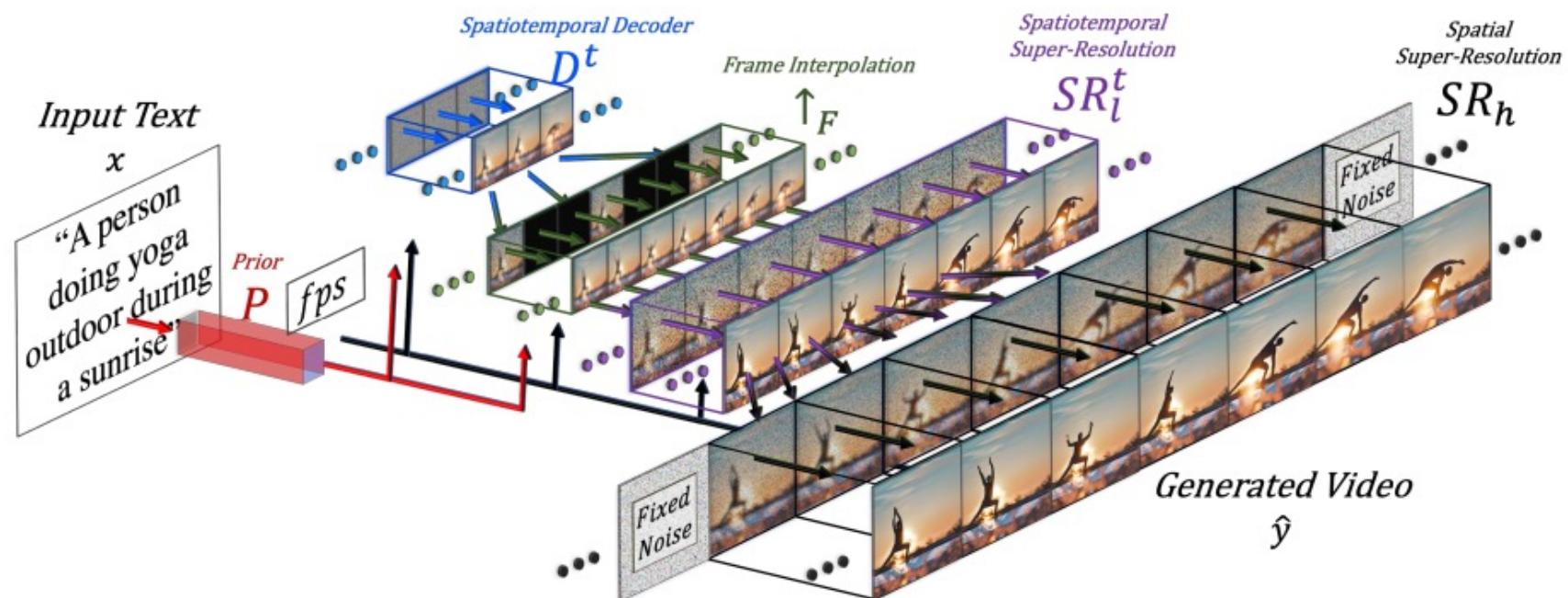


Figure 2: **Make-A-Video high-level architecture.** Given input text x translated by the prior P into an image embedding, and a desired frame rate fps , the decoder D^t generates 16 64×64 frames, which are then interpolated to a higher frame rate by \uparrow_F , and increased in resolution to 256×256 by SR_l^t and 768×768 by SR_h , resulting in a high-spatiotemporal-resolution generated video \hat{y} .

Make-A-Video: Text-to-Video Generation without Text-Video Data

Limited video data: 2.3B Text-image Pair + 20M Video Data
=> Adapt from Text-to-image Model

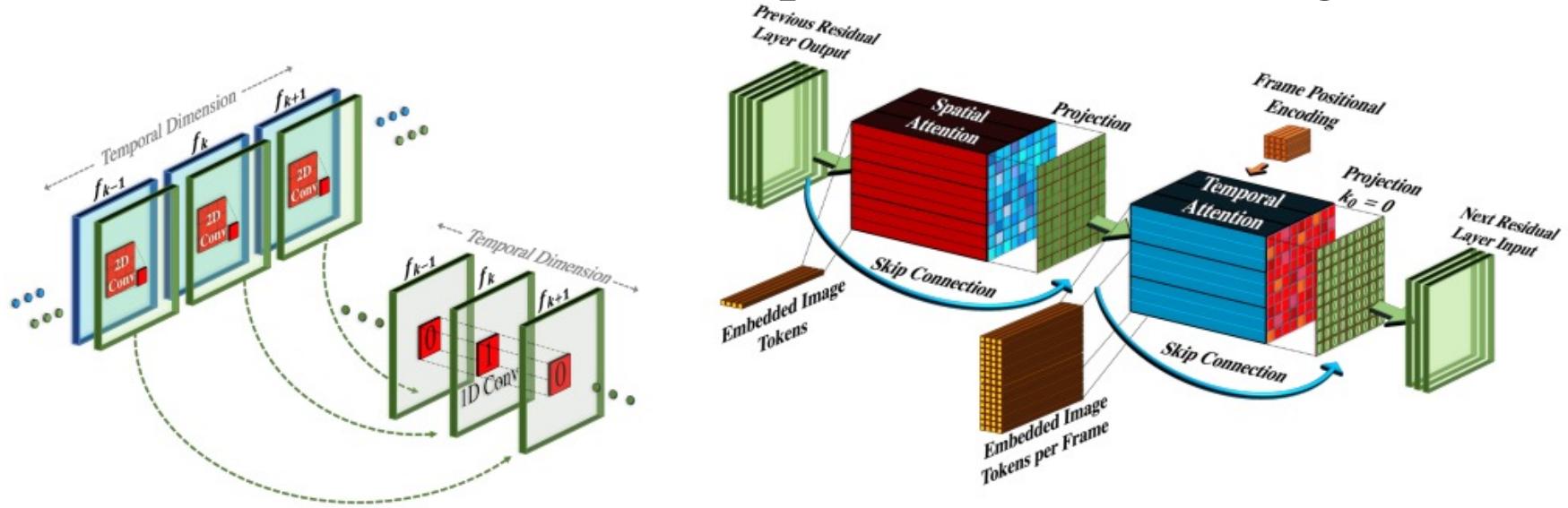
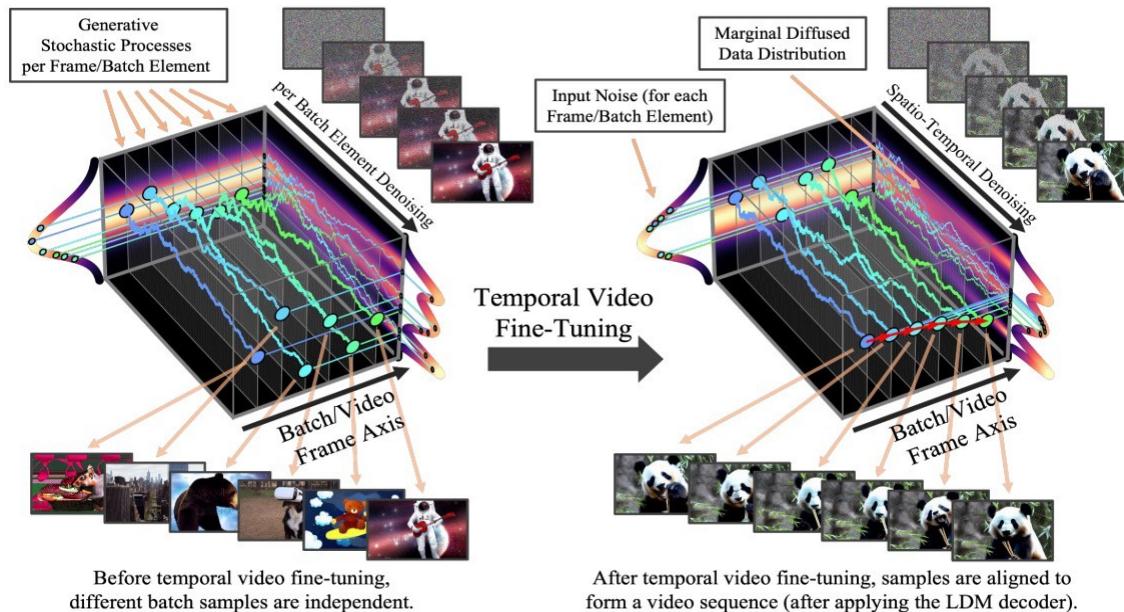
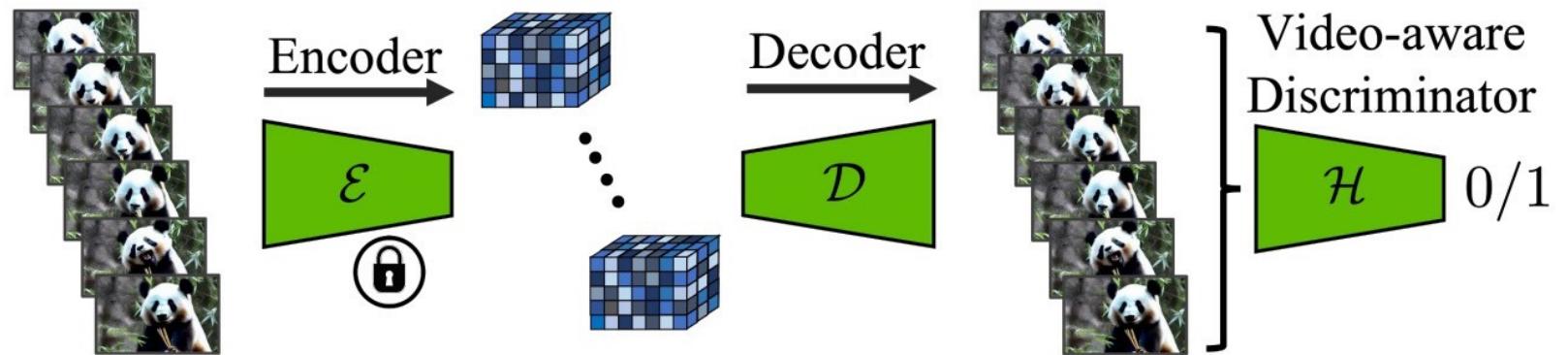


Figure 3: The architecture and initialization scheme of the Pseudo-3D convolutional and attention layers, enabling the seamless transition of a pre-trained Text-to-Image model to the temporal dimension. (left) Each spatial 2D conv layer is followed by a temporal 1D conv layer. The temporal conv layer is initialized with an identity function. (right) Temporal attention layers are applied following the spatial attention layers by initializing the temporal projection to zero, resulting in an identity function of the temporal attention blocks.

Video LDM

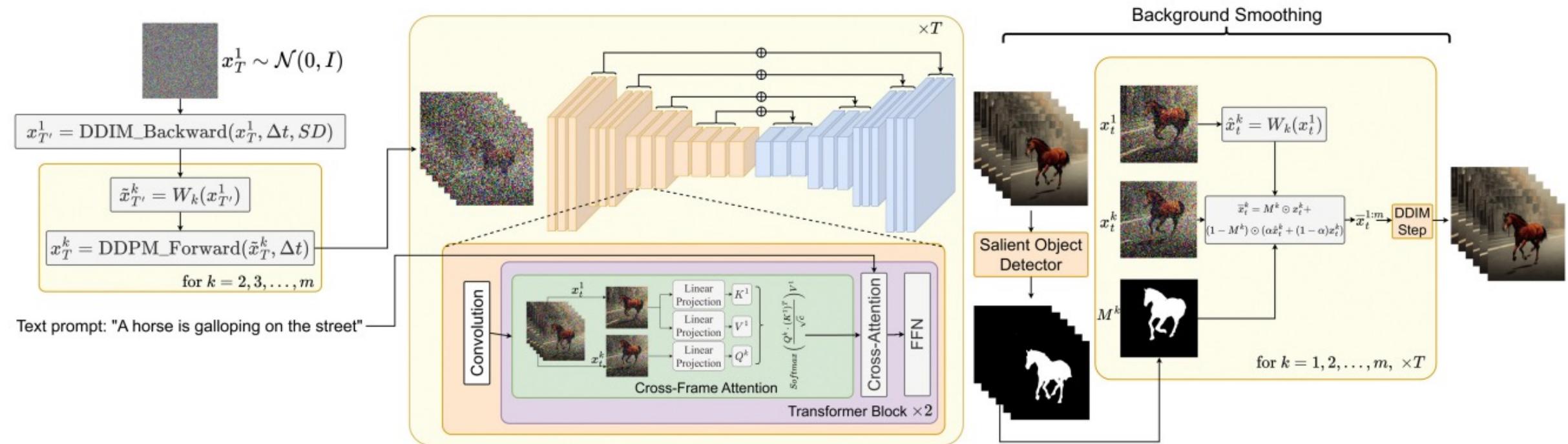


- Fine-tune the decoder to be video-aware, keeping encoder frozen
- Interleave spatial and temporal layers.
- The spatial layers are frozen, whereas temporal layers are trained.
- Temporal layers can be Conv3D or Temporal attentions.
- Context can be added for autoregressive generation.

Text2Video-Zero: Text-to-Image Diffusion Models are Zero-Shot Video Generators

A pretrained text-to-image diffusion model **without any further fine-tuning or optimization**

1. Encode motion dynamics in the latent codes
2. Reprogram each frame's self-attention using a new cross-frame attention



Conditional and Specialized Text-to-Video

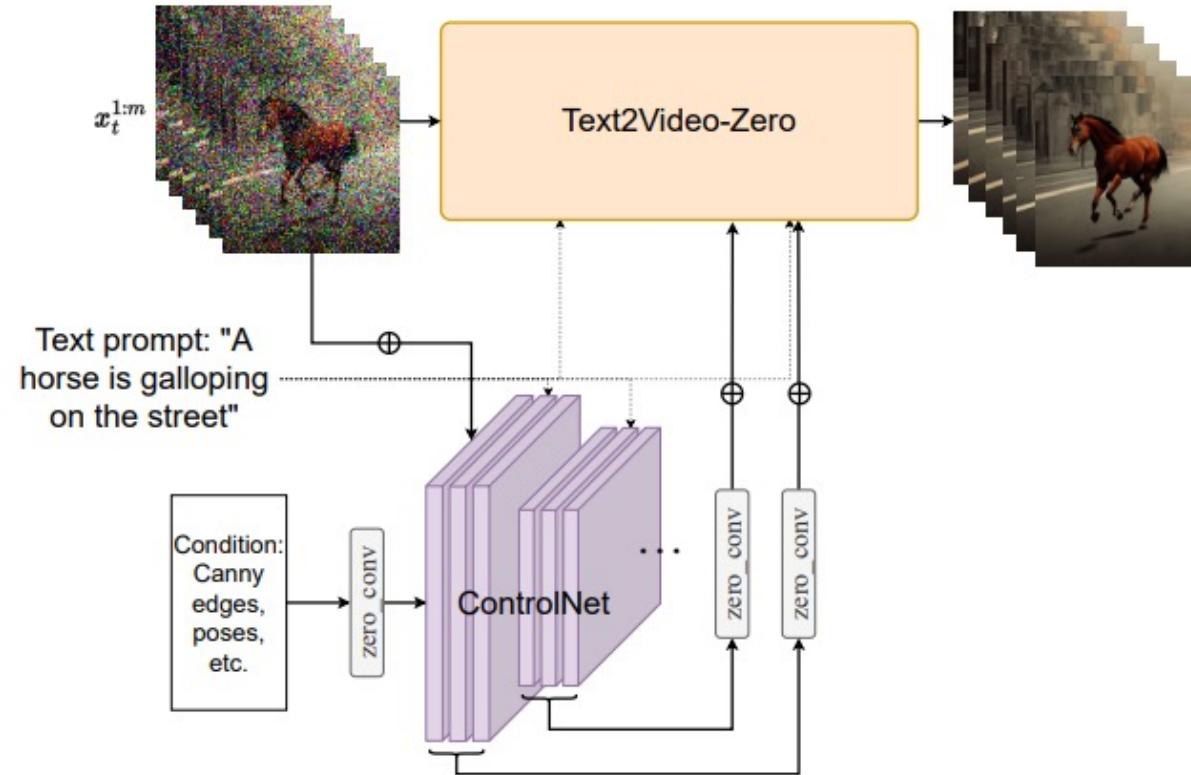
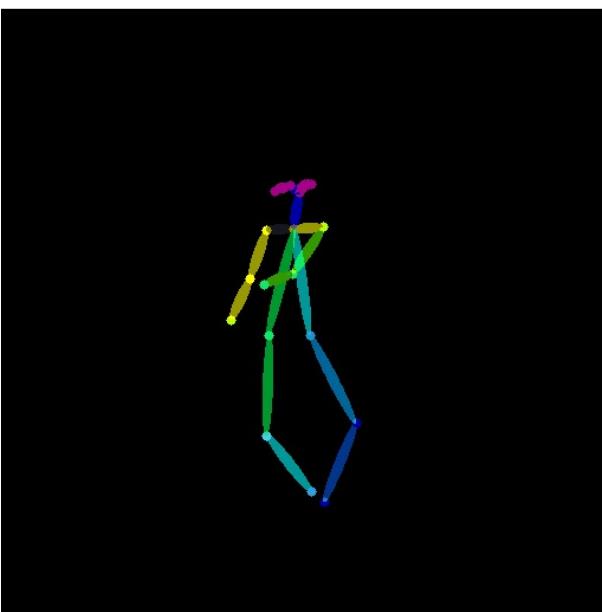
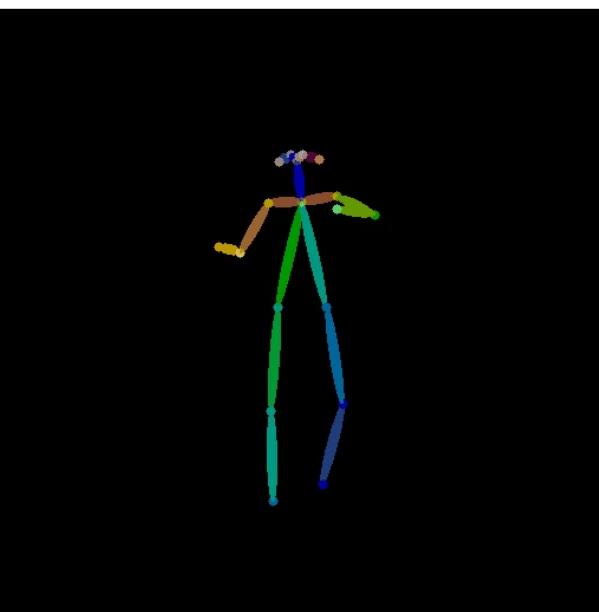
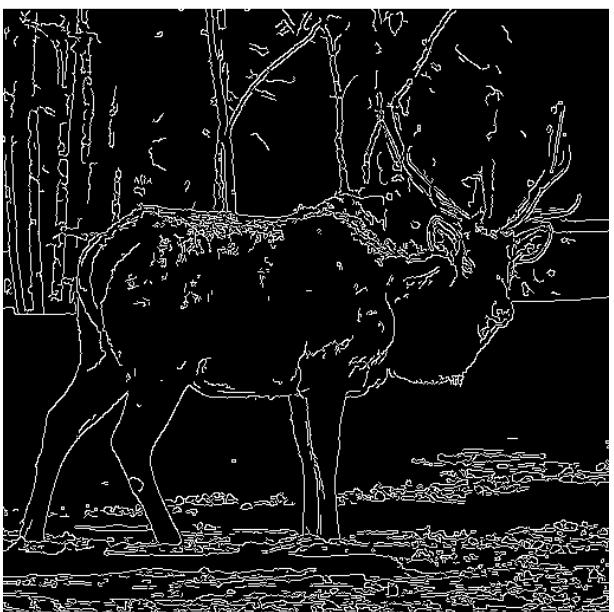
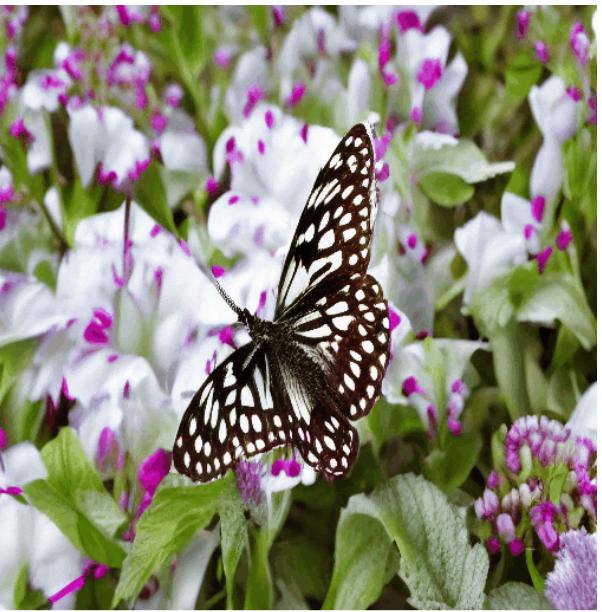


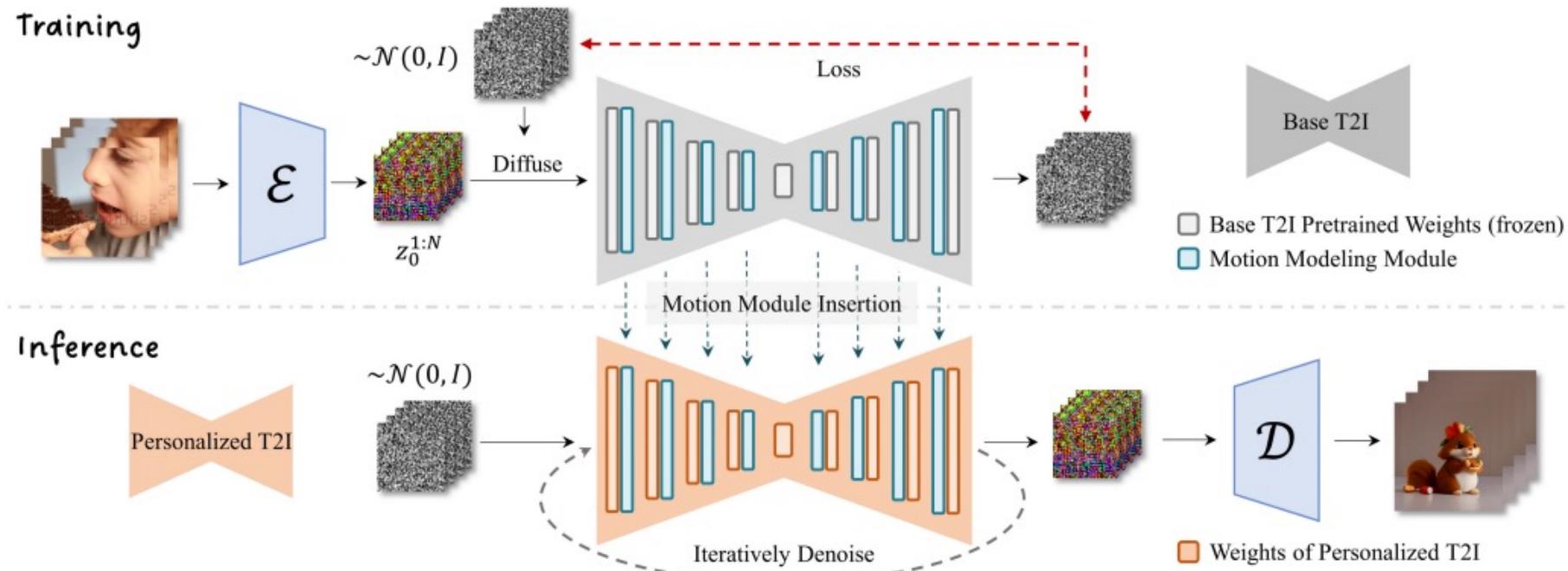
Figure 4: The overview of Text2Video-Zero + ControlNet





AnimateDiff: Animate Your Personalized Text-to-Image Diffusion Models without Specific Tuning

Base T2I contributes to the appearance and train an additional module for the motion



Training Pipeline of AnimateDiff

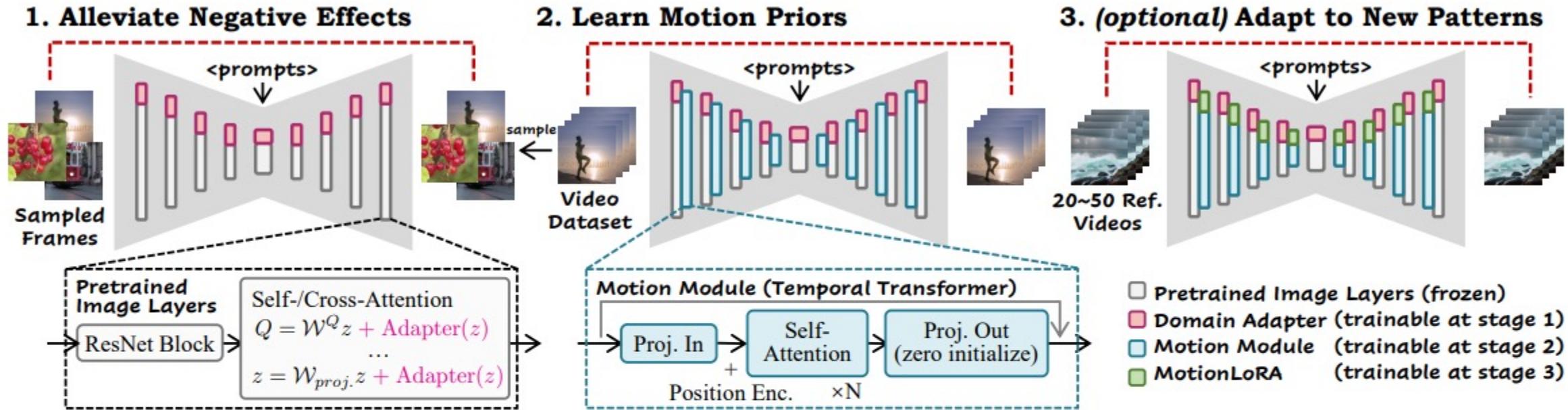
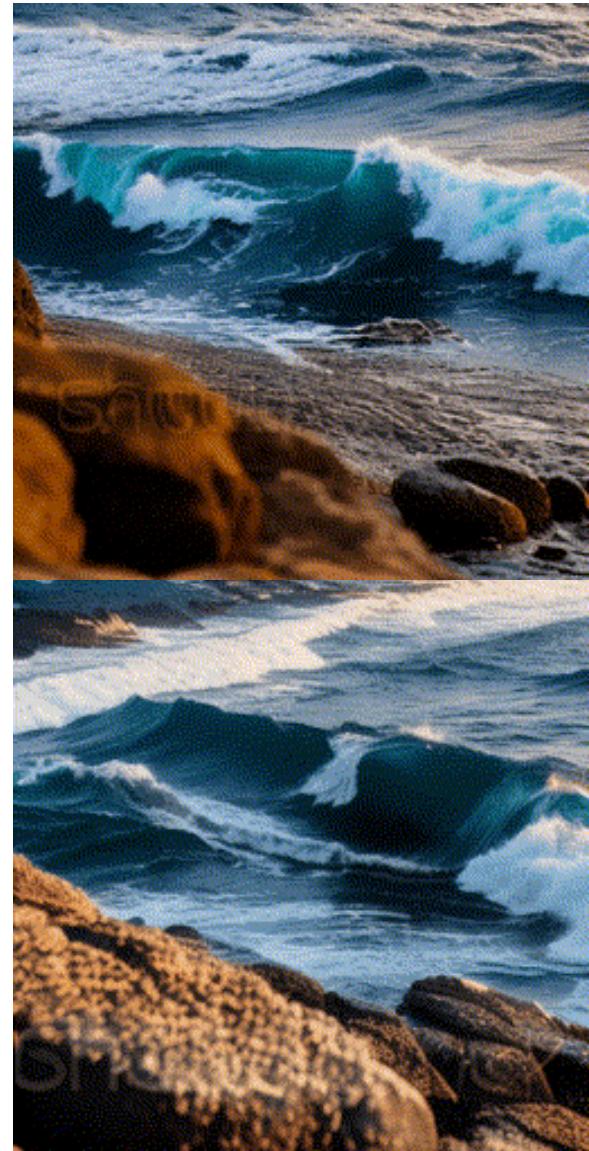
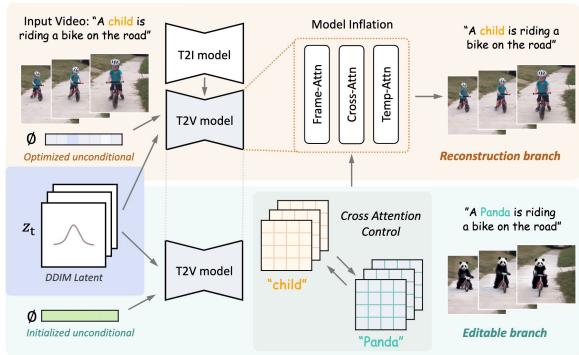


Figure 3: **Training pipeline of AnimateDiff.** AnimateDiff consists of three training stages for the corresponding component modules. Firstly, a domain adapter (Sec. 4.1) is trained to alleviate the negative effects caused by training videos. Secondly, a motion module (Sec. 4.2) is inserted and trained on videos to learn general motion priors. Lastly, MotionLoRA (Sec. 4.3) is trained on a few reference videos to adapt the pre-trained motion module to new motion patterns.

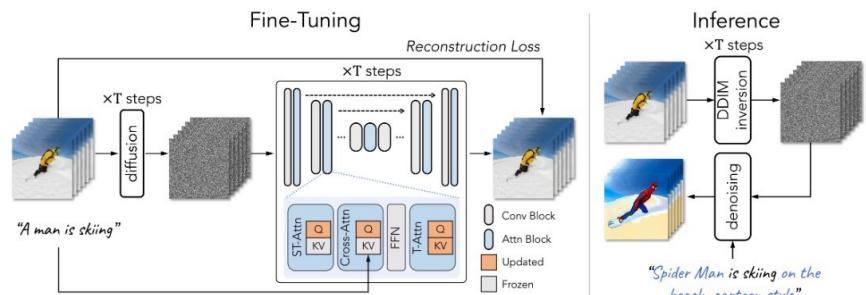
Fine-grained Control of Camera



More Concurrent / Related Works

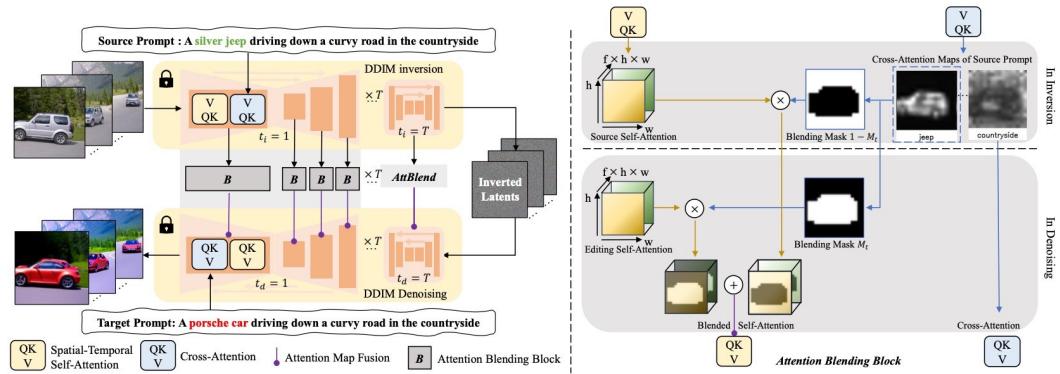


Video-P2P: Cross-Attention Control on text-to-video model

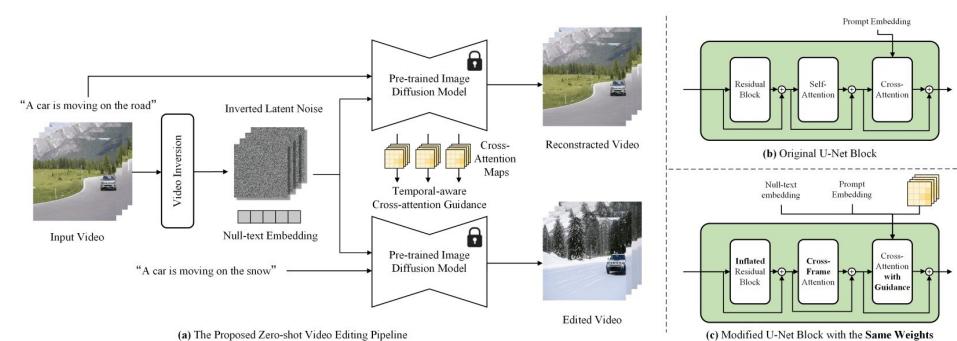


Given a text-video pair (e.g., "a man is skiing") as input, our method leverages the pretrained T2I diffusion models for T2V generation. During fine-tuning, we update the projection matrices in attention blocks using the standard diffusion training loss. During inference, we sample a novel video from the latent noise inverted from the input video, guided by an edited prompt (e.g., "Spider Man is surfing on the beach, cartoon style").

Tune-A-Video: Fine-tune projection matrices of the attention layers, from text2image model to text2video model.



FateZero: Store attention maps from DDIM inversion for later use



Vid2vid-zero: Learn a null-text embedding for inversion, then use cross-frame attention with original weights.

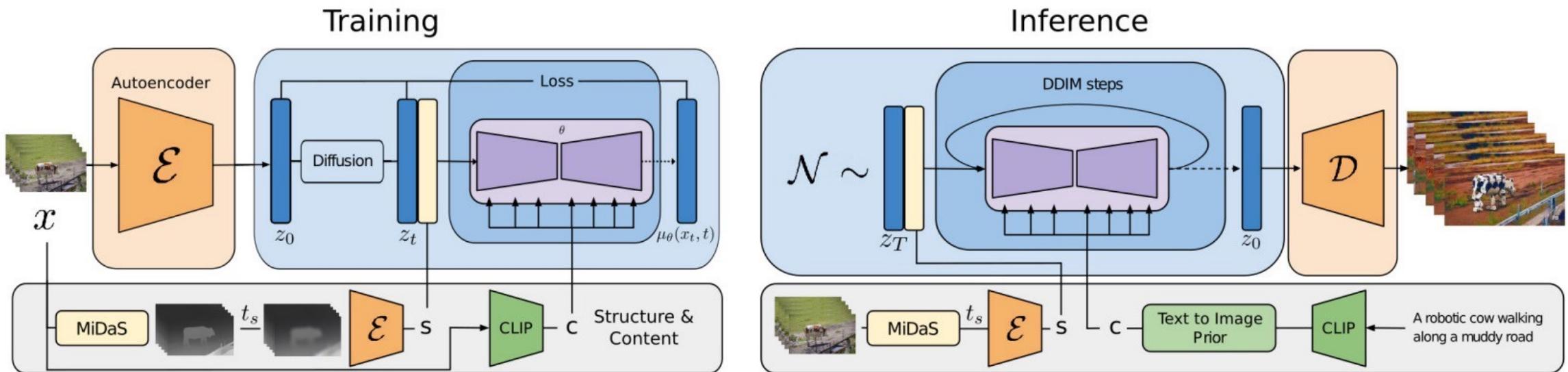
Gen-1 (video-to-video)

- Transfer the style of a video using text prompts given a “driving video”



Gen-1 (video-to-video)

- Condition on structure (depth) and content (CLIP) information.
- Depth maps are passed with latents as input conditions.
- CLIP image embeddings are provided via cross-attention blocks.
- During inference, CLIP text embeddings are converted to CLIP image embeddings.



Gen-2 (text-to-video, and more)



Gen-2 (the latest release, Nov 02 2023)





Gen AI in Film Making?

- From simple animation and quality enhancement, to nowadays Scriptwriting, Visual Effects (CGI), Subtitling, Scheduling, Trailers...
- **Next billion-dollar question:** end-to-end automated?



Earth Stood Still
1951
First cinematic
AI to become a
household name
(Gort)



The Terminator
1984



Due to the
popularity of the
franchise, people
still refer to any
digital totalitarian
force as "Skynet"



The Matrix
1999

A quintessential
Hollywood vision
of what an AI
takeover would
look like



WALL-E
2008



Figure source: enlightened-digital.com

Short Realistic Videos with Generative AI

- Rapidly generate assets
- Tell coherent stories and even evoke emotions
- Even surprise us with its creativity!



Sci-Fi Trailer Made with Generative AI



“AI Star Wars Teaser”,
made by the community
with current available
Generative AI techniques

Challenges Along the Road

- Generating highly accurate objects especially human faces and bodies, and their nuanced motions
- Creating complex, physically grounded motions
- Maintaining a cohesive narrative over the long time range

ARE WE
HERE
YET?





The Human Touch





The Ethical Concerns

01
0
1
10110100011011100100110001110101
0
0
0
11
01
01
01
11
11
00
11
11
00
10
11
11
00
10
1010
00110010
001110111001011
000110010100110

--- **CLOSER THAN WE THINK!**

by Rodebaugh

ROBOT DRIVING

Plans have already been perfected for control devices that will make intercity driving completely automatic.

One system, devised by RCA's famous Vladimir Zworykin, has been tested on a Nebraska public highway. Buried cables and loops of wire radiate signals that guide a specially equipped car and dictate its speed. General Motors has demonstrated robot driving with this system.

Another type of device would employ a continuous strip in the center of a thruway traffic lane. You'd drive over it, then push a button to start a magnetic follower electrode to follow the strip. The car would then guide itself and sensing devices would prevent collisions fore and aft.

The Future of GenAI Film Making

Rejuvenated Downtowns

© 1969 by
The Chicago Tribune. 2-22-10



The University of Texas at Austin
**Electrical and Computer
Engineering**
Cockrell School of Engineering