Diffusion Models for Visual Computing

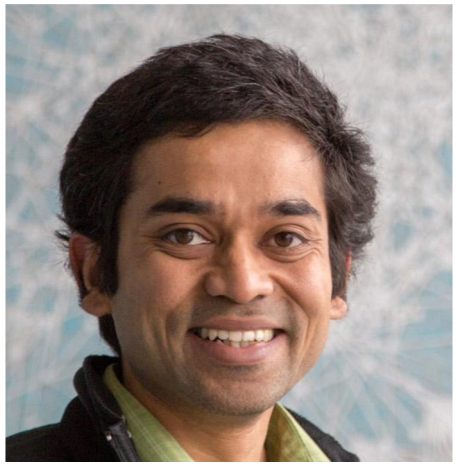
Niloy Mitra, Daniel Cohen-Or, Minhyuk Sung,

Chun-Hao Huang, Duygu Ceylan, Paul Guerrero

Part 1: Introduction to Diffusion Models

https://geometry.cs.ucl.ac.uk/courses/diffusion4VC_eg24/

People



Niloy Mitra

Duygu Ceylan

Daniel Cohen-Or

ChunHao Huang

Paul Guerrero

Minhyuk Sung

Why do we need this Tutorial?

What are diffusion model?

What are the design choices?

Controls and adaptation in the context of Visual Computing

Learn together

Related Materials

- Survey papers
- Past tutorials/courses
- Blogs and recorded videos

Presentation Schedule

Introduction to Diffusion Models

Guidance and Conditioning Sampling

Attention

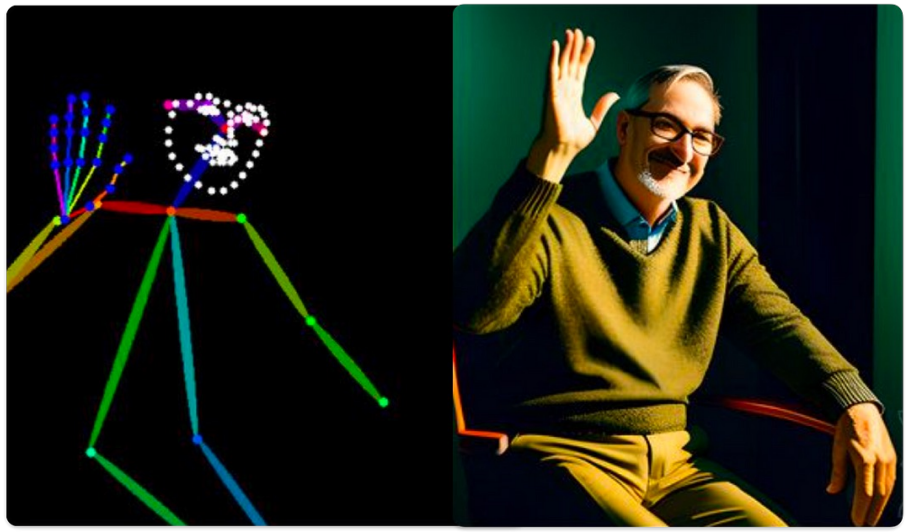
Break

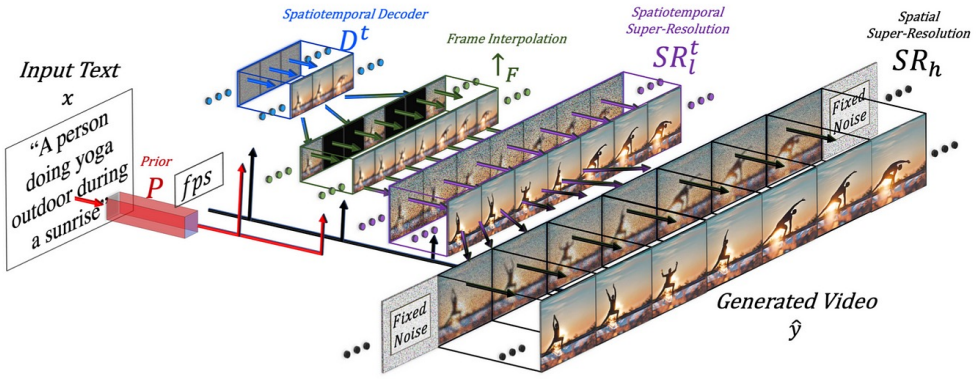
Personalization and Editing

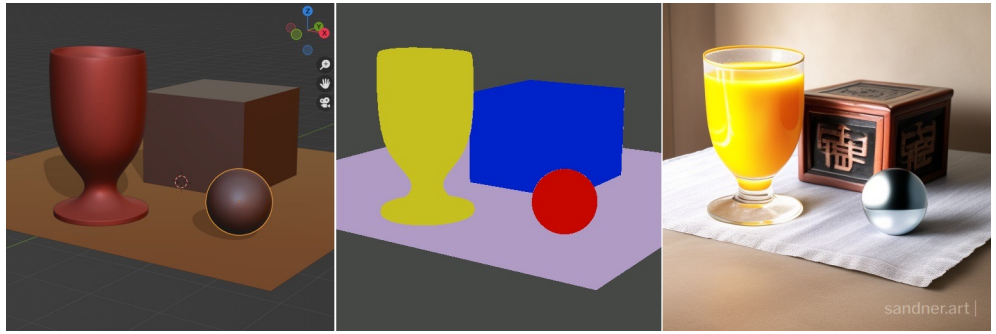
Beyond Single (RGB) Image Generation

Diffusion Models for 3D Generation

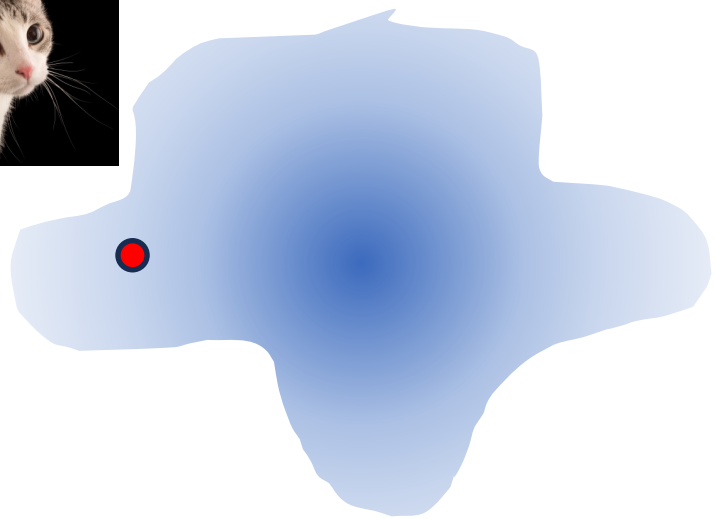
Applications



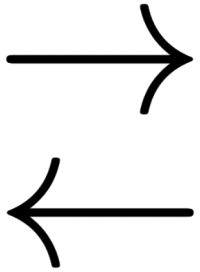




What is a Diffusion Process?



unknown map



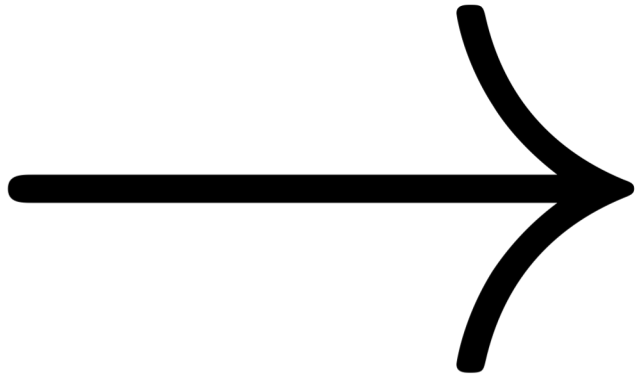
(unknown) data distribution

known distribution

Sampling $\Leftrightarrow$ *(Unconditional generation)*

Mapping between Distributions

$\mathbf{x}_0$



$\mathbf{x}_T$

$\mathcal{N}(\mathbf{0}, \mathbb{I})$

data distribution

known distribution

Gaussian (Normal) Distribution

- Uniquely defined by Mean and Variance

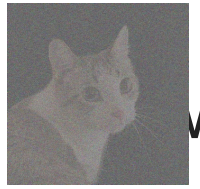
$$\mu, \Sigma$$

- Reparameterization ‘trick’

$$\mathcal{N}(\mu, \Sigma) \qquad\qquad x \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

- Many results on combining Gaussian distributions

Mapping in Many Steps



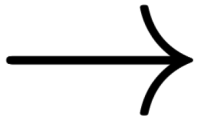
ward mapping

$$\mathbf{x}_0 \rightarrow \qquad \mathbf{x}_{t-1} \rightarrow \mathbf{x}_t \qquad \rightarrow \qquad \rightarrow \mathbf{x}_T$$

$$\mathcal{N}(\mathbf{0}, \mathbb{I})$$

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathbb{I})$$

Mapping in Many Steps



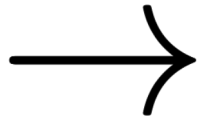
forward mapping

$$\mathbf{x}_0 \rightarrow \quad \mathbf{x}_{t-1} \rightarrow \mathbf{x}_t \quad \rightarrow \quad \rightarrow \mathbf{x}_T$$

$$\mathcal{N}(\mathbf{0}, \mathbb{I})$$

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathbb{I})$$

Mapping in Many Steps



forward mapping

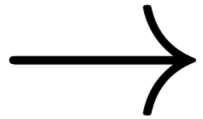
$$\mathbf{x}_0 \rightarrow \qquad \mathbf{x}_{t-1} \rightarrow \mathbf{x}_t \qquad \rightarrow \qquad \rightarrow \mathbf{x}_T$$

$$\mathcal{N}(\mathbf{0}, \mathbb{I})$$

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathbb{I})$$

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\overline{\alpha}_t}x_0, (1-\overline{\alpha}_t)\mathbb{I}) \qquad \overline{\alpha}_t = \Pi_{i=1}^{t}\alpha_i$$

Mapping in Many Steps



forward mapping

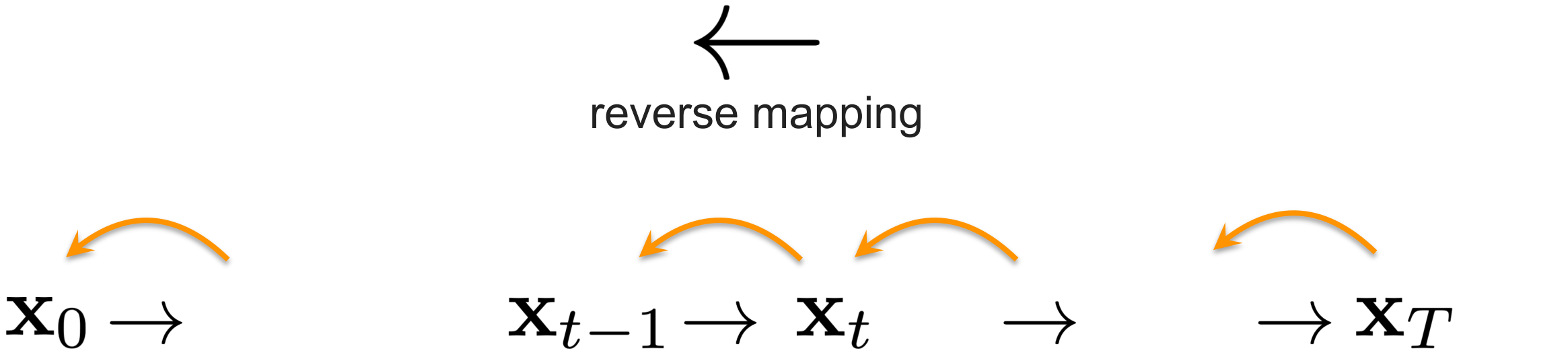
$$\mathbf{x}_0 \rightarrow \quad \mathbf{x}_{t-1} \rightarrow \mathbf{x}_t \quad \rightarrow \quad \rightarrow \mathbf{x}_T$$

$$\mathcal{N}(\mathbf{0}, \mathbb{I})$$

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbb{I})$$

$$x_t = \hat{\epsilon}(x_0) = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{(1-\bar{\alpha}_t)}\epsilon_t \qquad \bar{\alpha}_t = \Pi_{i=1}^{t}\alpha_i$$

Generative Modeling: Sampling



$$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{(1-\overline{\alpha}_t)}\epsilon_t$$

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \sqrt{\overline{\alpha}_{t-1}} D_\theta(x_t, t), (1-\overline{\alpha}_{t-1})\mathbb{I})$$

Loss Functions

$$\mathcal{L}_{simple}(\theta) = \mathbb{E}_{t,x_0,\epsilon}[C_t \| \epsilon_\theta(x_t, t) - \epsilon \|^2]$$

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\epsilon,x_0}\left[C_t \| D_\theta(\hat{\epsilon}_t(x_0), t) - x_0 \|^2\right]$$

$$p(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \sqrt{\overline{\alpha}_{t-1}} D_\theta(x_t, t), (1 - \overline{\alpha}_{t-1})\mathbb{I})$$

Algorithm (How to Train?)

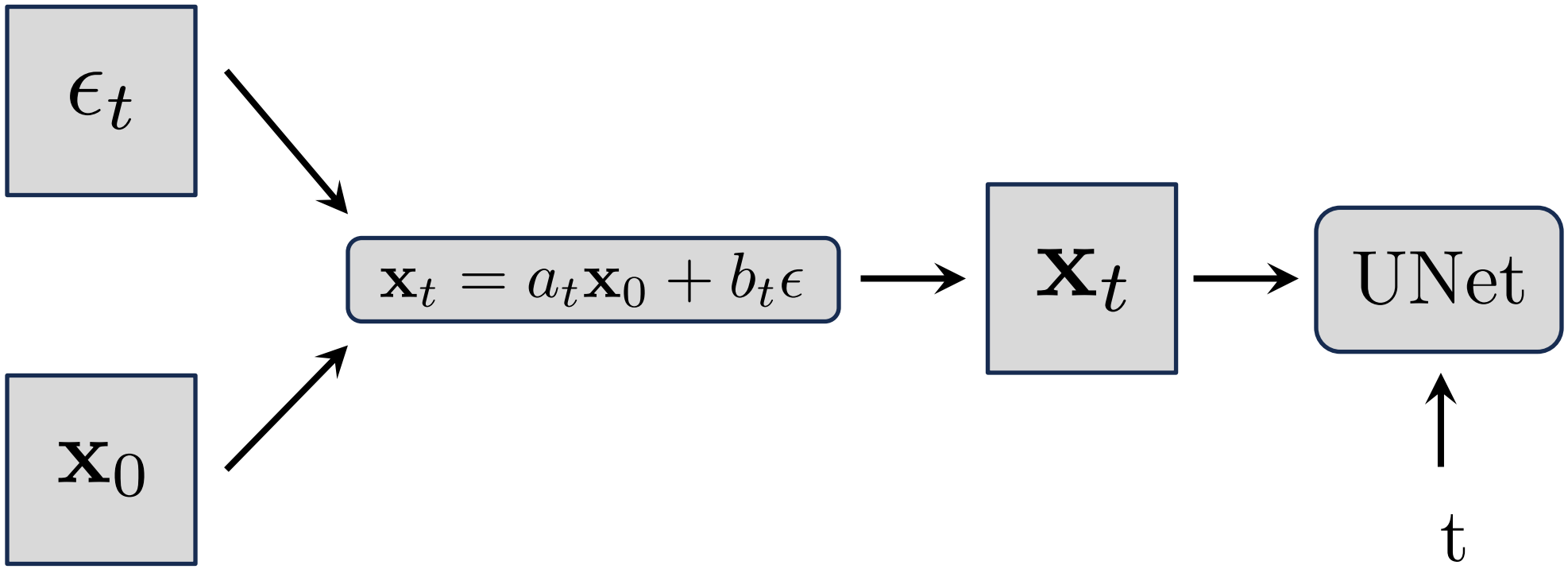
Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on

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

6: **until** converged

Training Loss



Three Interpretations

- Predict Noise ϵ_t
- Predict clean image $\mathbf{x}_0$
- Score-based optimization $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_0) = -\frac{\epsilon}{\sqrt{1-\bar{\alpha}_t}}$

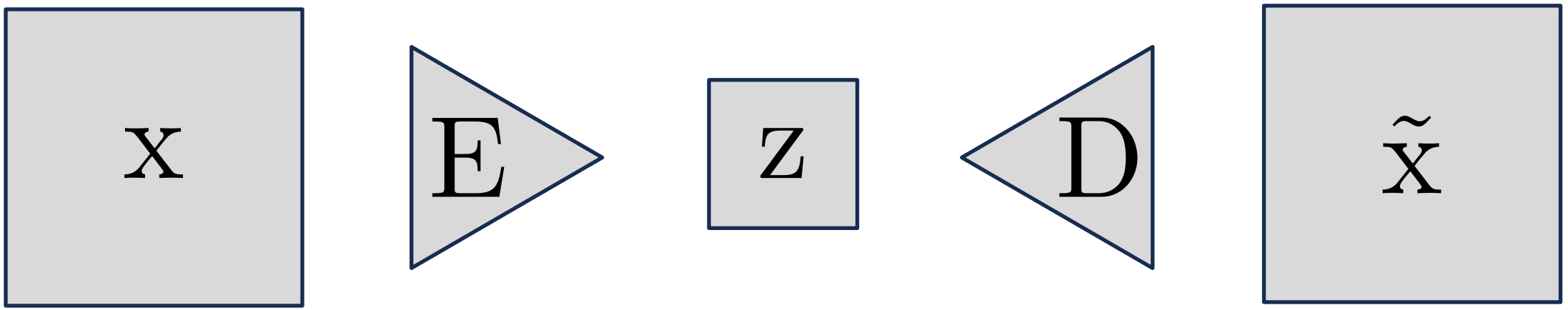
they are equivalent!!

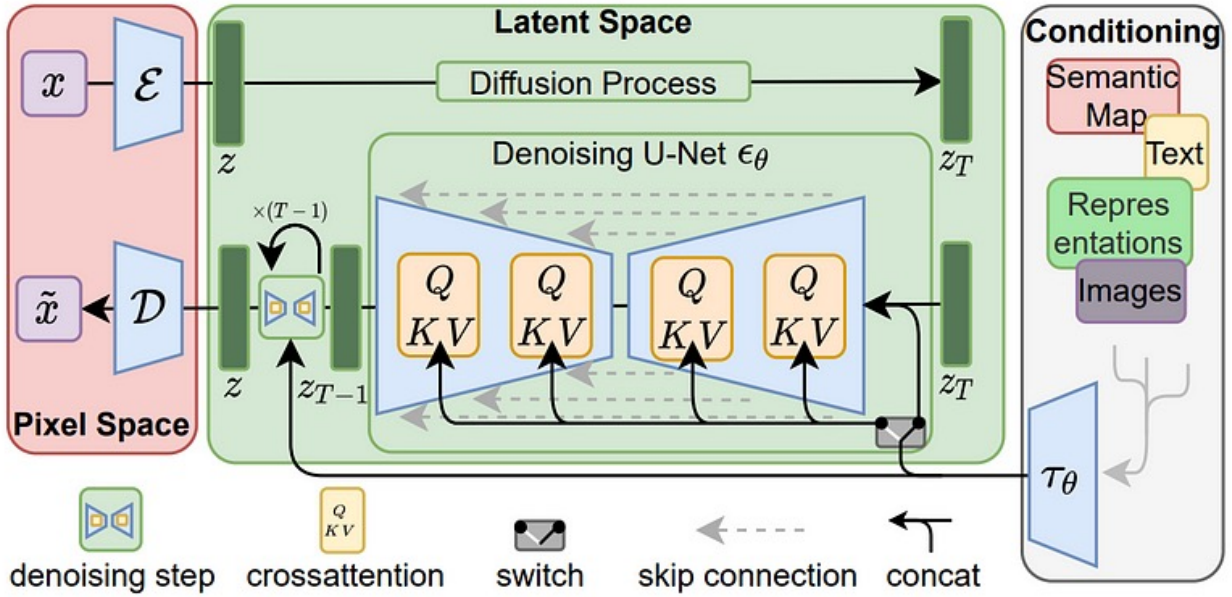
Algorithm (How to Sample?)

Algorithm 2 Sampling

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

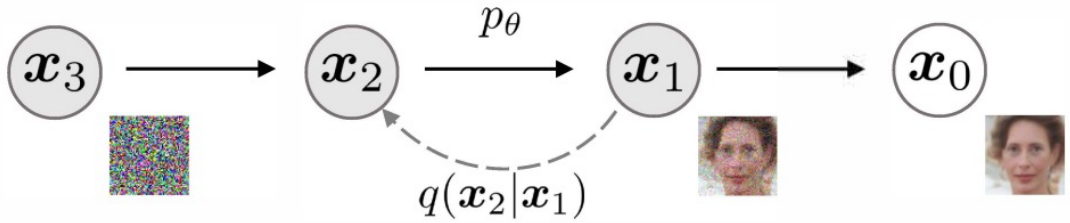
Latent Diffusion Model



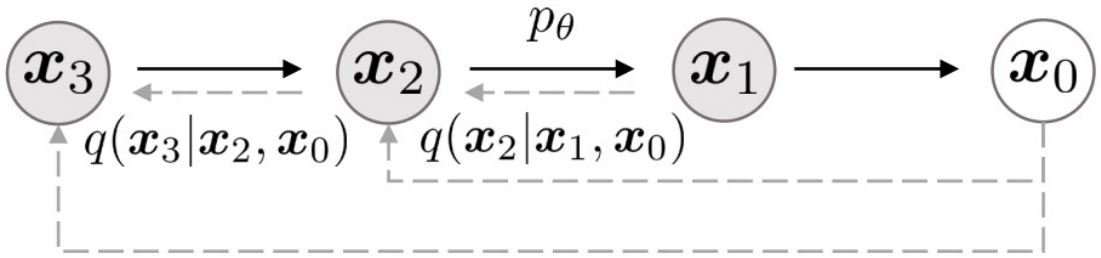


DDPM vs DDIM

- DDPM: Markovian process



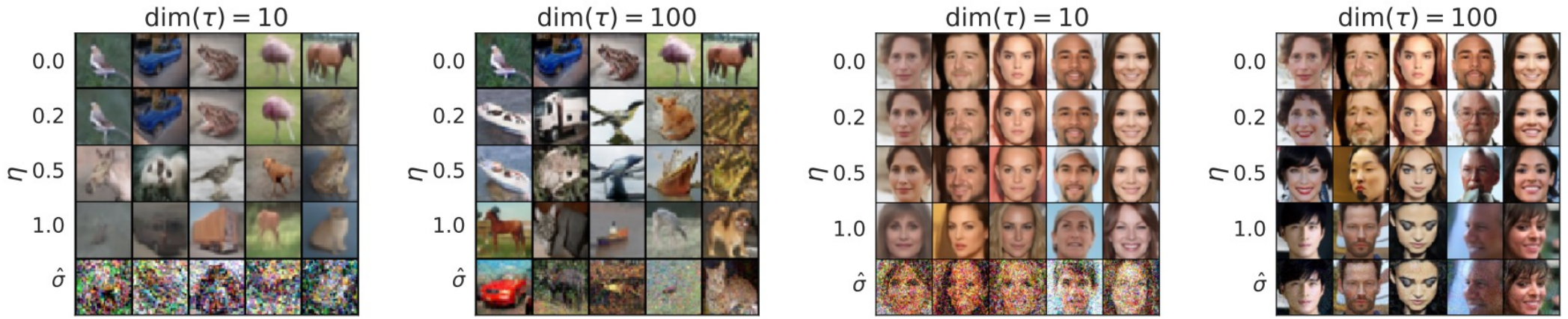
- DDIM: Non-Markovian process but 10-50x faster!!
  - Trained w/ pretrained DDPM diffusion



$$x_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{x_t - \sqrt{1-\alpha_t}\epsilon_\theta^{(t)}(x_t)}{\sqrt{\alpha_t}} \right)}_{\text{“ predicted } x_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma_t^2} \cdot \epsilon_\theta^{(t)}(x_t)}_{\text{“direction pointing to } x_t\text{”}} + \underbrace{\sigma_t \epsilon_t}_{\text{random noise}}$$

DDPM vs DDIM

	S	CIFAR10 (32 × 32) 10	20	50	100	1000	CelebA (64 × 64) 10	20	50	100	1000
η	0.0	**13.36**	**6.84**	**4.67**	**4.16**	4.04	**17.33**	**13.73**	**9.17**	**6.53**	3.51
	0.2	14.04	7.11	4.77	4.25	4.09	17.66	14.11	9.51	6.79	3.64
	0.5	16.66	8.35	5.25	4.46	4.29	19.86	16.06	11.01	8.09	4.28
	1.0	41.07	18.36	8.01	5.78	4.73	33.12	26.03	18.48	13.93	5.98
$\hat{\sigma}$		367.43	133.37	32.72	9.99	**3.17**	299.71	183.83	71.71	45.20	**3.26**



Summary so far

Algorithm 1 Training

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

Algorithm 2 Sampling

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

Presentation Schedule

Introduction to Diffusion Models

Guidance and Conditioning Sampling

Attention

Break

Personalization and Editing

Beyond Single (RGB) Image Generation

Diffusion Models for 3D Generation

Diffusion Models for Visual Content Generation

Guidance

Presenter: Minhyuk Sung

Eurographics 2024 Tutorial

Guidance Using Additional Information

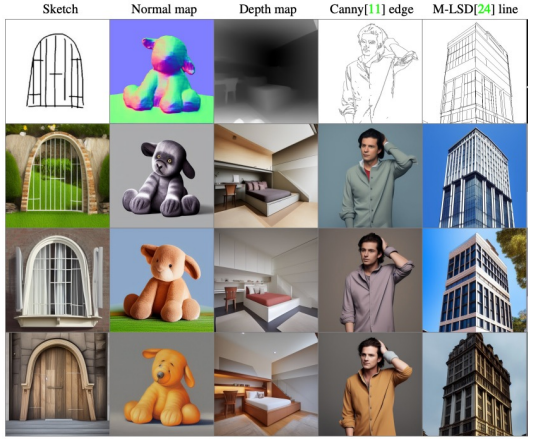
If we have additional information about the data, such as class labels for images, can we utilize this information to improve the quality of generated outputs?

Ho and Salimans, Classifier-Free Diffusion Guidance, NeurIPS 2021 Workshop.

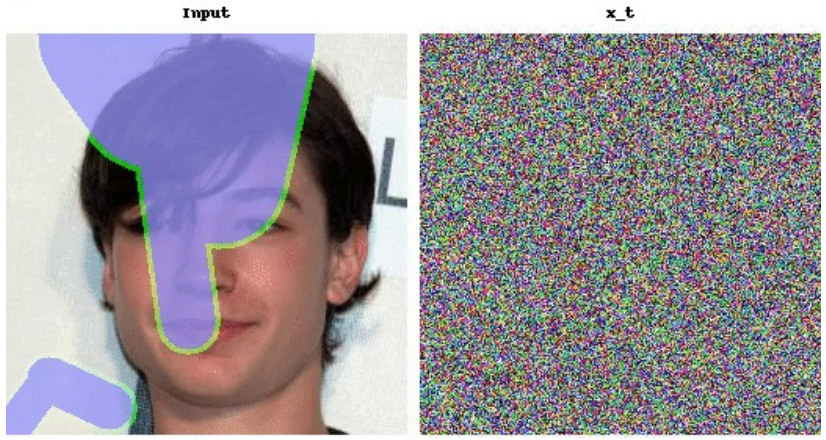
Zero-Shot / Few-Shot Adaptations

Can we apply a pretrained diffusion model to various conditional generation setups in a zero-shot or few-shot manner?

Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022.



Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.



Lugmayr et al., RePaint: Inpainting using Denoising Diffusion Probabilistic Models, CVPR 2022.

Content

1. Classifier Guidance / Classifier-Free Guidance
2. ControlNet
3. SDEdit / RePaint

Classifier Guidance / Classifier-Free Guidance (CFG)

Guidance Using Additional Information

How to use class labels or some additional information about the data to improve the quality of generated outputs?

Ho and Salimans, Classifier-Free Diffusion Guidance, NeurIPS 2021 Workshop.

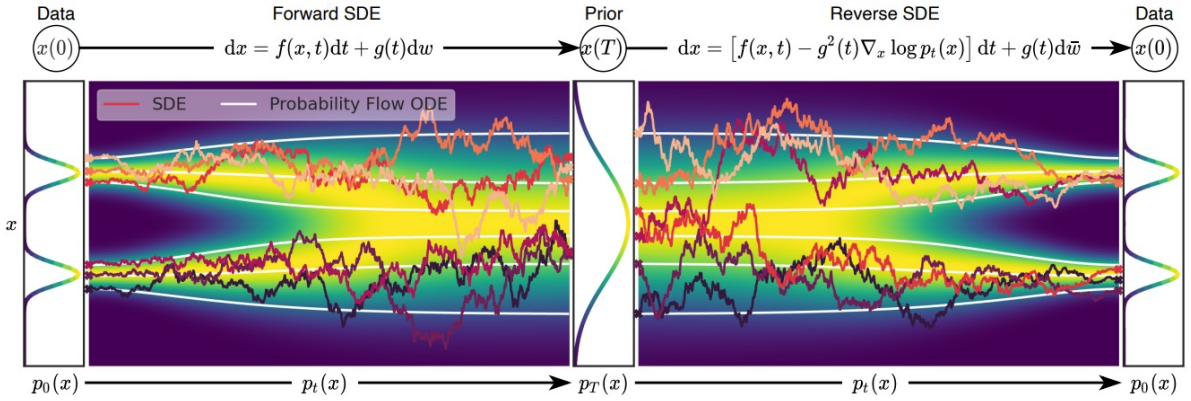
Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021.

Recap: Stochastic Differential Equations

In a continuous time domain, the reverse process is formulated as the following stochastic differential equation:

$$d\mathbf{x} = [\mathbf{f}(\mathbf{x}, t)dt - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]dt + g(t)d\mathbf{w}$$

DDPM is a specific discretization of the SDE formulation.



Recap: Stochastic Differential Equations

The gradient of the logarithm of the PDF $\nabla_{\mathbf{x}} \log q(\mathbf{x}_t)$ is referred to as a score function.

$$\nabla_{\mathbf{x}} \log q(\mathbf{x}_t) = \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)} \left[\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_0) \right]$$

$$= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \mathbb{E}_{\mathbf{x}_0 \sim q(\mathbf{x}_0)} \left[\boldsymbol{\varepsilon}(\mathbf{x}_t, \mathbf{x}_0) \right] = -\frac{\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}$$

The noise predicted by the nerual network $\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)$ is the scaled score!

Classifier Guidance

- Assume that data $\mathbf{x}$ and class label y pairs are given:

$$p(\mathbf{x}_t, y) = p(\mathbf{x}_t)p(y|\mathbf{x}_t)$$

- At each timestep t, we are interested in the score function $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t, y)$:

$$\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t, y) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t)$$

$$= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) + \nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t)$$

$$= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \left(\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) - \sqrt{1-\bar{\alpha}_t} \nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t) \right)$$

Noise predictor

How to compute this?

Classifier Guidance

How to compute $\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t)$?

- Train a classfier $p_\phi(y|\mathbf{x}_t)$, taking noisy data $\mathbf{x}_t$ (and timestep t) as input and classifying it.

- Use $\nabla_{\mathbf{x}_t} \log p_\phi(y|\mathbf{x}_t)$ as an approximation of $\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t)$.

Classifier Guidance

- Update the noise predictor $\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)$ as follows:

$$\bar{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t) = \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) - \sqrt{1-\bar{\alpha}_t}\nabla_{\mathbf{x}_t} \log p_\phi(y|\mathbf{x}_t)$$

- The strength of the classifier guidance can be controlled by adding a weight w:

$$\bar{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, t) = \boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t) - w\sqrt{1-\bar{\alpha}_t}\nabla_{\mathbf{x}_t} \log p_\phi(y|\mathbf{x}_t)$$

- **Limitation**: Additional training of a classifier is required.

Classifier-Free Guidance (CFG)

- Jointly train both the conditional and unconditional diffusion models.
- For the unconditional case, simply feed a null token $\emptyset$ as the condition.
- In the reverse process, given the condition y, take $\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, y, t)$.

- Take a linear combination of unconditional and conditional noises as follows:

$$\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, y, t) = (1+w)\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, y, t) - w\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, \emptyset, t) \qquad w \geq 0$$

Increase ↑ the conditional noise

Decrease ↓ the unconditional noise

Classifier-Free Guidance (CFG)

Model	FID (↓)
BigGAN-deep, max IS (Brock et al., 2019)	25
BigGAN-deep (Brock et al., 2019)	5.7
CDM (Ho et al., 2021)	3.52
LOGAN (Wu et al., 2019)	3.36
Classifier Guidance ADM-G (Dhariwal & Nichol, 2021)	2.97
Ours	$T = 128$ / 256 / 1024
$w = 0.0$	8.11 / 7.27 / 7.22
$w = 0.1$	5.31 / 4.53 / 4.5
$w = 0.2$	3.7 / 3.03 / 3
$w = 0.3$	3.04 / **2.43** / **2.43**
$w = 0.4$	3.02 / 2.49 / 2.48
$w = 0.5$	3.43 / 2.98 / 2.96
$w = 0.6$	4.09 / 3.76 / 3.73
$w = 0.7$	4.96 / 4.67 / 4.69
$w = 0.8$	5.93 / 5.74 / 5.71
$w = 0.9$	6.89 / 6.8 / 6.81
$w = 1.0$	7.88 / 7.86 / 7.8
$w = 2.0$	15.9 / 15.93 / 15.75
$w = 3.0$	19.77 / 19.77 / 19.56
$w = 4.0$	21.55 / 21.53 / 21.45

$T = 256$

ImageNet Results

Classifier-Free Guidance (CFG)

- $(+)$ The classifier does not need to be trained.
- $(+)$ It is more versatile and can be used not only for class labels but also for any additional information (e.g., text descriptions).
- $(-)$ In the generation process, the noise predictor needs to be evaluated twice.
- $(-)$ Determining the optimal weight w can be challenging.

Example: Audio Conditioned Generation

Example: Audio Conditioned Generation

Text Prompt: *"Aurora Borealis Lights"*

Waterfall Burbling

Text Prompt: *"Lego"*

Snow

Text Prompt: *"Surreal Dreamscapes"*

Forest

Lee et al., InterHandGen: Two-Hand Interaction Generation via Cascaded Reverse Diffusion, CVPR 2024.

Example: Two-Hand Interaction Generation

One hand is generated based on the condition of the other hand.

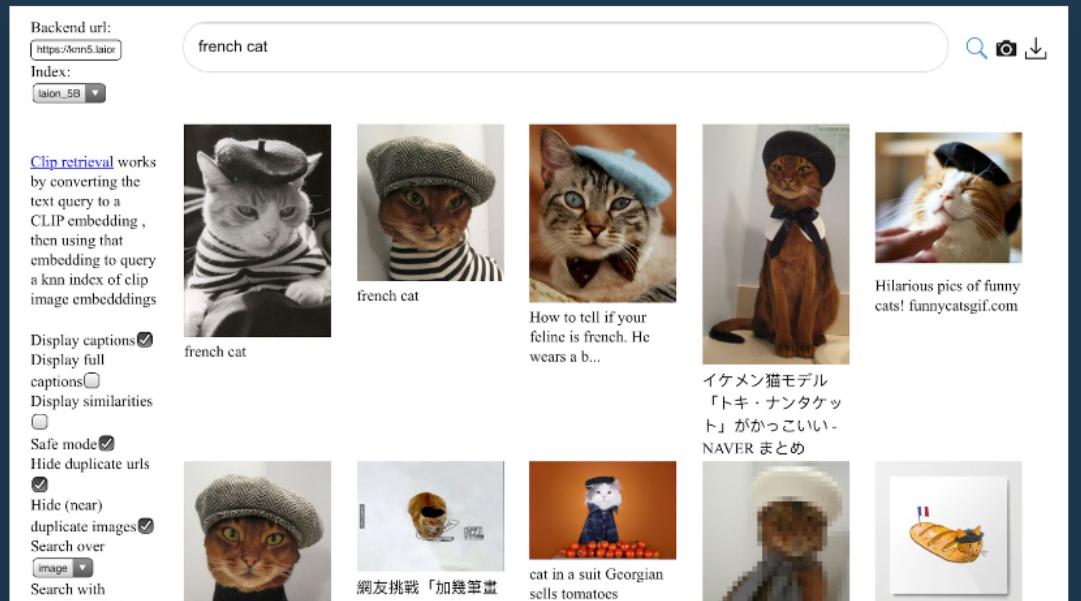
ControlNet:
Few-Shot Adaptation

LAION-5B: A NEW ERA OF OPEN LARGE-SCALE MULTI-MODAL DATASETS

by: Romain Beaumont, 31 Mar, 2022

We present a dataset o 5,85 billion CLIP-filtered image-text pairs, 14x bigger than LAION-400M, previously the biggest openly accessible image-text dataset in the world - see also our NeurIPS2022 paper

Authors: Christoph Schuhmann, Richard Vencu, Romain Beaumont, Theo Coombes, Cade Gordon, Aarush Katta, Robert Kaczmarczyk, Jenia Jitsev



Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.

ControlNet [Zhang et al., 2023]

- Should we have 5 billion input-output pairs to train conditional image diffusion models?
- Should we have the new dataset for every type of condition?

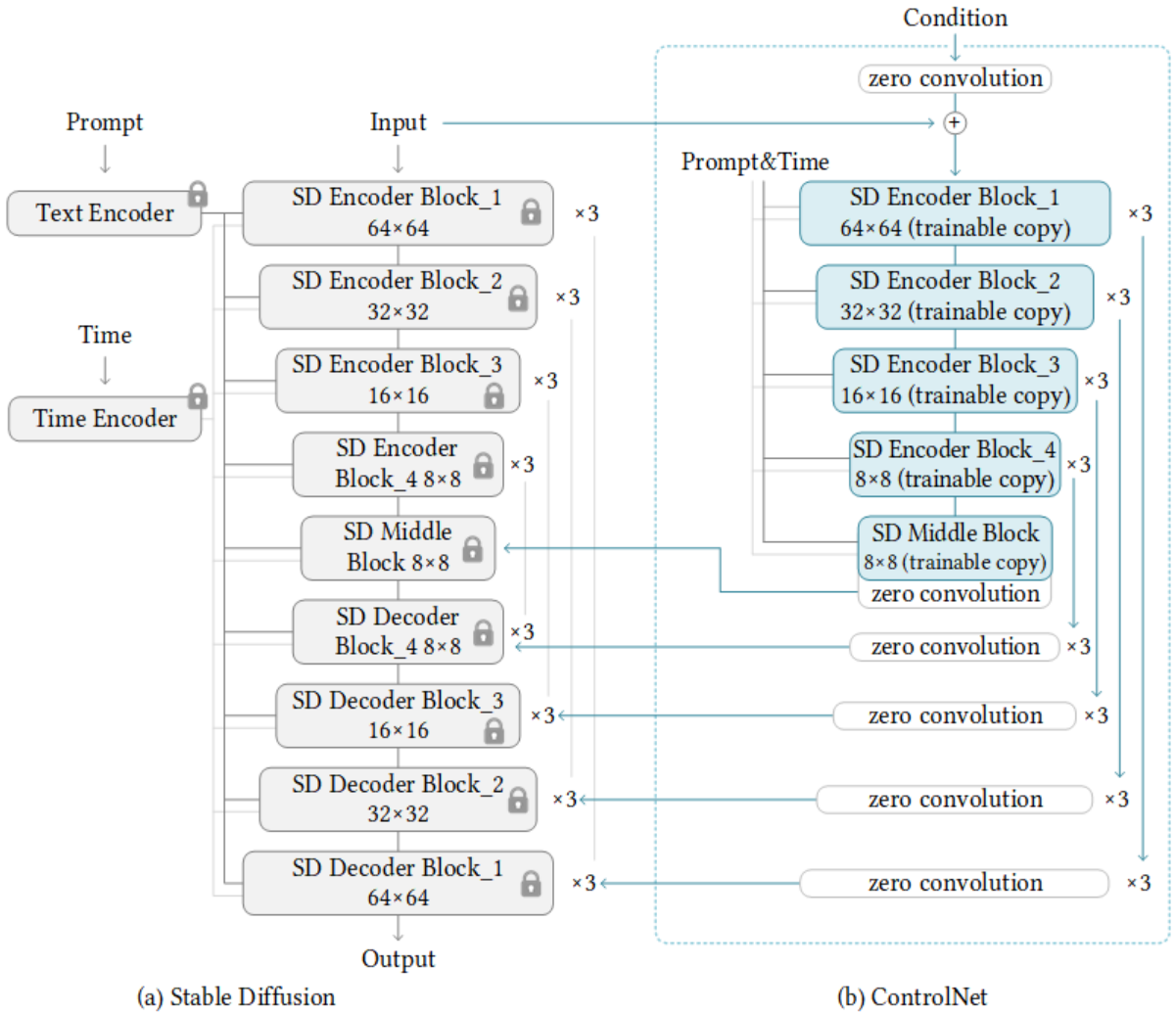
Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.

ControlNet [Zhang et al., 2023]

How to convert a pretrained unconditional image diffusion model into an image-conditioned generative model using a relatively smaller set of input-output pairs ($\sim 100k$)?

Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.

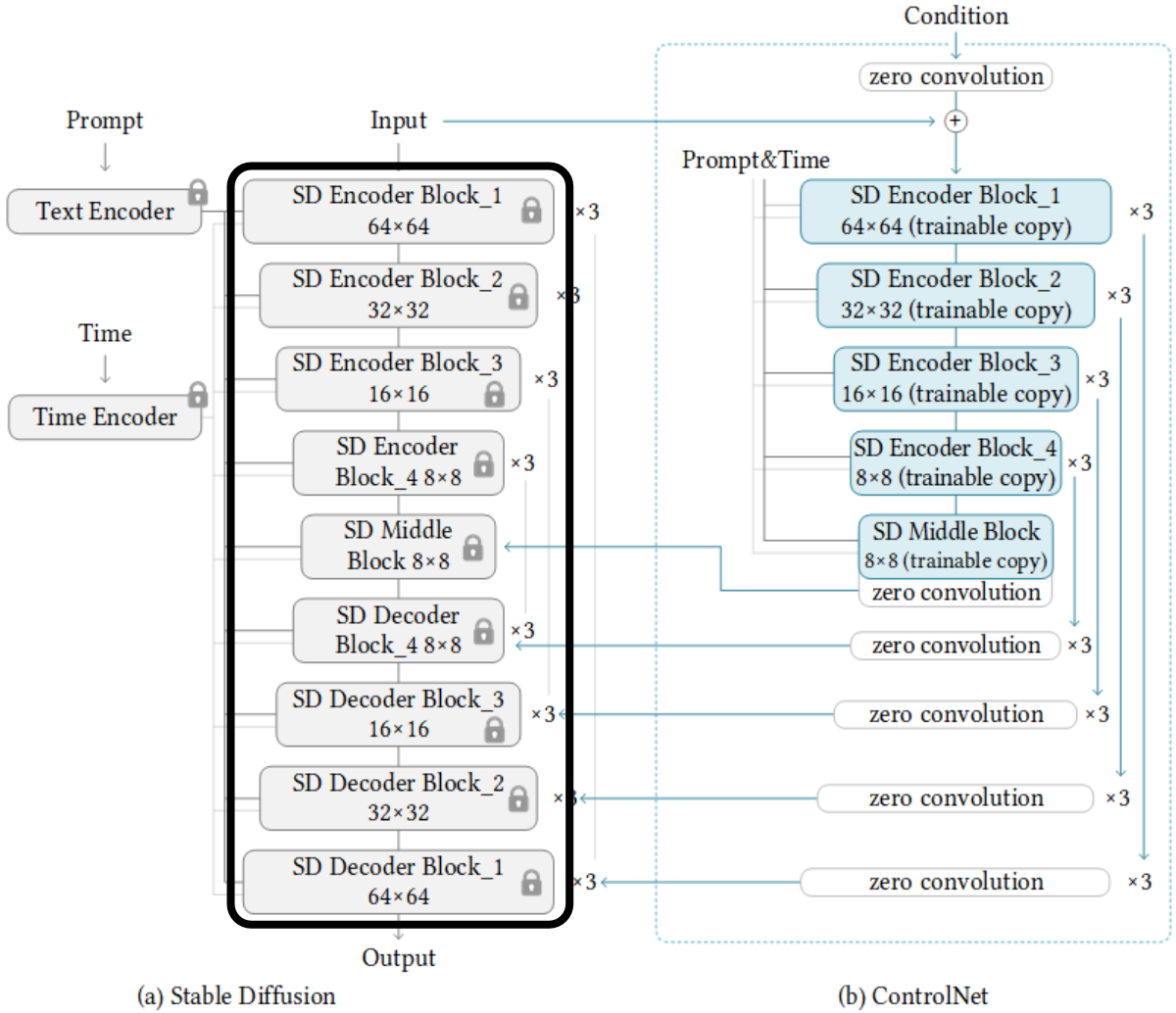
ControlNet [Zhang et al., 2023]



Key Idea:

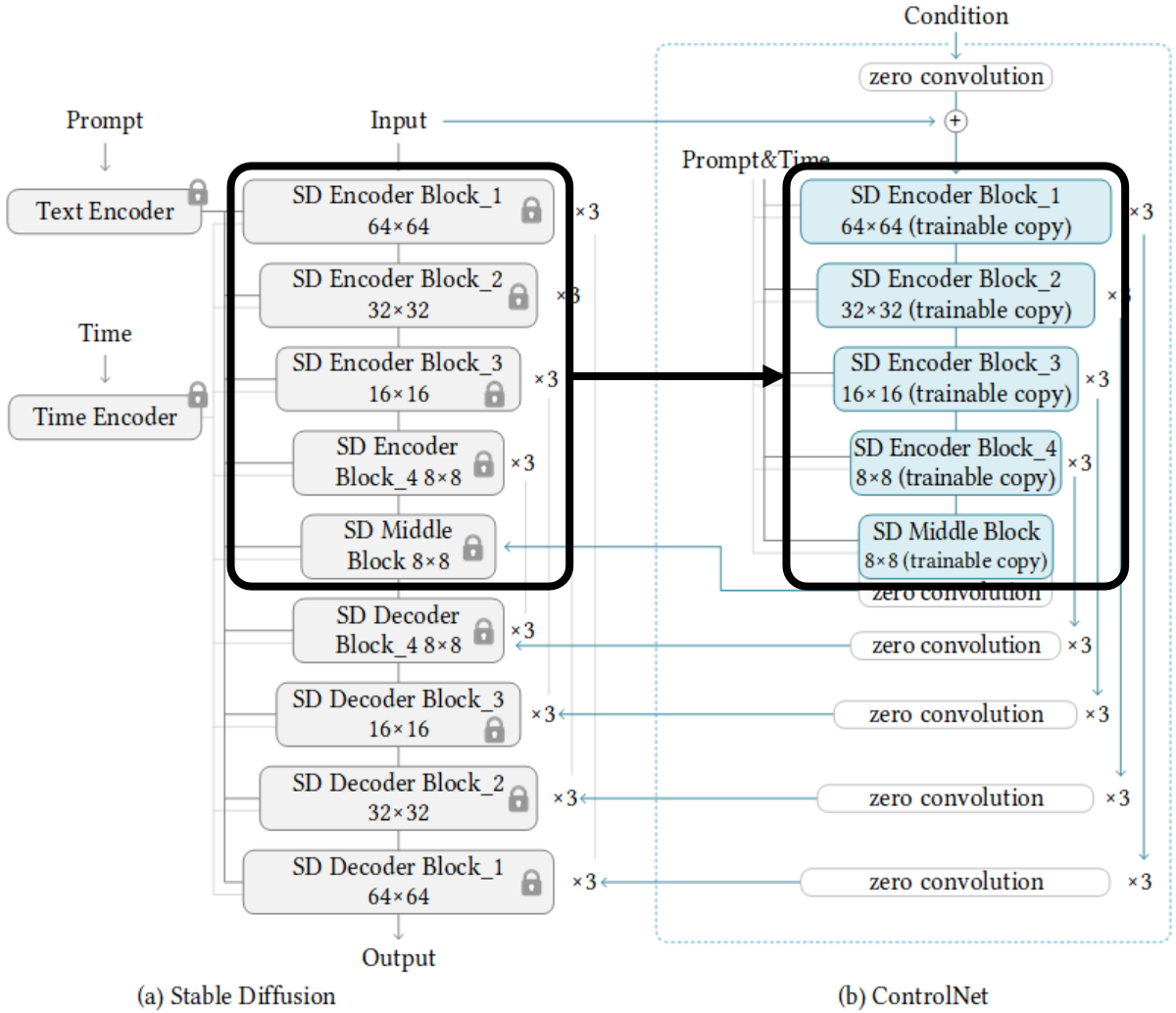
Fully leverage the pretrained noise prediction network to process the conditional image.

ControlNet [Zhang et al., 2023]



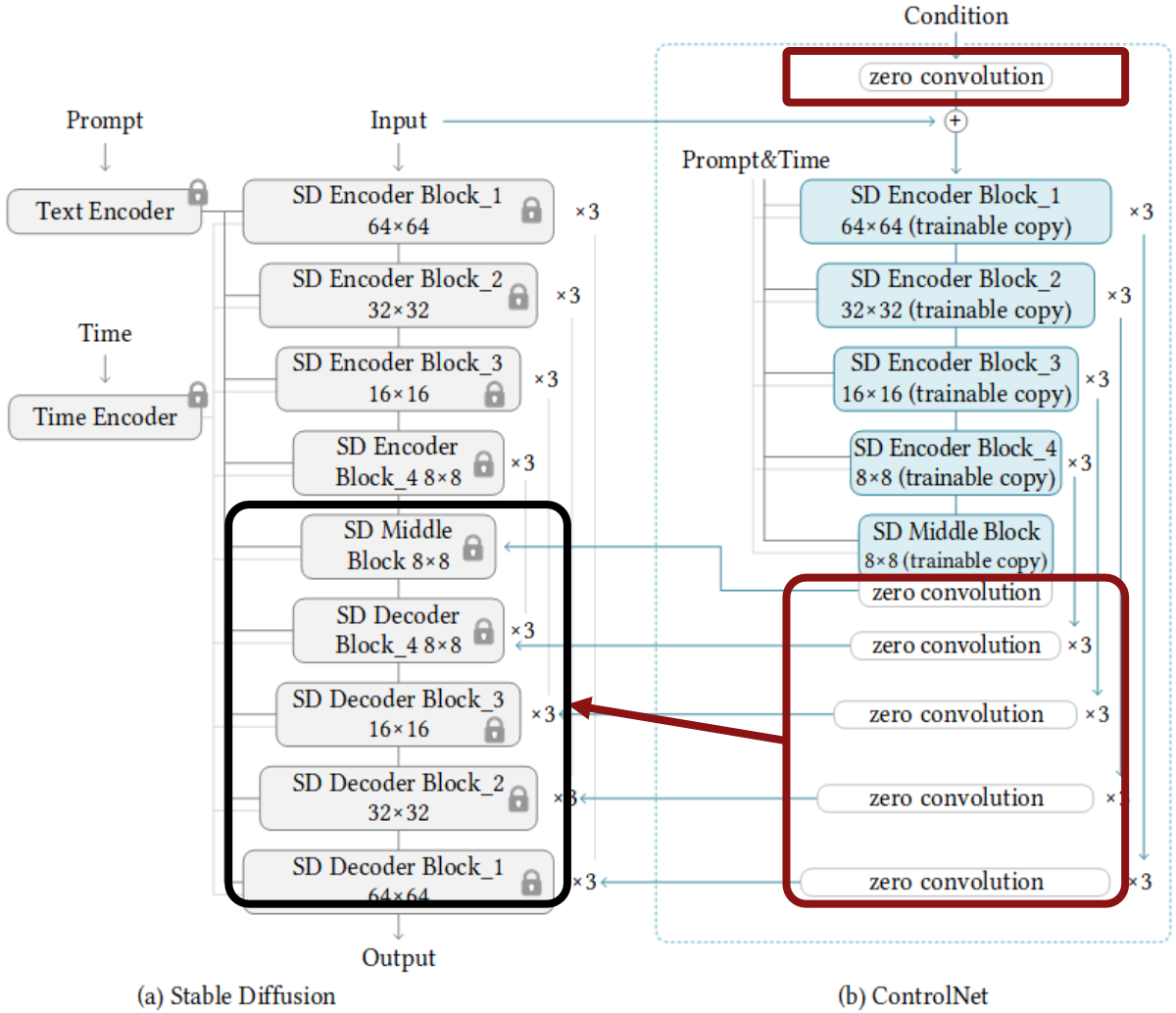
1. Freeze the noise prediction network.

ControlNet [Zhang et al., 2023]



2. For the encoding of the conditional image, copy the pretrained encoder parameters while allowing them to be updated during fine-tuning.

ControlNet [Zhang et al., 2023]



3. Combine the encoded conditional image information with the noisy image using zero convolution.

ControlNet [Zhang et al., 2023]

Zero Convolution Z is a 1×1 convolution layer with learnable weight (scaling) parameters $\boldsymbol{a}$ and bias (offset) parameters $\boldsymbol{b}$, both of which are initialized with zero:

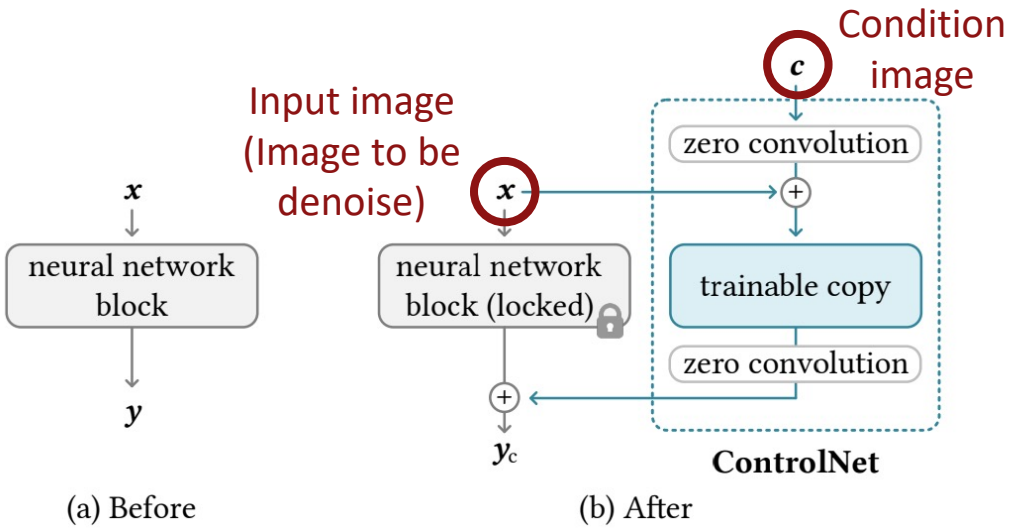
$$Z(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a} \cdot \boldsymbol{x} + \boldsymbol{b}$$

Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.

ControlNet [Zhang et al., 2023]

ControlNet modifies each neural network block using zero convolution as follows:

$$\boldsymbol{y_c} = F(\boldsymbol{x}; \Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c}; \boldsymbol{a_1}, \boldsymbol{b_1}); \Theta_c); \boldsymbol{a_2}, \boldsymbol{b_2})$$

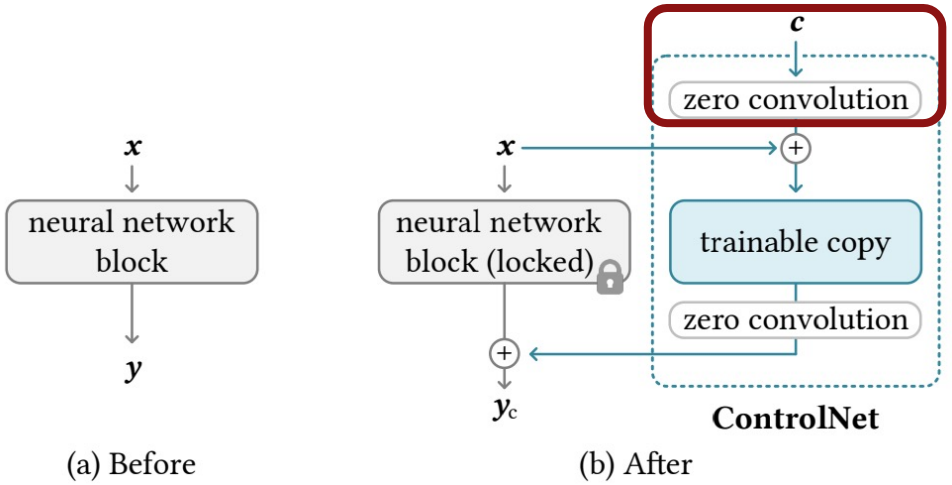


ControlNet [Zhang et al., 2023]

ControlNet modifies each neural network block using zero convolution as follows:

Zero in the beginning since $\boldsymbol{a_1} = \boldsymbol{b_1} = \mathbf{0}$.

$$\boldsymbol{y_c} = F(\boldsymbol{x}; \Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c}; \boldsymbol{a_1}, \boldsymbol{b_1}); \Theta_c); \boldsymbol{a_2}, \boldsymbol{b_2})$$

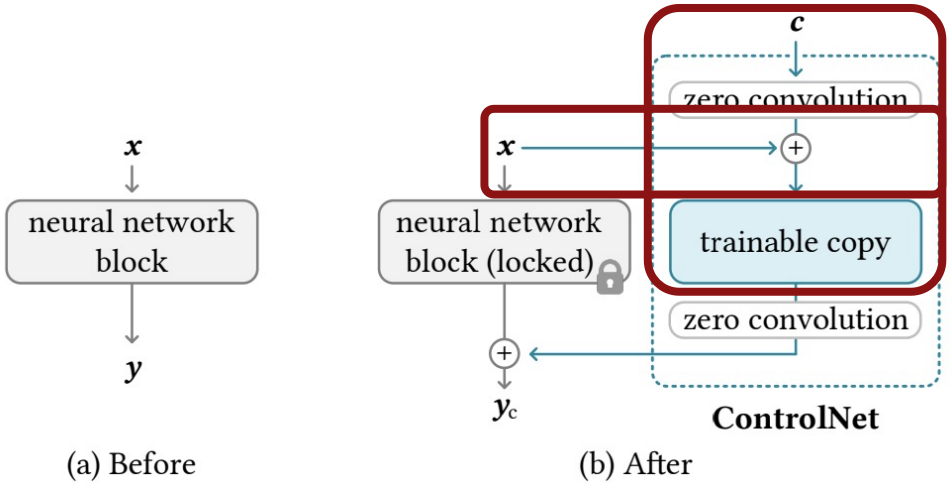


ControlNet [Zhang et al., 2023]

ControlNet modifies each neural network block using zero convolution as follows:

Same as $F(x; \Theta)$ in the beginning since $\Theta_c = \Theta$.

$$\boldsymbol{y_c} = F(\boldsymbol{x}; \Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c}; \boldsymbol{a_1}, \boldsymbol{b_1}); \Theta_c); \boldsymbol{a_2}, \boldsymbol{b_2})$$

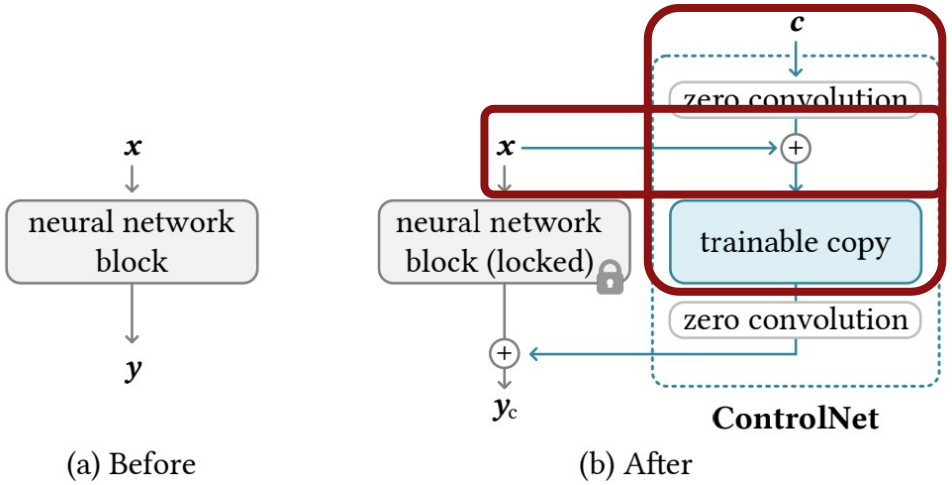


ControlNet [Zhang et al., 2023]

For encoding of the conditional image c, begin with the state where the input is x, while gradually incorporating c.

Same as $F(x; \Theta)$ in the beginning since $\Theta_c = \Theta$.

$$y_c = F(x; \Theta) + Z(F(x + Z(c; a_1, b_1); \Theta_c); a_2, b_2)$$

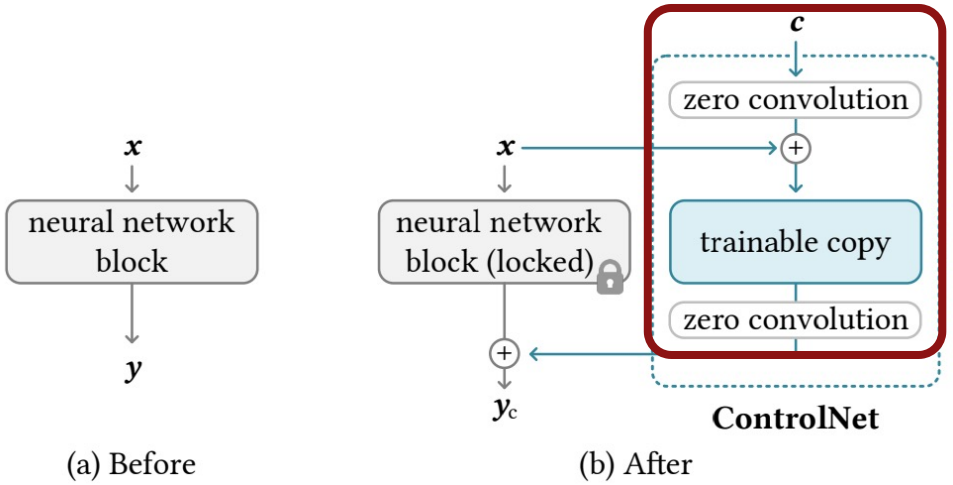


ControlNet [Zhang et al., 2023]

ControlNet modifies each neural network block using zero convolution as follows:

Zero in the beginning since $\boldsymbol{a_2} = \boldsymbol{b_2} = \mathbf{0}$.

$$\boldsymbol{y_c} = F(\boldsymbol{x}; \Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c}; \boldsymbol{a_1}, \boldsymbol{b_1}); \Theta_c); \boldsymbol{a_2}, \boldsymbol{b_2})$$

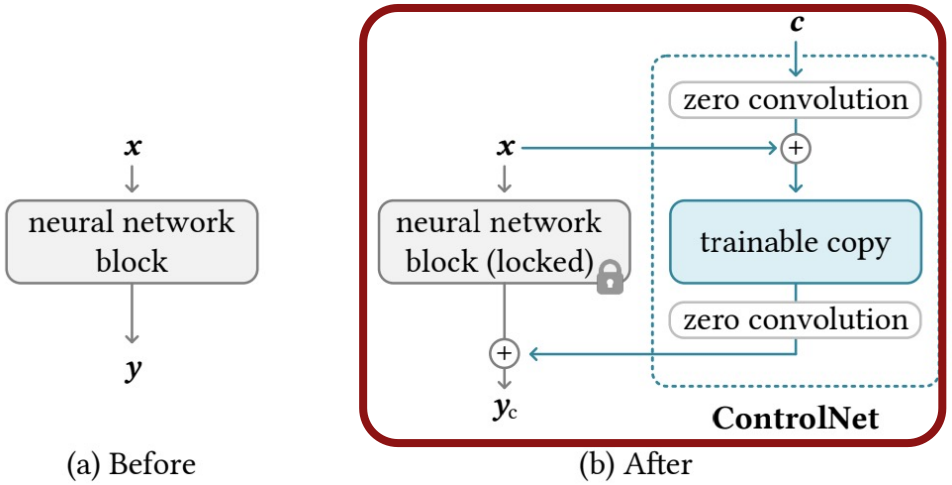


ControlNet [Zhang et al., 2023]

ControlNet modifies each neural network block using zero convolution as follows:

Same as $\mathbf{F}(\mathbf{x}; \boldsymbol{\Theta})$ in the beginning.

$$\boldsymbol{y_c} = F(\boldsymbol{x}; \Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c}; \boldsymbol{a_1}, \boldsymbol{b_1}); \Theta_c); \boldsymbol{a_2}, \boldsymbol{b_2})$$

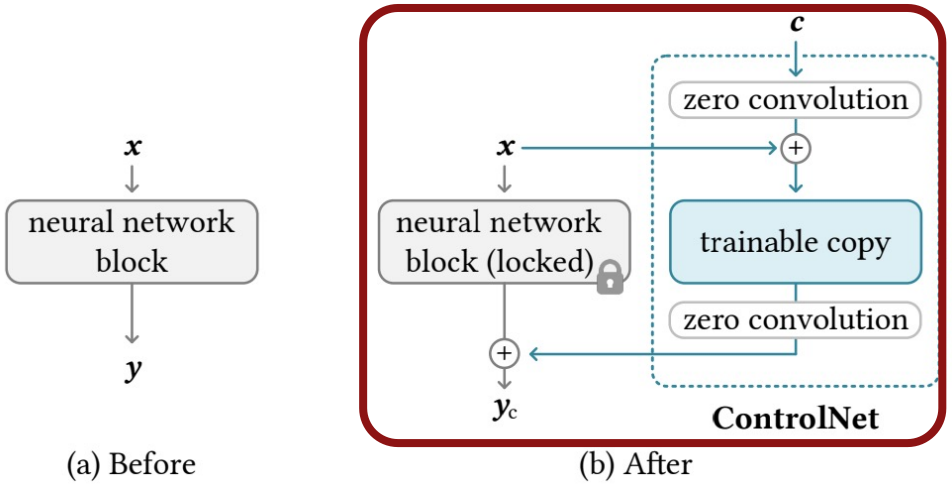


ControlNet [Zhang et al., 2023]

Also for the noise prediction, gradually incorporate the output of conditional image encoding.

Same as $\mathbf{F}(\mathbf{x}; \boldsymbol{\Theta})$ in the beginning.

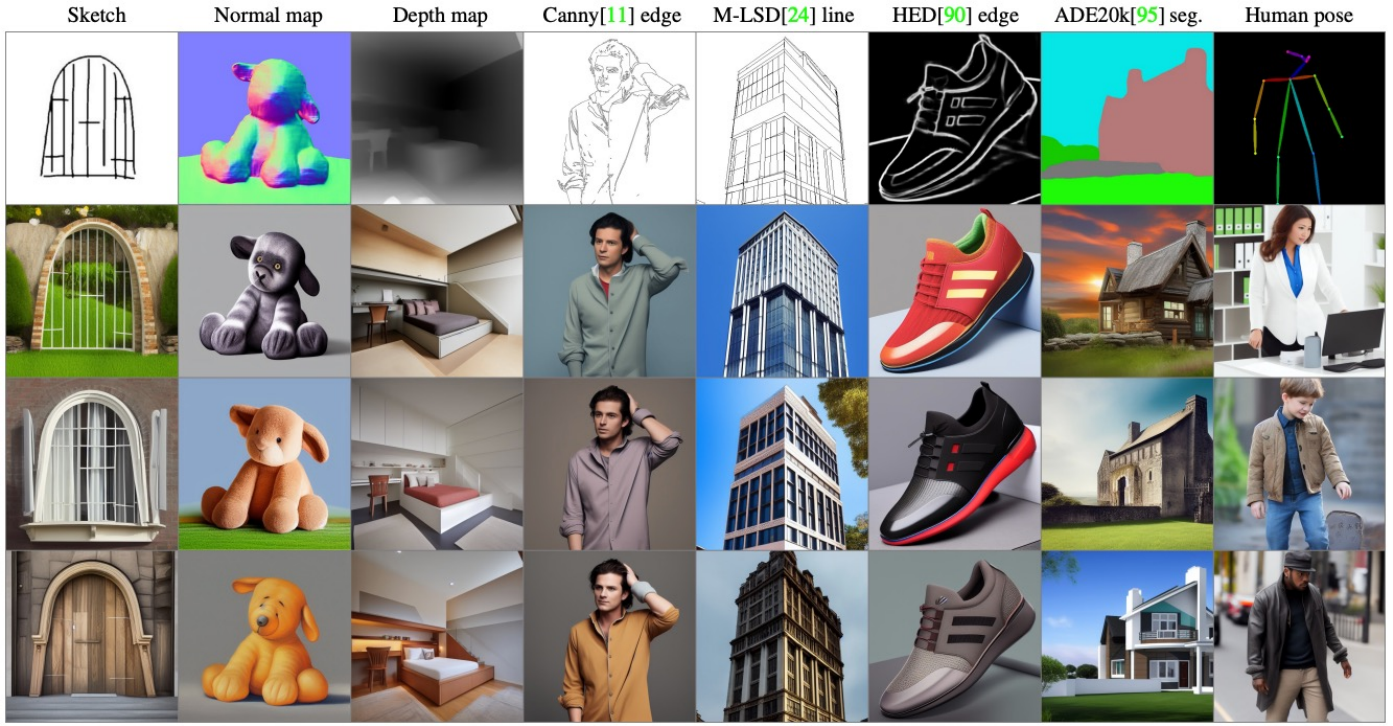
$$\boldsymbol{y_c} = F(\boldsymbol{x}; \Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c}; \boldsymbol{a_1}, \boldsymbol{b_1}); \Theta_c); \boldsymbol{a_2}, \boldsymbol{b_2})$$



Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.

ControlNet [Zhang et al., 2023]

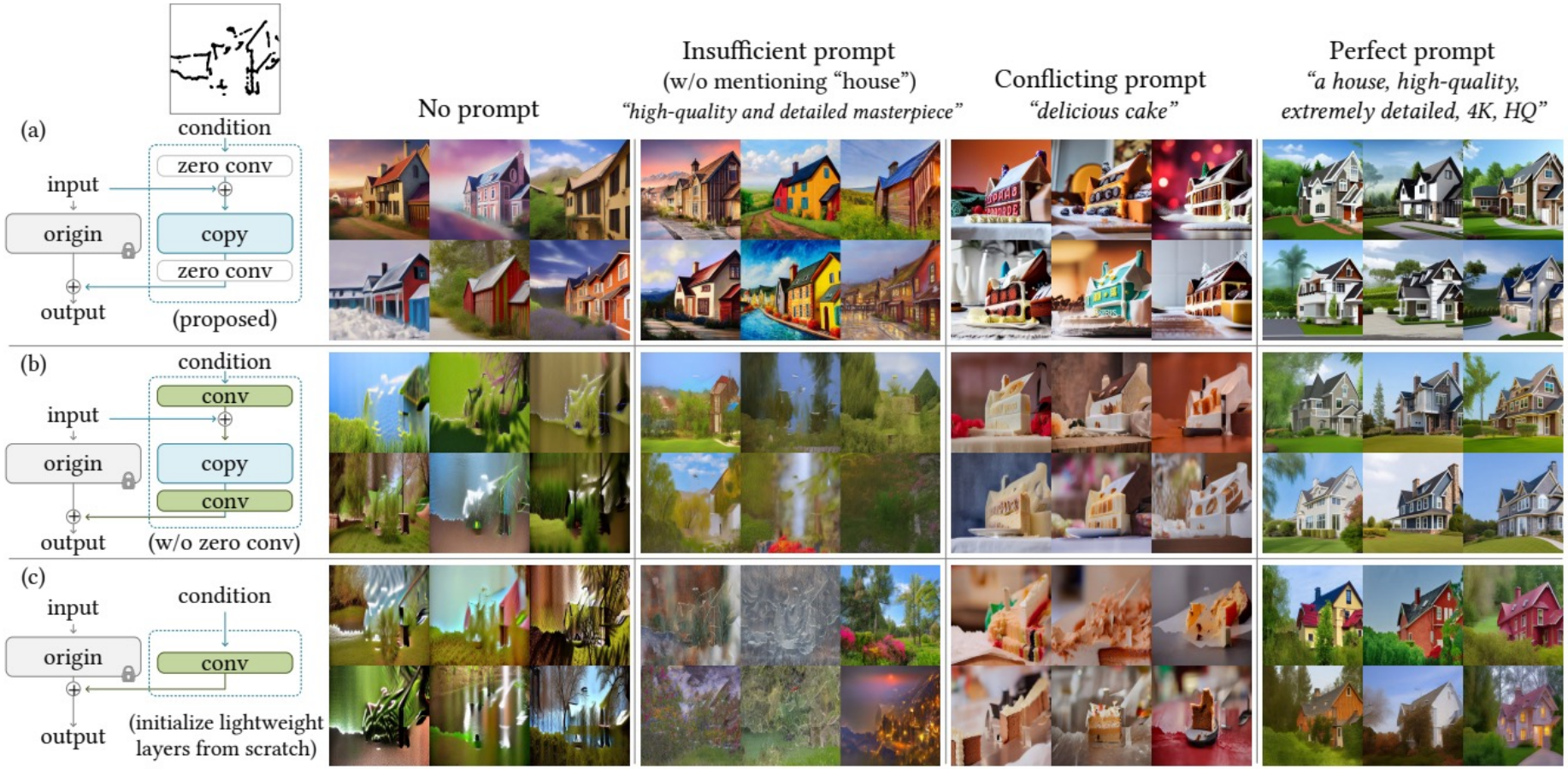
- The idea can be used for any image conditioning.
- The training dataset is " ~100k in size, which is 50,000 time smaller than the LAION-5B."



Zhang et al., Adding Conditional Control to Text-to-Image Diffusion Models, ICCV 2022.

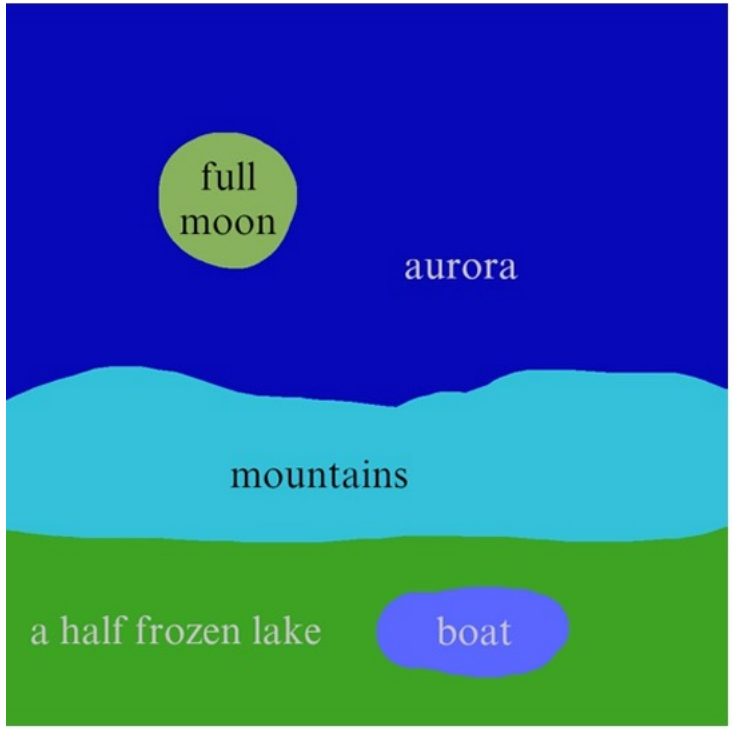
ControlNet [Zhang et al., 2023]

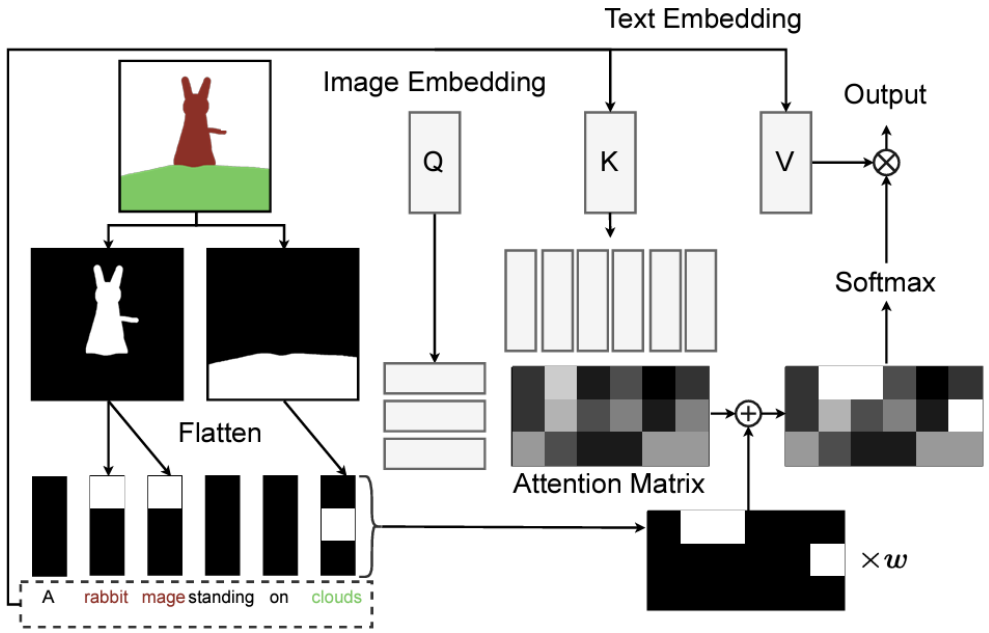
Ablative study of different architectures.



Adherence of Conditioning Signal

Regional Control of Conditioning Signal





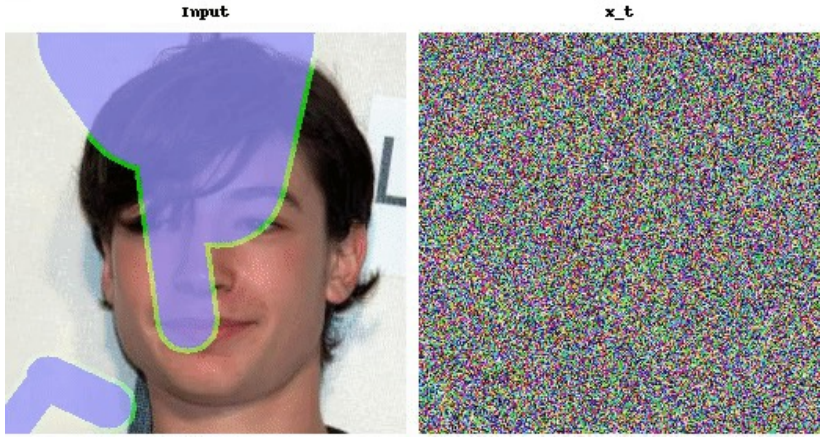
cross attention layer

Zero-Shot Applications

Zero-Shot Adaptations

How to edit and inpaint images using a pretrained image diffusion model even without the need for fine-tuning?

Meng et al., SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations, ICLR 2022.

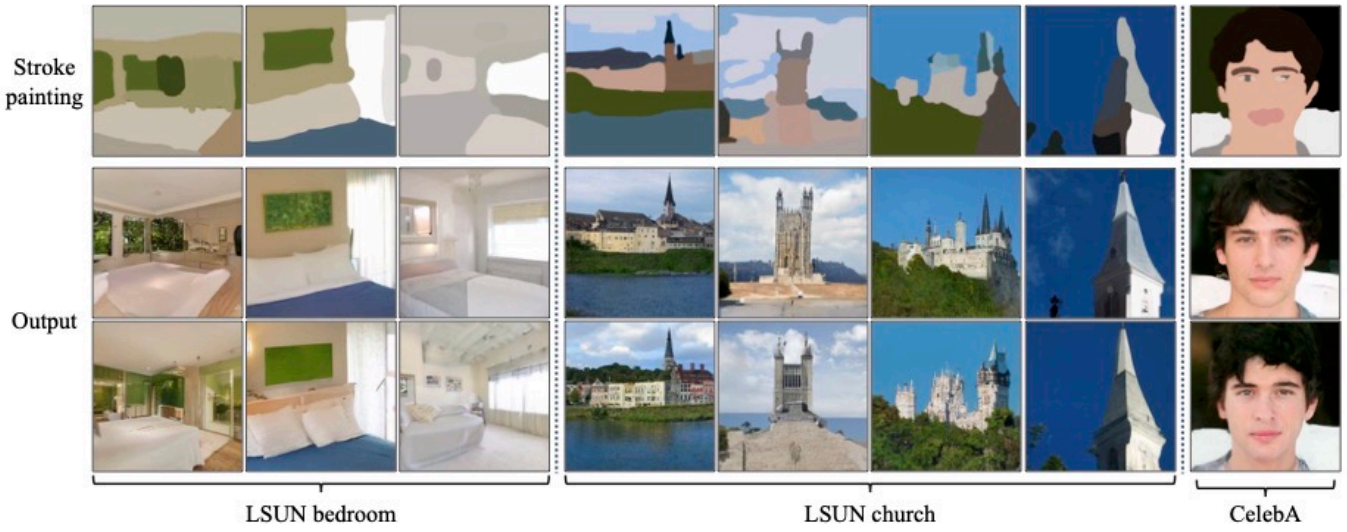


Lugmayr et al., RePaint: Inpainting using Denoising Diffusion Probabilistic Models, CVPR 2022.

SDEdit [Meng et al., 2022]

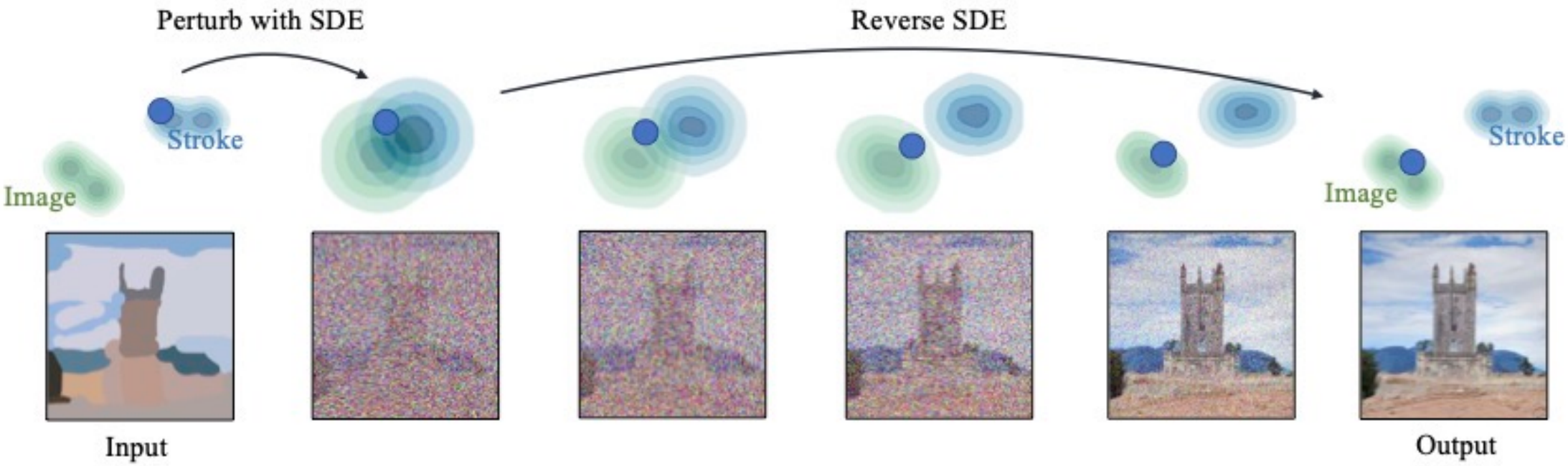
Image generation/editing through user interaction

- Image generation from sketches
- Image editing from scribbles



SDEdit [Meng et al., 2022]

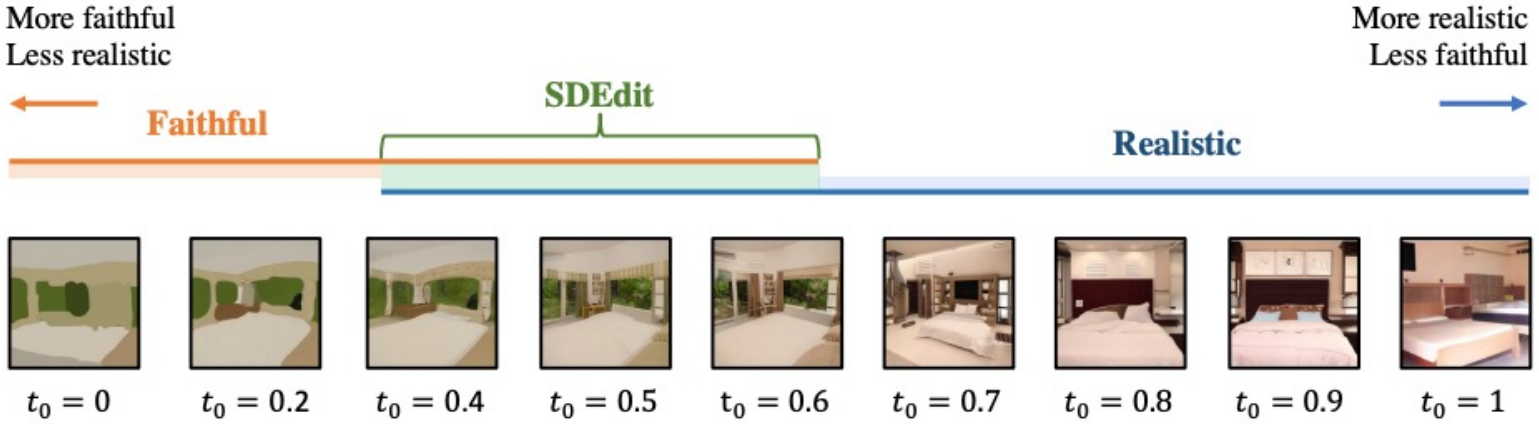
Perform the forward process for a bit and then reverse the process.



SDEdit [Meng et al., 2022]

Realism vs. faithfulness

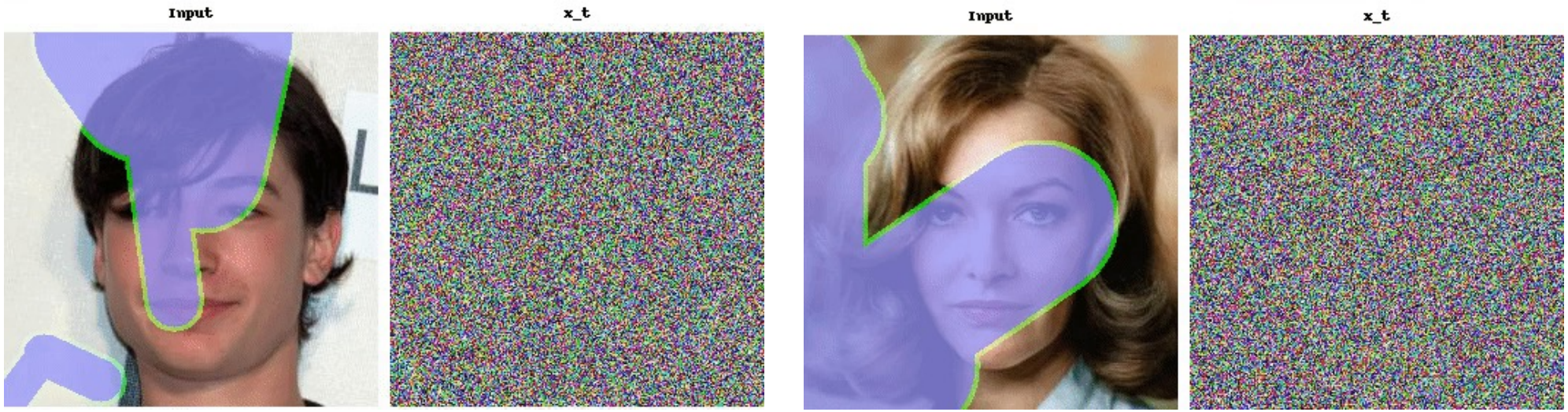
As you perform the forward process with a longer timestep, the output image becomes more realistic but less faithful (deviating from the given condition).



Lugmayr et al., RePaint: Inpainting using Denoising Diffusion Probabilistic Models, CVPR 2022.

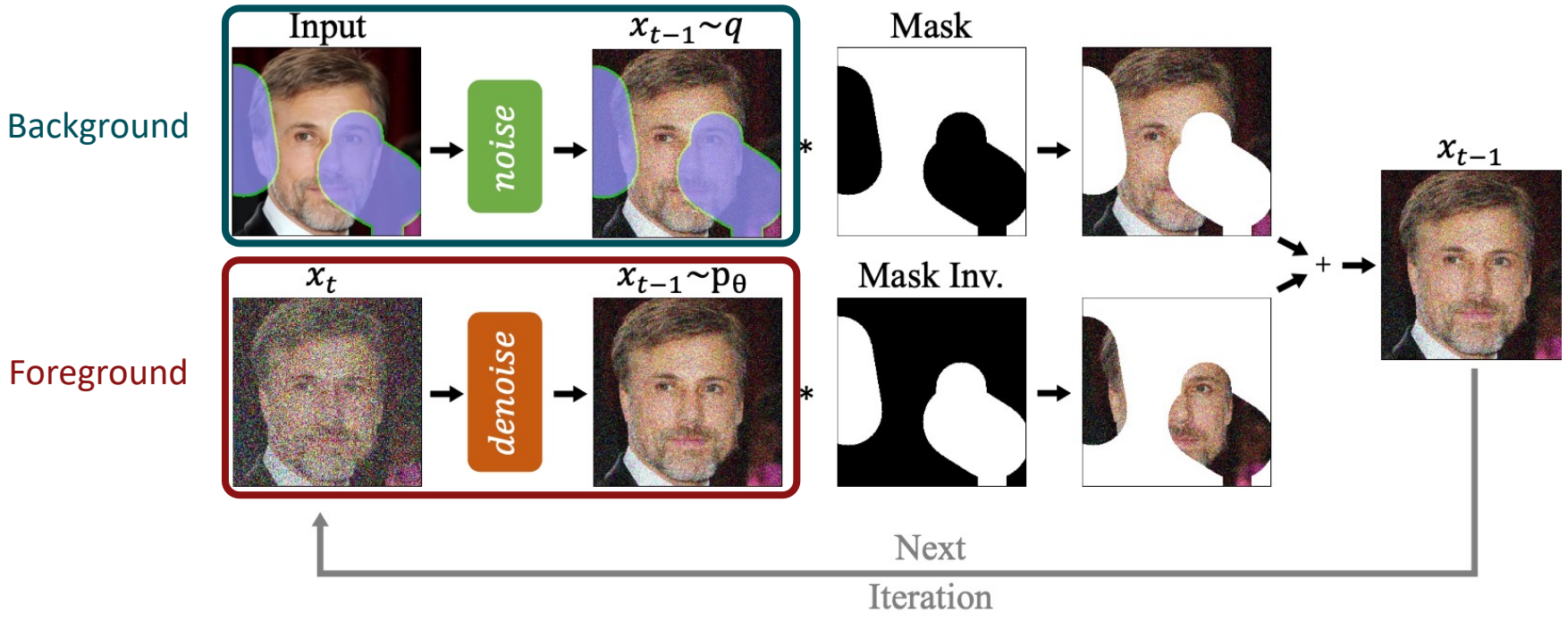
RePaint [Lugmayr et al., 2022]

- Filling regions in an image using a pretrained image diffusion model.
- Assume that the missing foreground is filled while the background is fixed.



RePaint [Lugmayr et al., 2022]

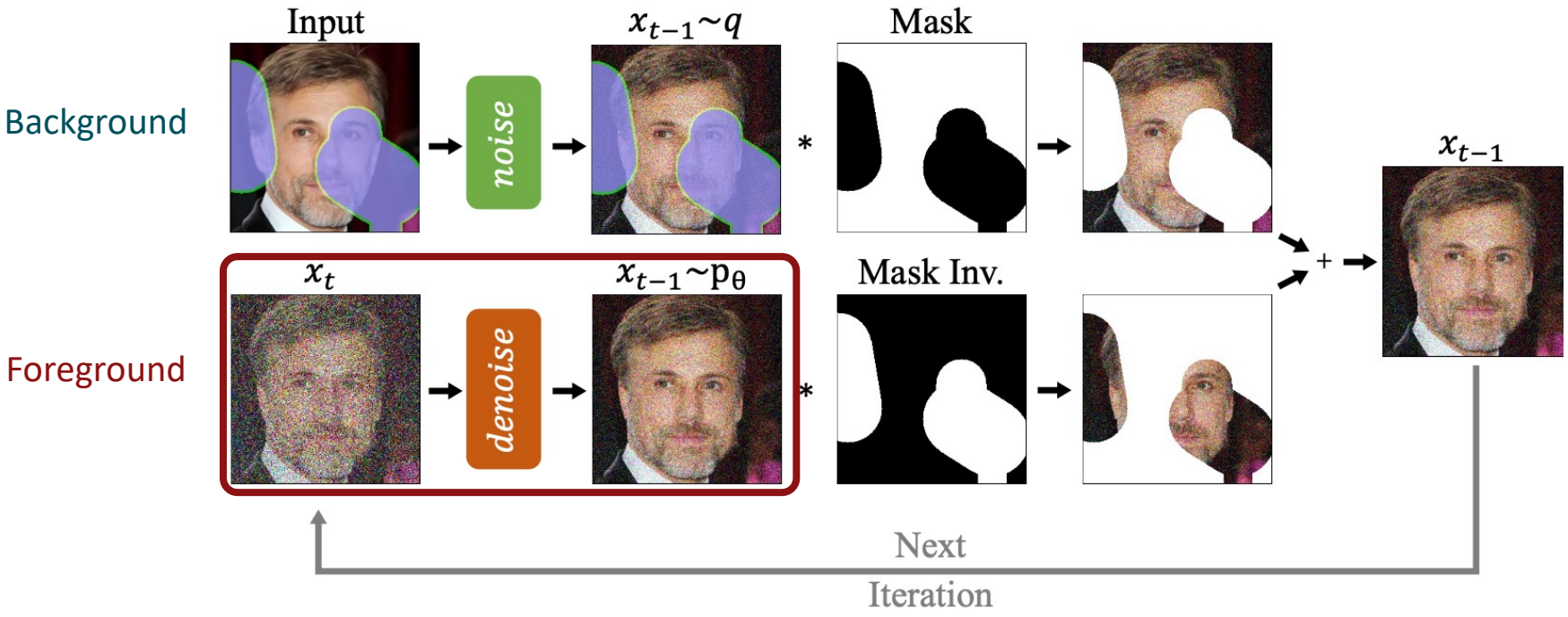
Combine denoised foreground images (to be filled) and noisy background (to be fixed) images at each iteration of the reverse process.



RePaint [Lugmayr et al., 2022]

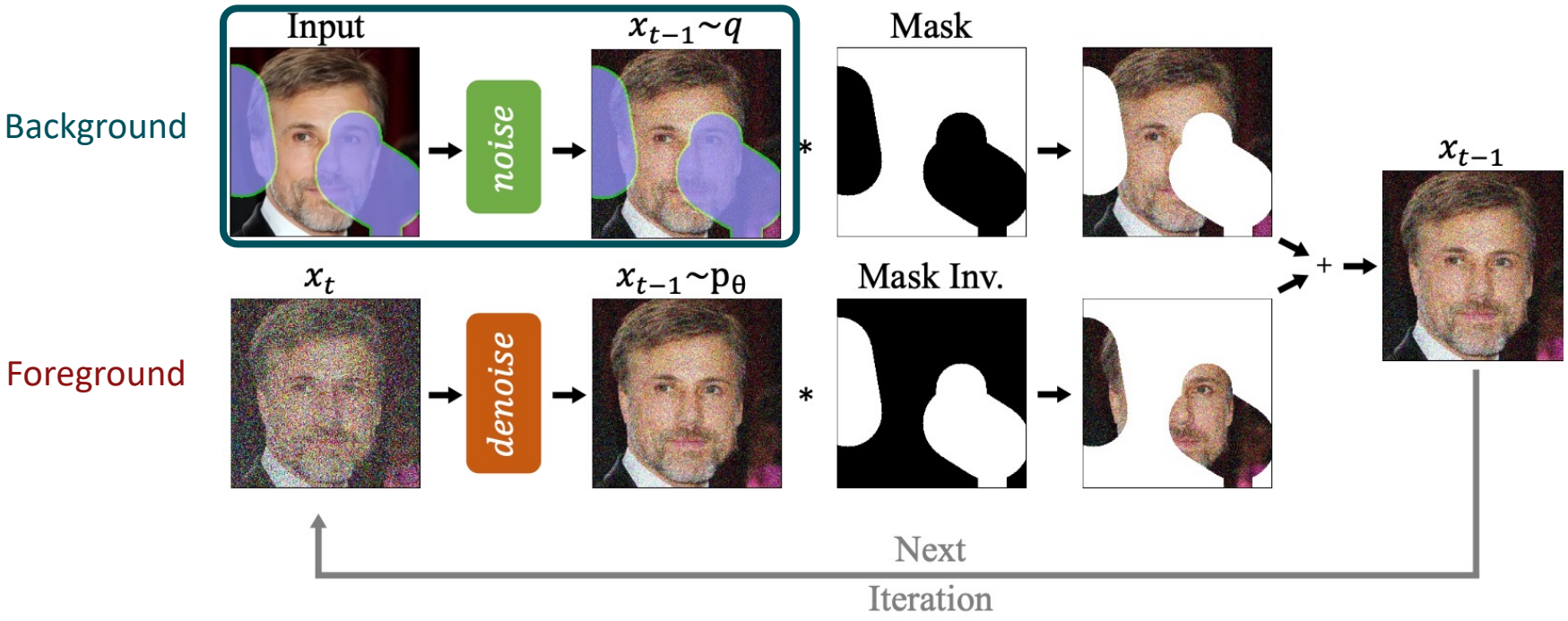
1. Starting from $\boldsymbol{x}_T$, at each timestep t,

   denoise $\boldsymbol{x}_t$ one step (reverse process), resulting $\boldsymbol{x}_{t-1}^{f}$.



RePaint [Lugmayr et al., 2022]

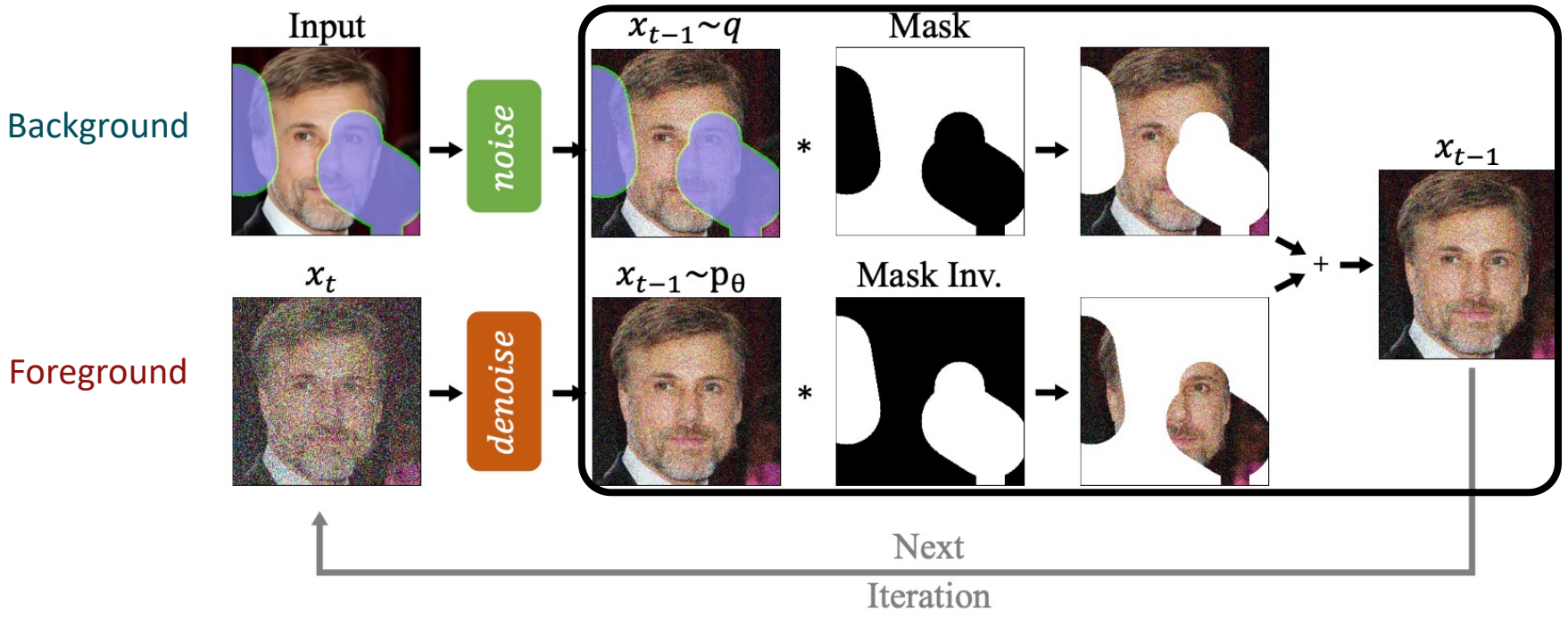
2. Perturb the input background image $\boldsymbol{x}_0^b$ (forward process) with a noise scale of the timestep $t-1$, resulting $\boldsymbol{x}_{t-1}^b$.



RePaint [Lugmayr et al., 2022]

3. Combine $\boldsymbol{x}_{t-1}^{b}$ and $\boldsymbol{x}_{t-1}^{f}$ (where M is the background mask):

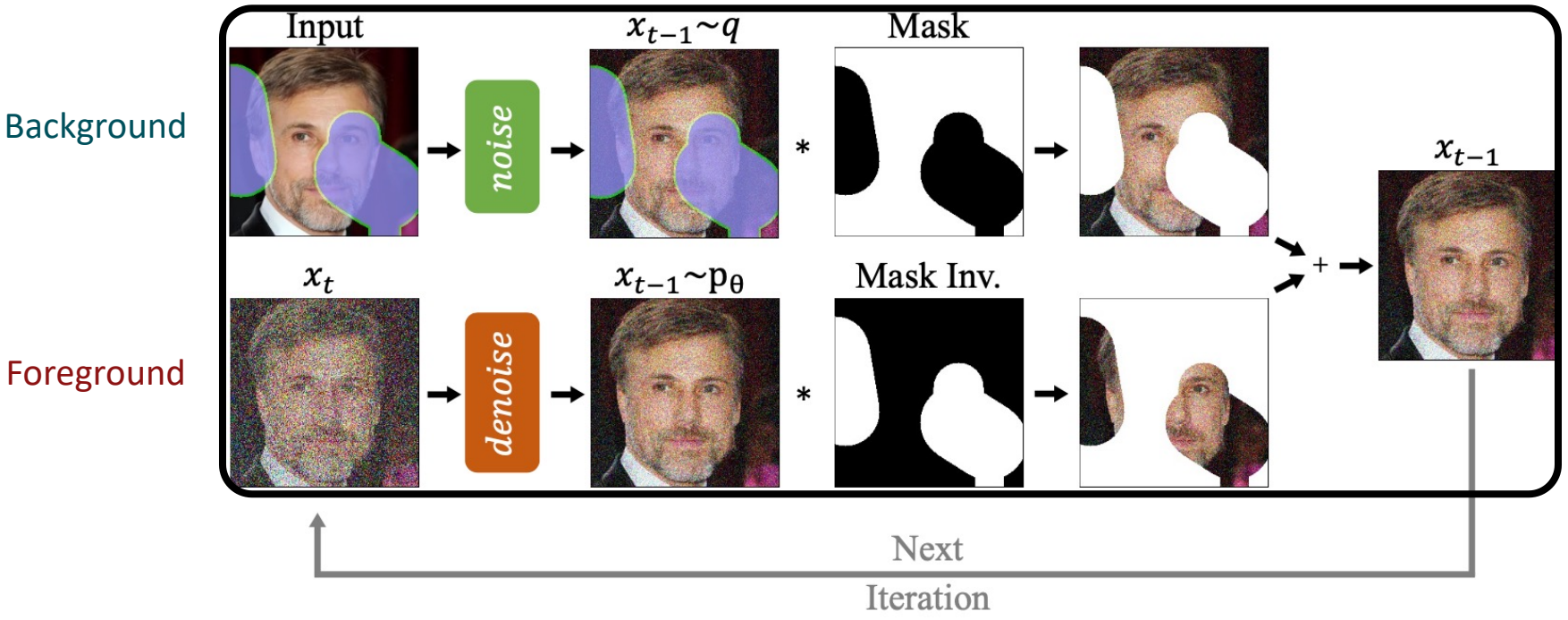
$$\boldsymbol{x}_{t-1} = M \odot \boldsymbol{x}_{t-1}^{b} + (1 - M) \odot \boldsymbol{x}_{t-1}^{f}$$



RePaint [Lugmayr et al., 2022]

4. Repeat this process for t $= T, \ldots, 1$.

(You may need to replay the forward/reverse processes in intermediate intervals.)



Lugmayr et al., RePaint: Inpainting using Denoising Diffusion Probabilistic Models, CVPR 2022.

RePaint [Lugmayr et al., 2022]

Results with different masks

Summary

1. **Classifier Guidance / Classifier-Free Guidance**
   Enhancing the quality of generated outputs using additional information.
2. **ControlNet**
   Adapting the pretrained diffusion model with relatively few conditional data using zero convolution.
3. **SDEdit / RePaint**
   Utilizing the pretrained diffusion model for editing and image inpainting.

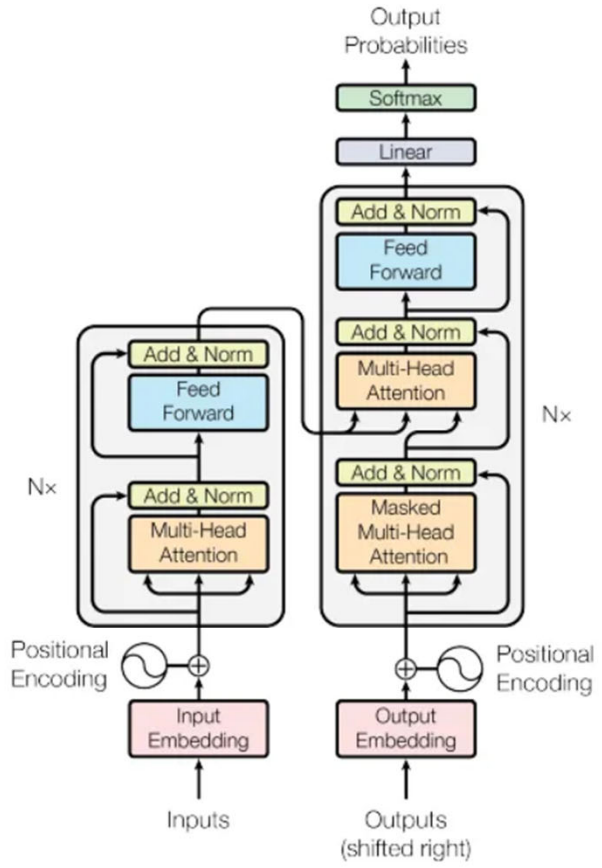
Attention Please

Attention Mechanism in Generative Models

Daniel Cohen-Or

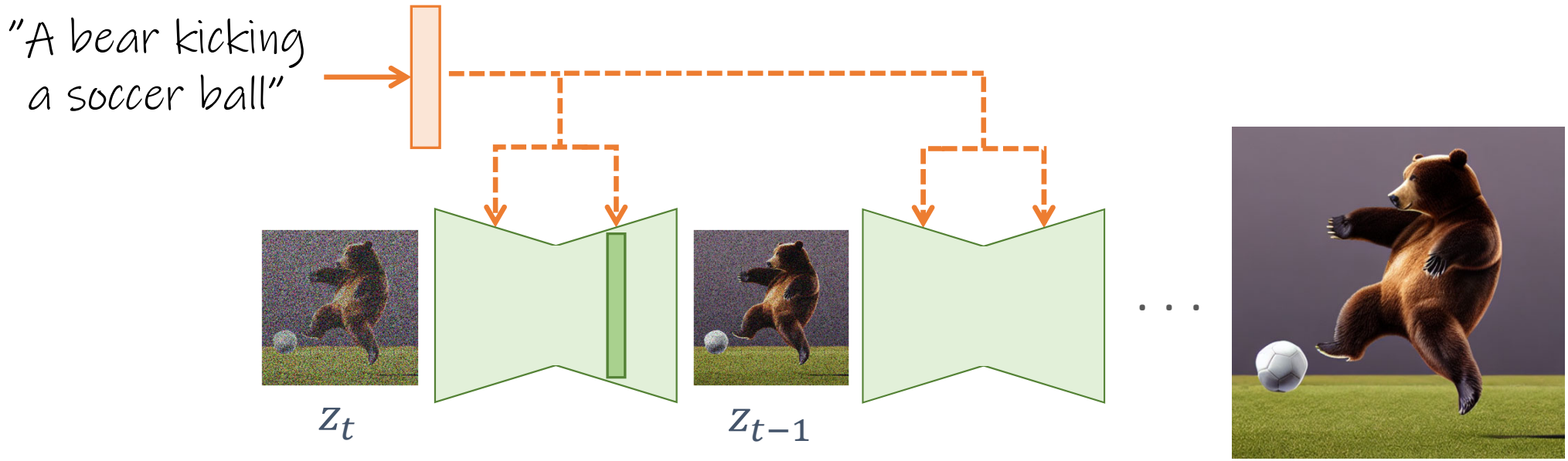
“Attention is all you need” , Waswani et al. 2007

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



From “Attention is all you need” paper by Vaswani, et al., 2017 [1]

The Denoising Process with Attention Layers



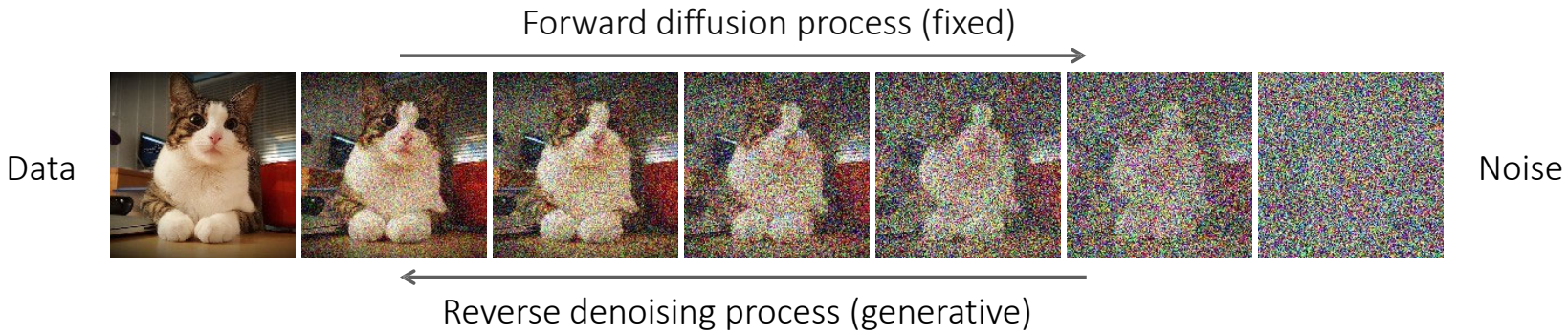
Denoising network

Denoising Diffusion Models

Learning to generate by denoising

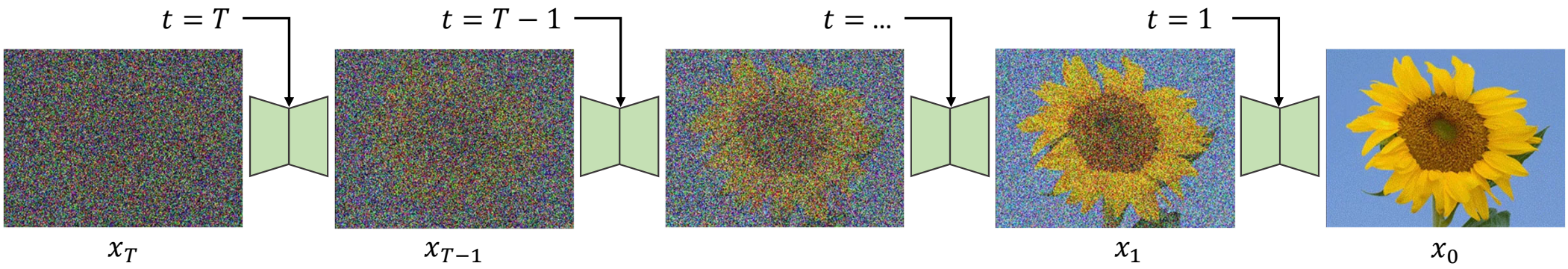
Denoising diffusion models consist of two processes:

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

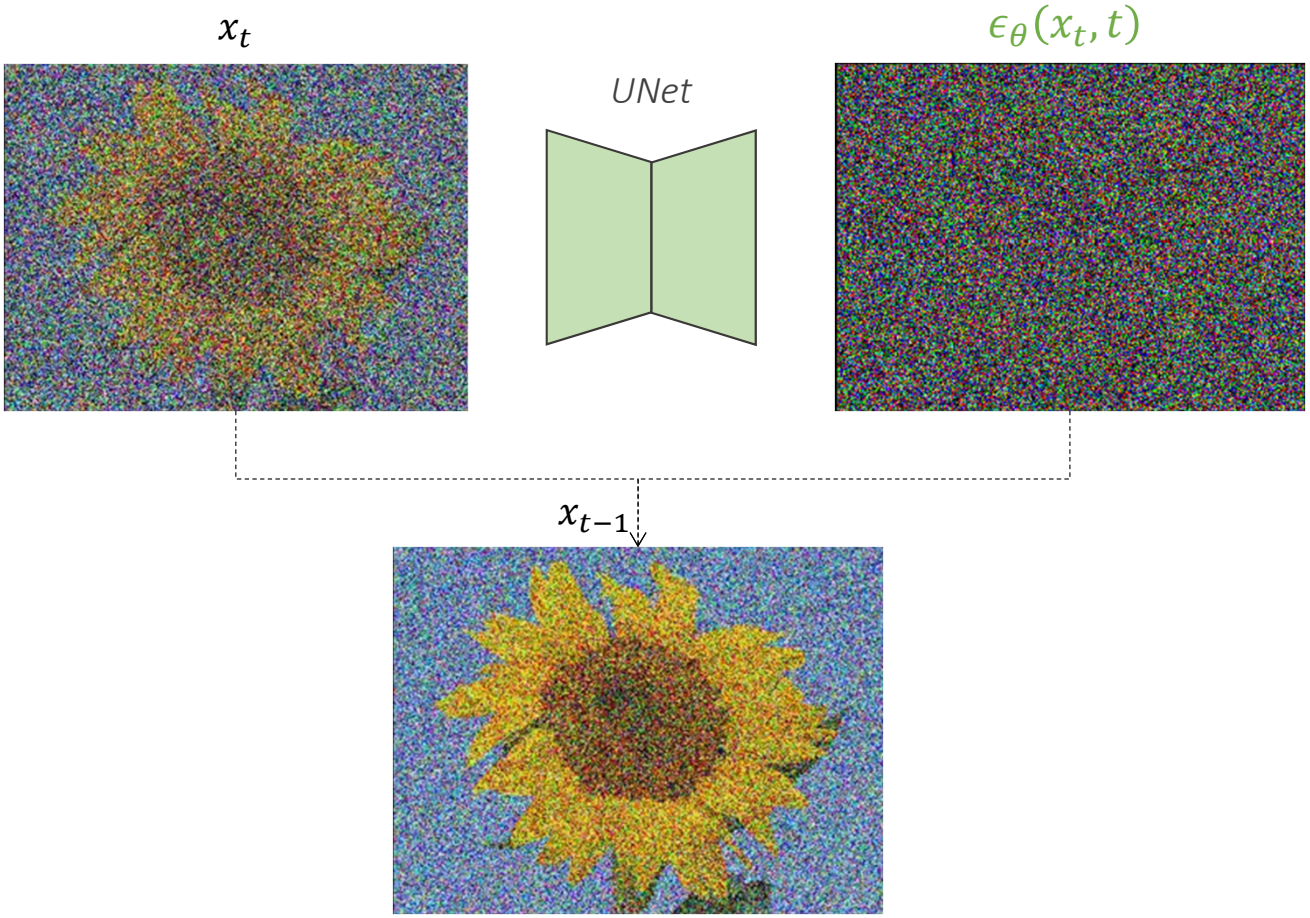


Sohl-Dickstein et al., Deep Unsupervised Learning using Nonequilibrium Thermodynamics, ICML 2015 Ho et al., Denoising Diffusion Probabilistic Models, NeurIPS 2020
Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021

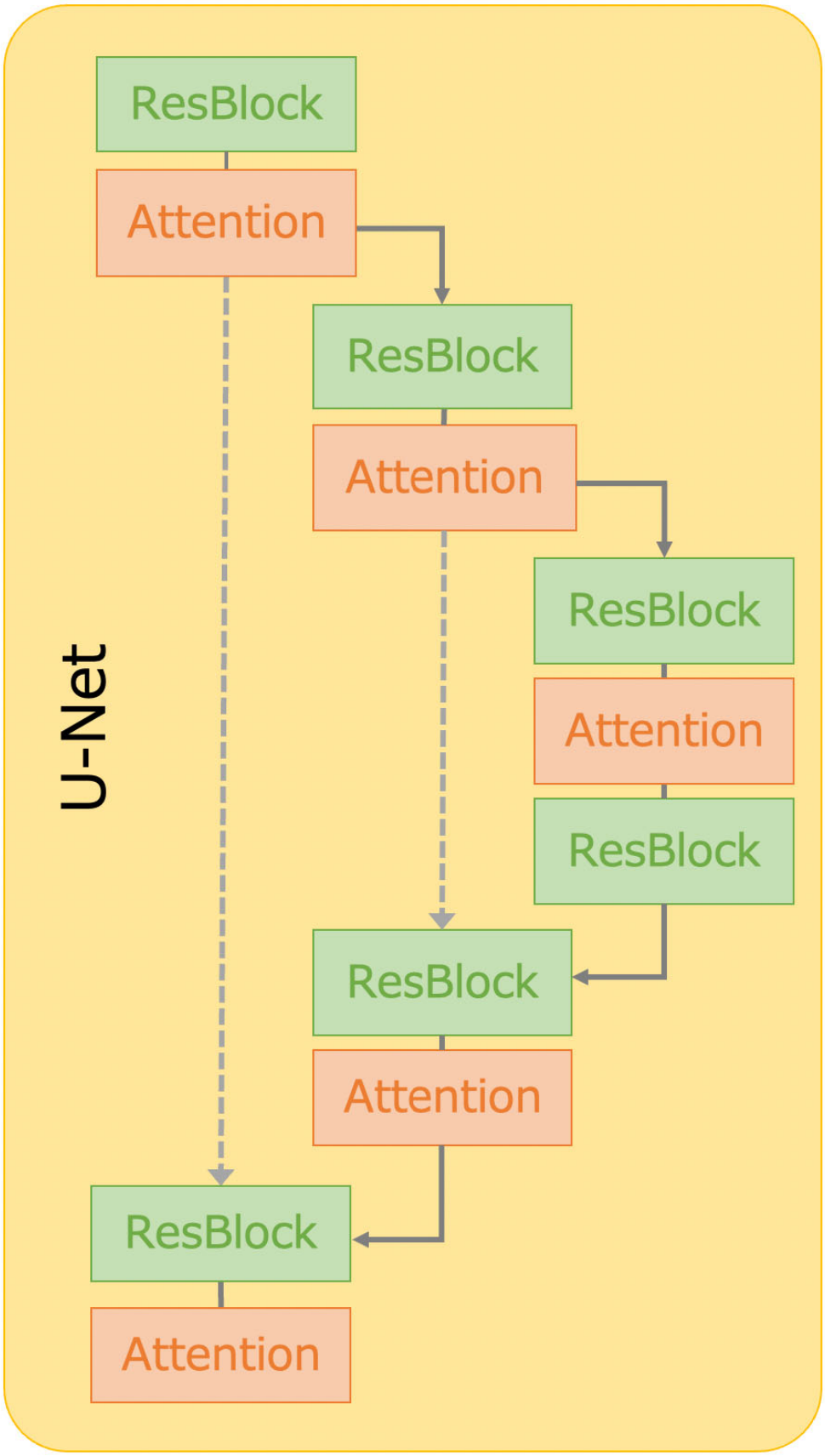
Diffusion Models

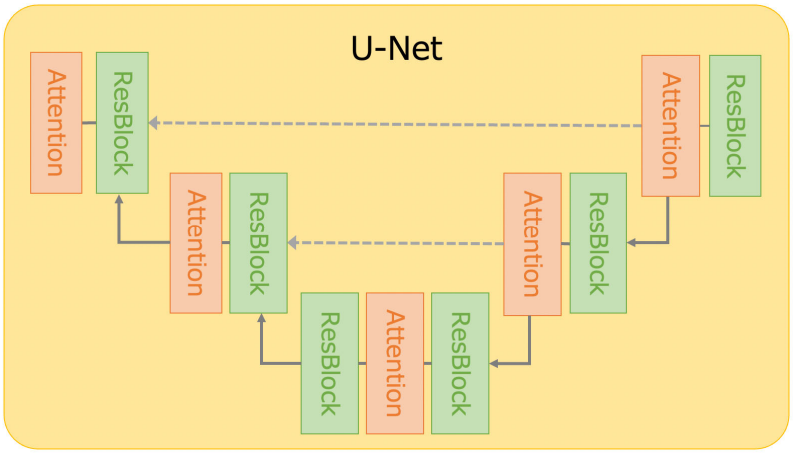


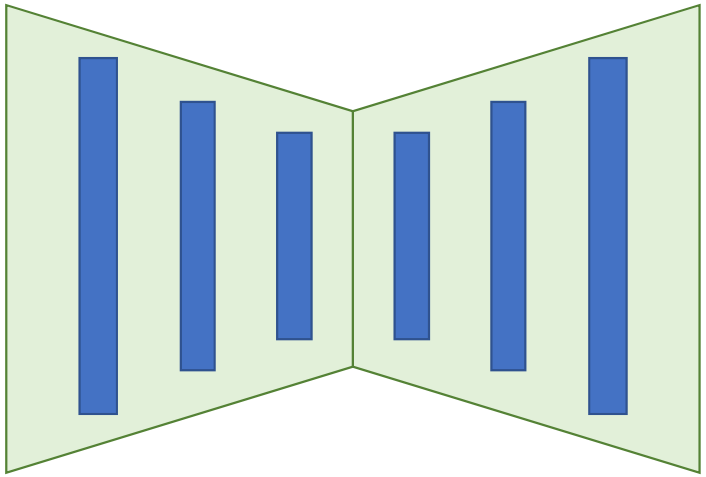
Diffusion Models



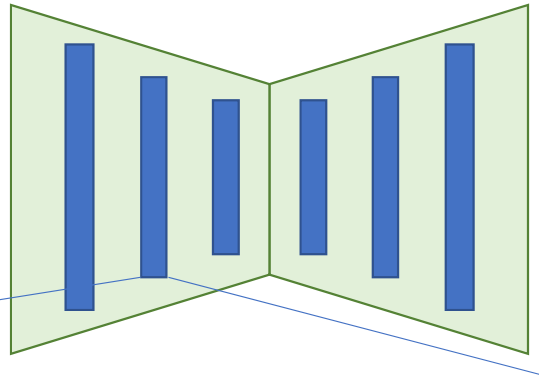
Unet

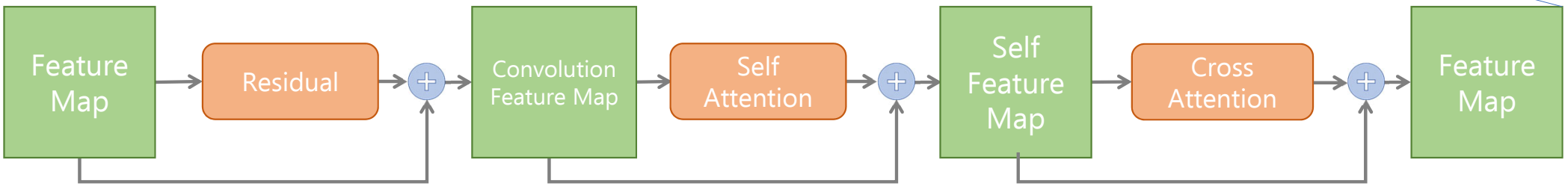

U-Net
ResBlock
Attention
ResBlock
Attention
ResBlock
Attention
ResBlock
ResBlock
Attention
ResBlock
Attention


U-Net
Attention
ResBlock
Attention
ResBlock
Attention
ResBlock
Attention
ResBlock
ResBlock
Attention
ResBlock
Attention
ResBlock



The UNet Layer

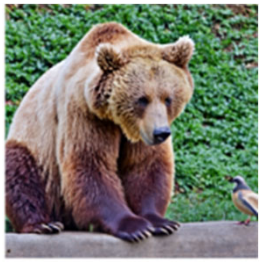




Cross-Attention

$N = 6$

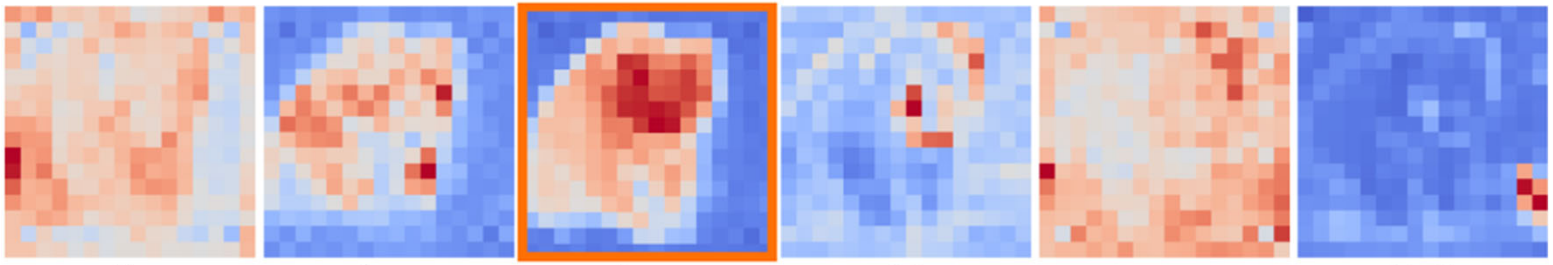
P^2



A
furry
bear
watches
a
bird

Q K

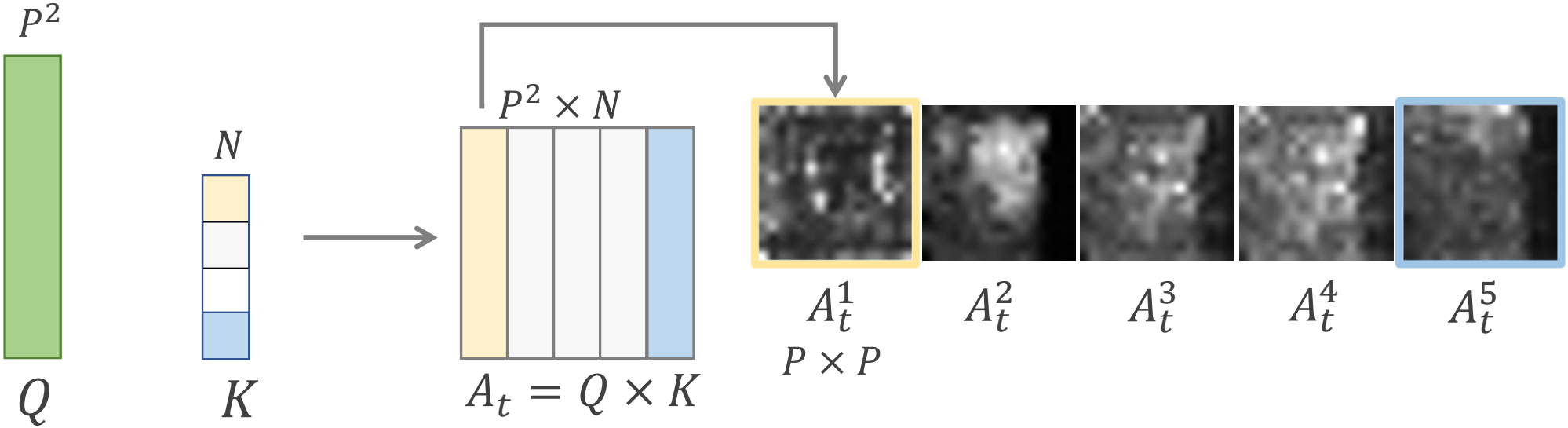
$P^2 \times N$



$$A_t = Q \times K$$

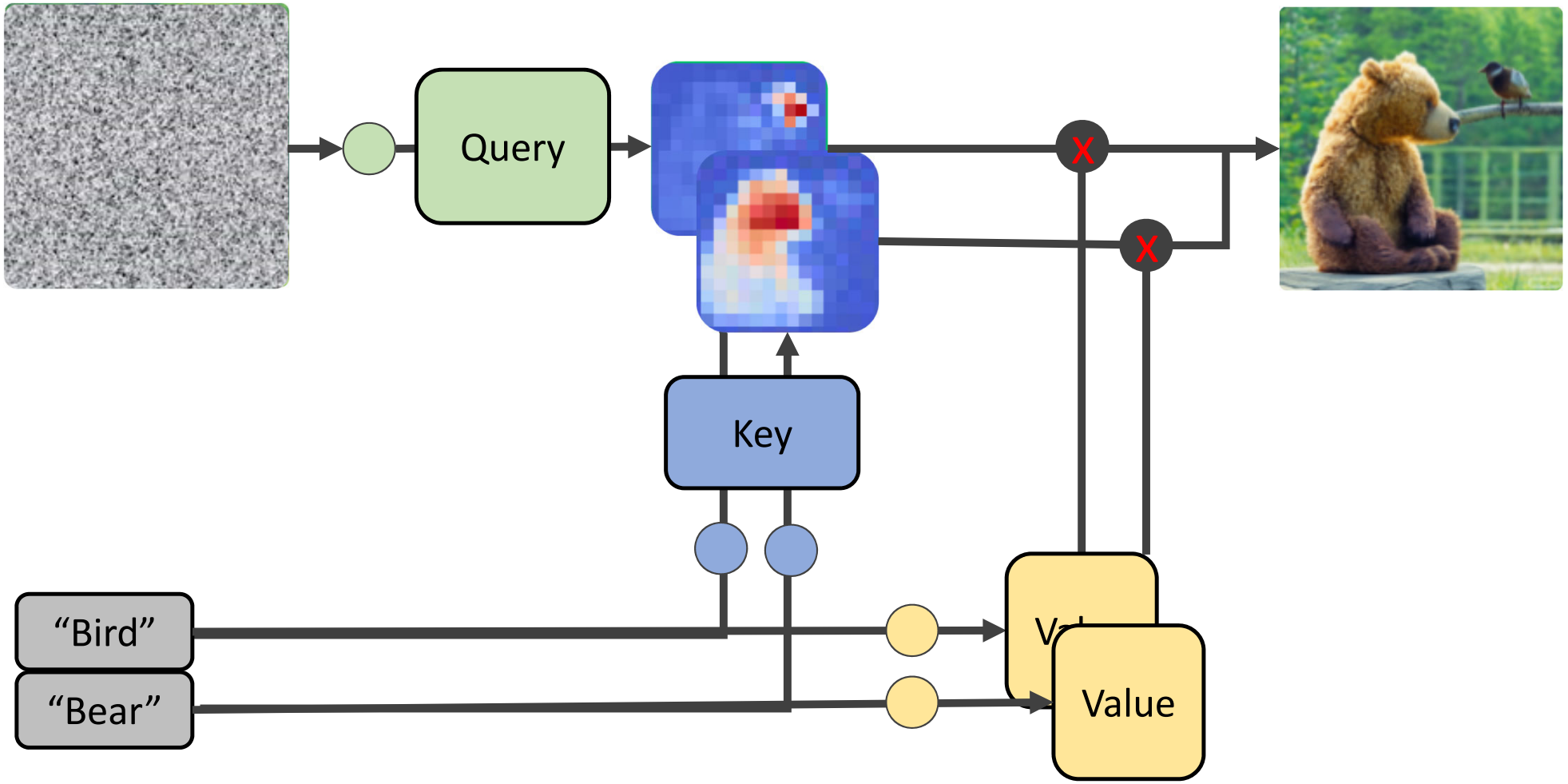
A_t[i,j,n] = "amount of information" passed from token n to patch (i,j)

The Cross-Attention

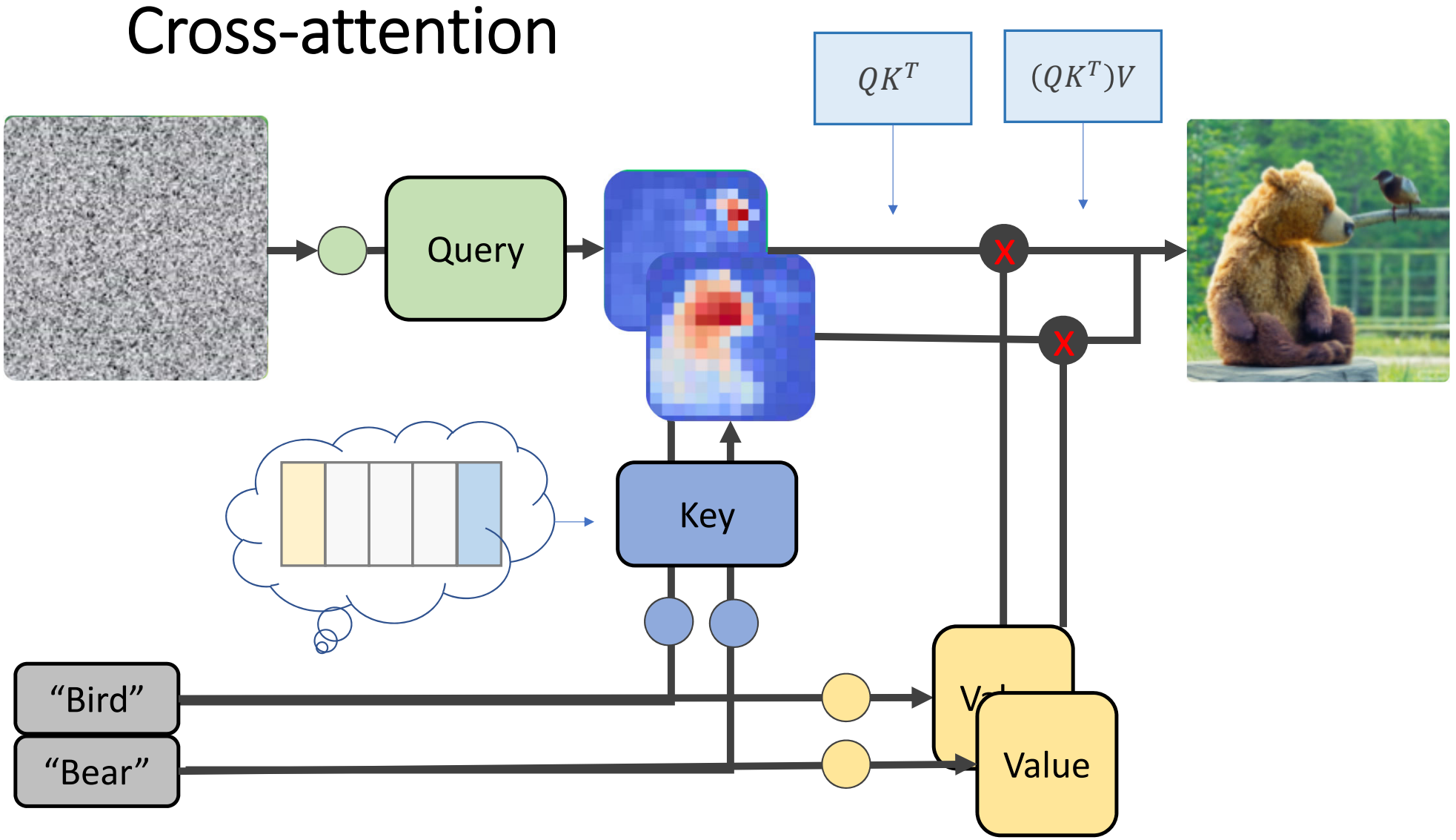


A_t[i,j,n] = "amount of information" passed from token n to patch (i,j)

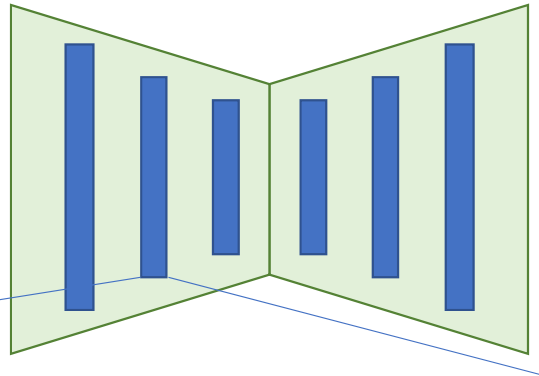
Cross-attention

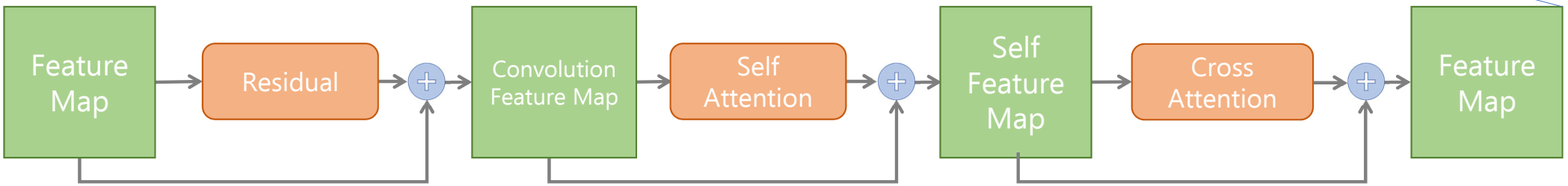


Cross-attention



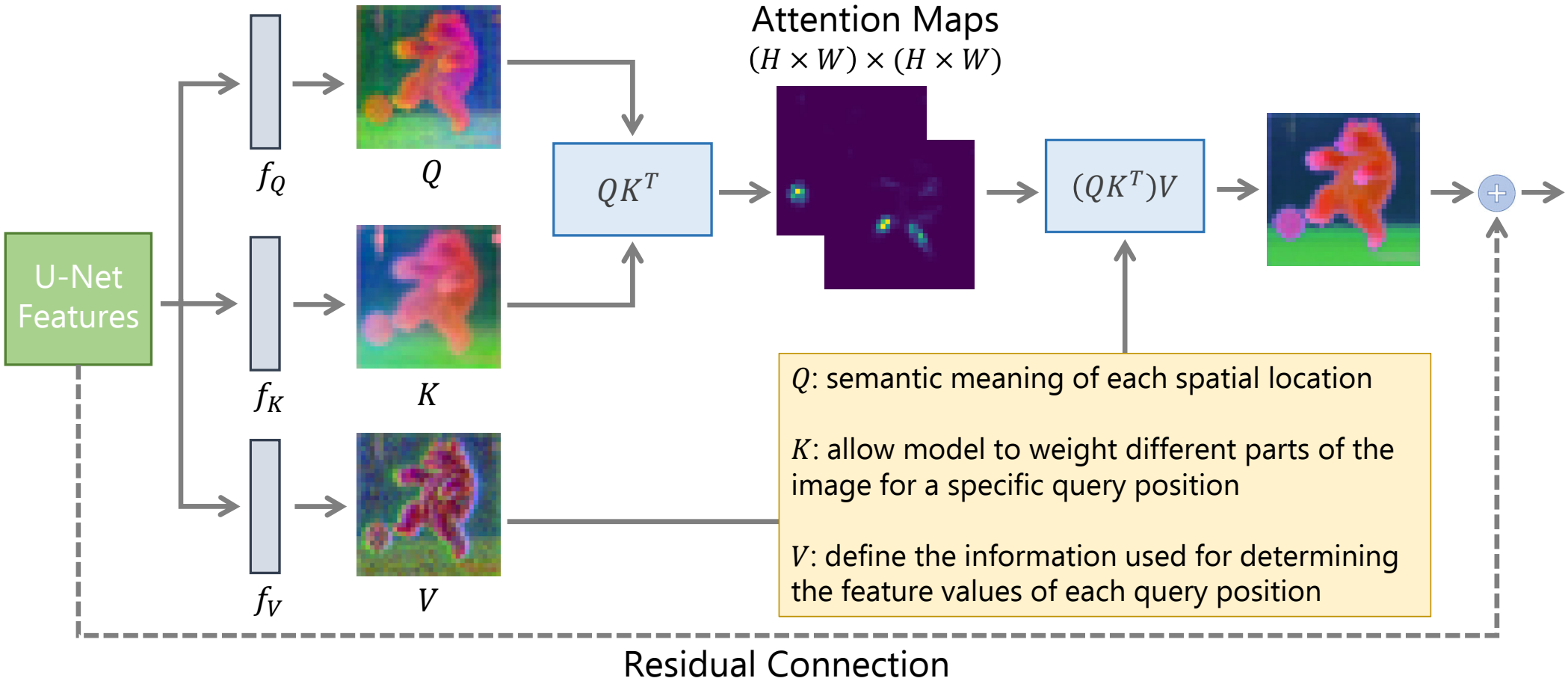
The UNet Layer





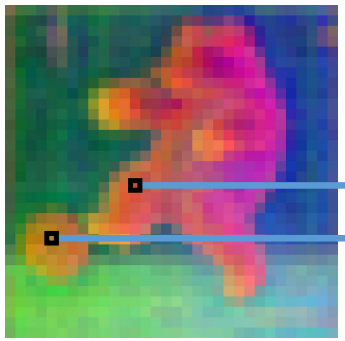
The Self-Attention

Represents where the model "looks" in the image for each spatial position in Q

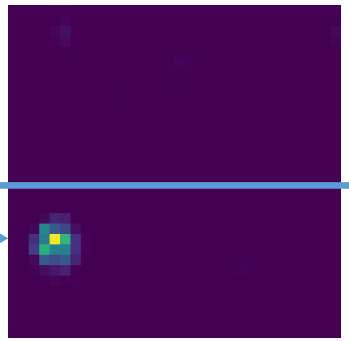


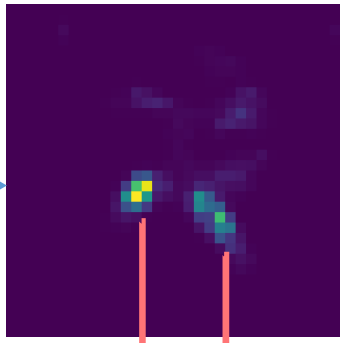
The Self-Attention

Attention Maps
$(H \times W) \times (H \times W)$

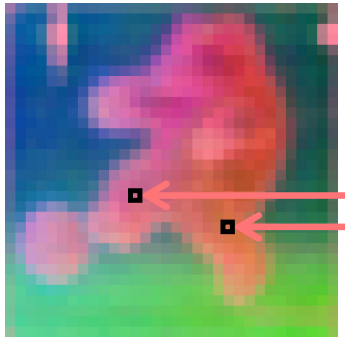


Queries
$H \times W \times C$





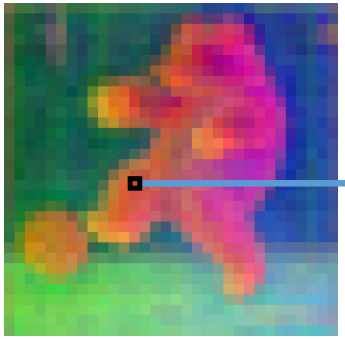
Each query defines to a
$H \times W$ attention map!



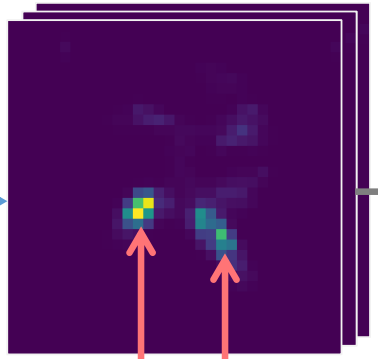
Keys
$H \times W \times C$

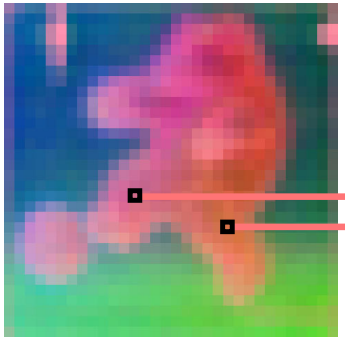
A **query** on the leg of the bear "attends"
to **keys** located on the leg of the bear!

The Self-Attention

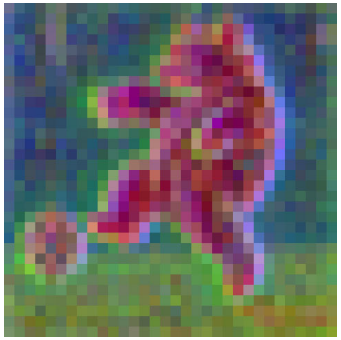


Queries
$H \times W \times C$





Keys
$H \times W \times C$



Values
$H \times W \times C$

Values
$H \times W \times C$

$(QK^T)V$

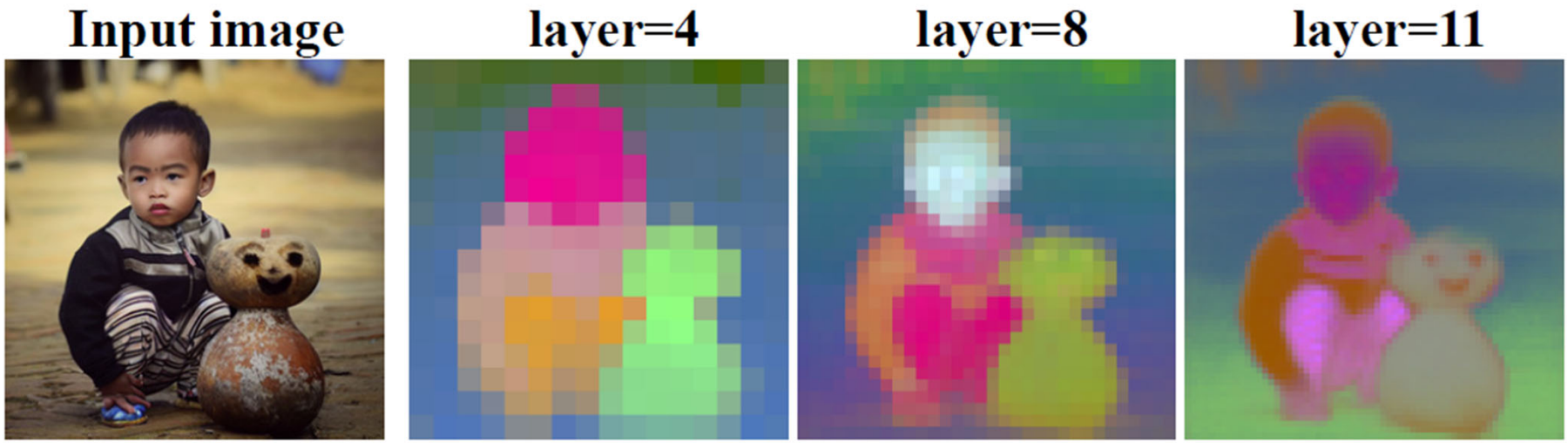
t = 0.6, layer: 10 / 70

t = 0.6, layer: 35 / 70

Self-Segmentation

Localizing Object-level Shape Variations [Patashnik et al., ICCV 2023]

Self-Attention Maps



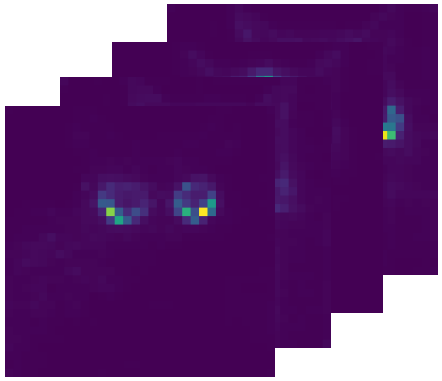
Are these PCA on the self-attention ? On what exactly the QK maps?

Plug-and-Play [Tumanyan et al., CVPR 2023]

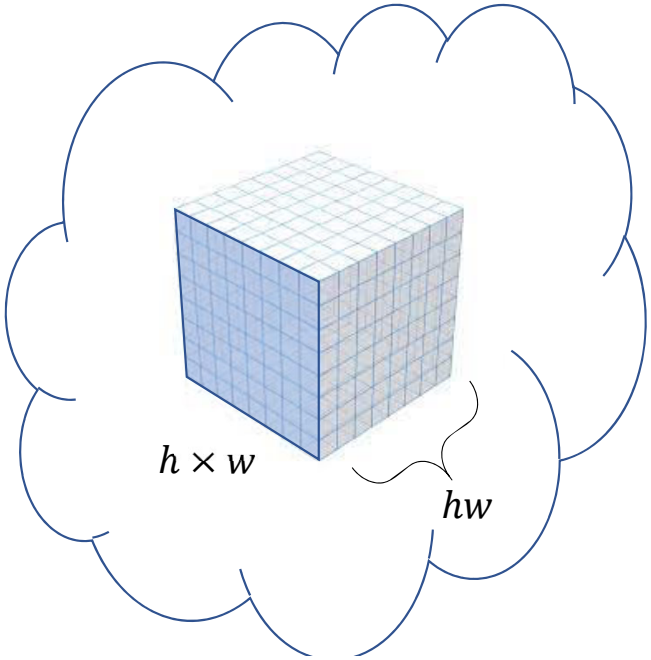
Self-Segmentation

"a **cat** is wearing sunglasses"

There is a lot of semantics in the self attention features!!!



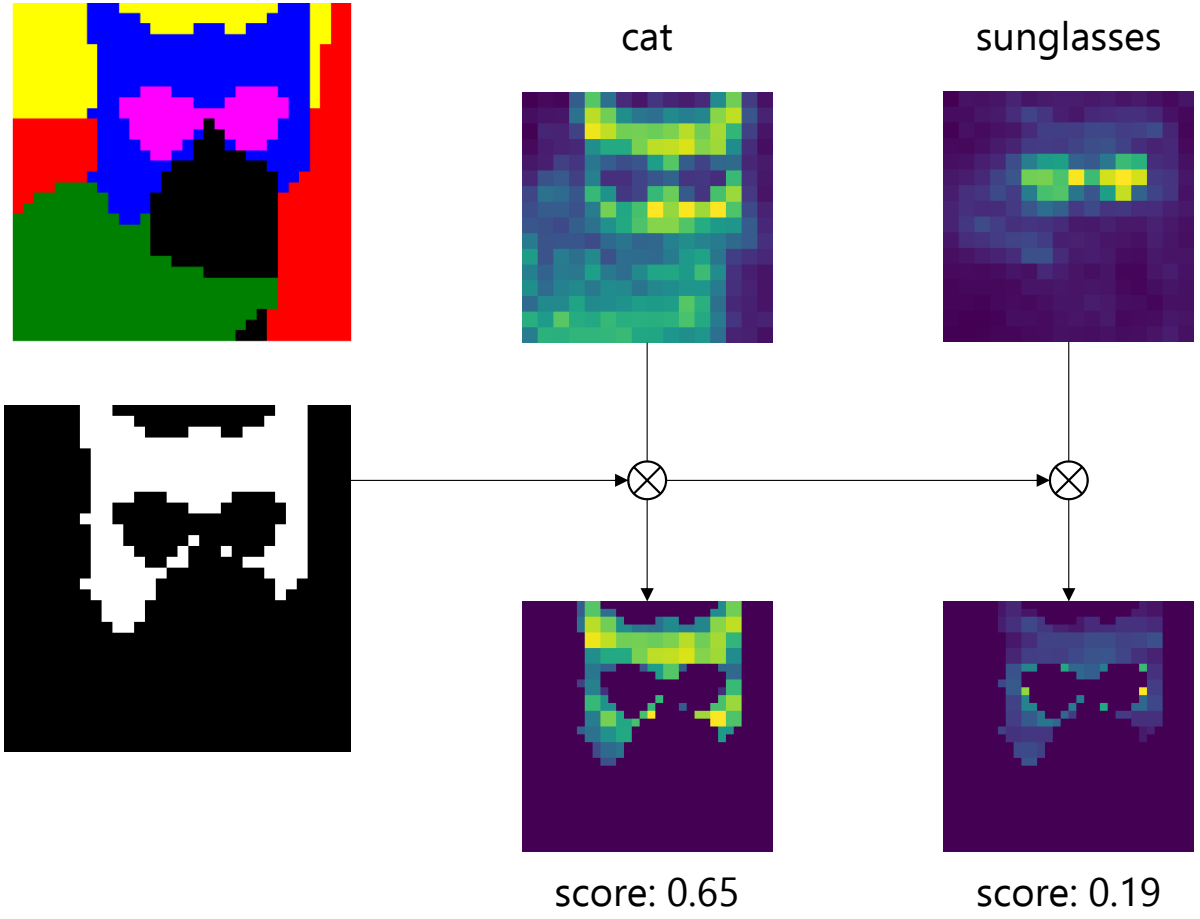
$hw \times (h \times w)$



cluster

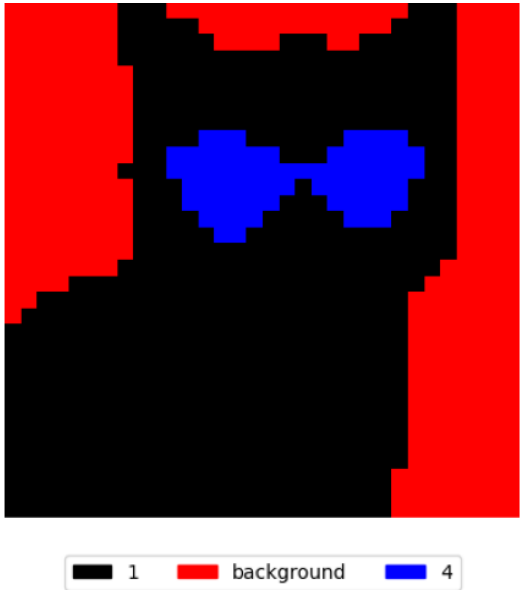
Segments labeling

“a cat is wearing sunglasses”



Segments labeling

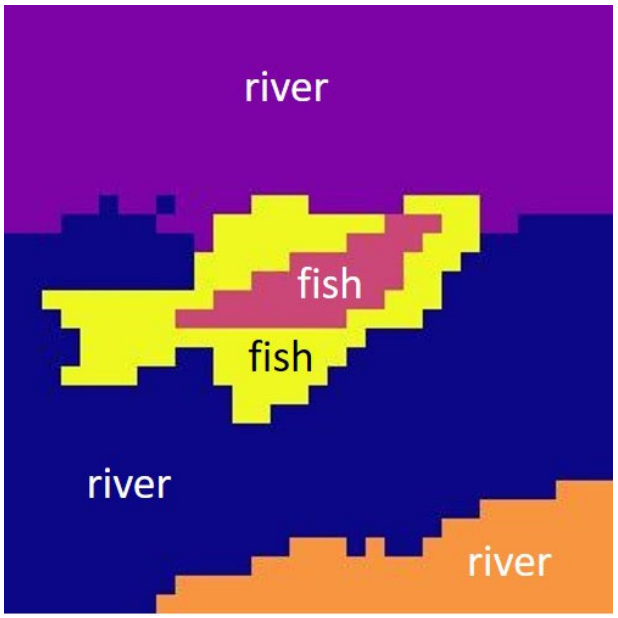
“a cat is wearing sunglasses”



1-cat, 4-sunglasses

Self-Segmentation Results

Self-Segmentation Results



Cross-Image Attention

Cross-Image Attention for Zero-Shot Appearance Transfer

Yuval Alaluf*, Daniel Garibi*, Or Patashnik, Hadar Averbuch Elor, Daniel Cohen-Or

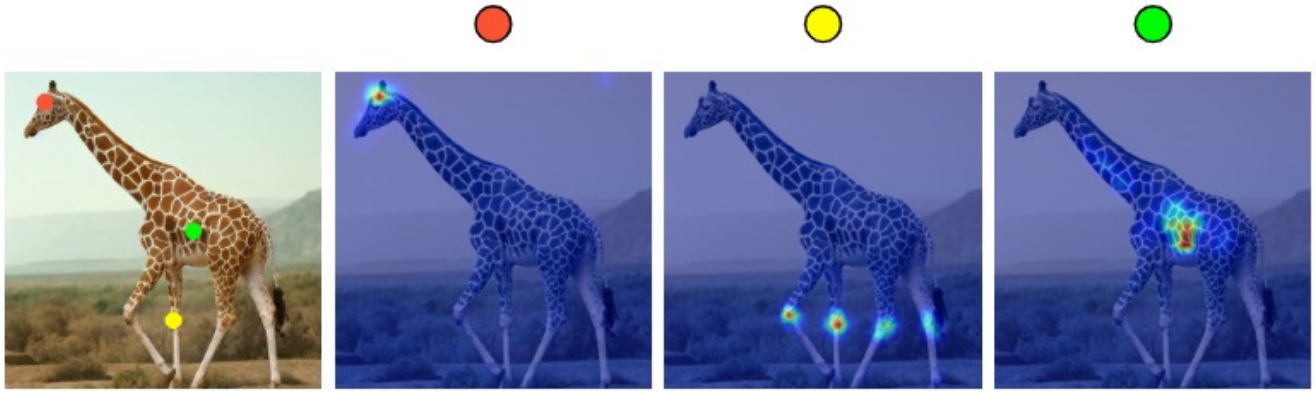
Motivation

Structure

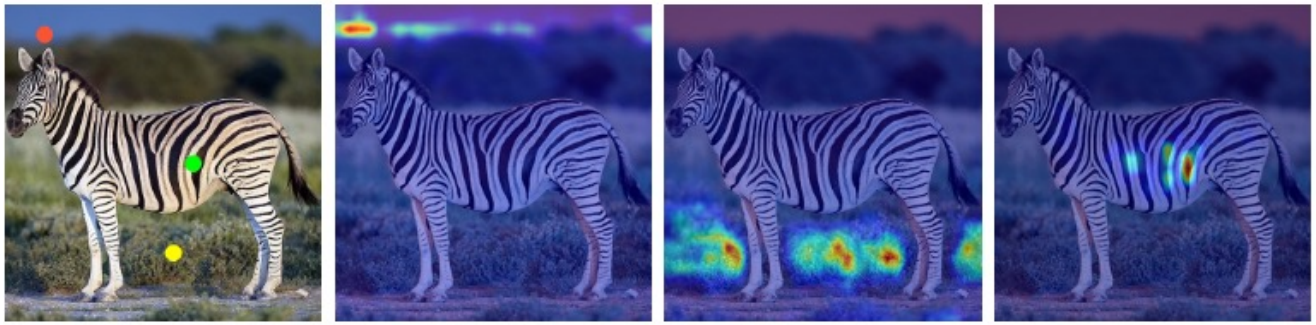
Appearance

Output

The Roles of the Queries, Keys, and Values



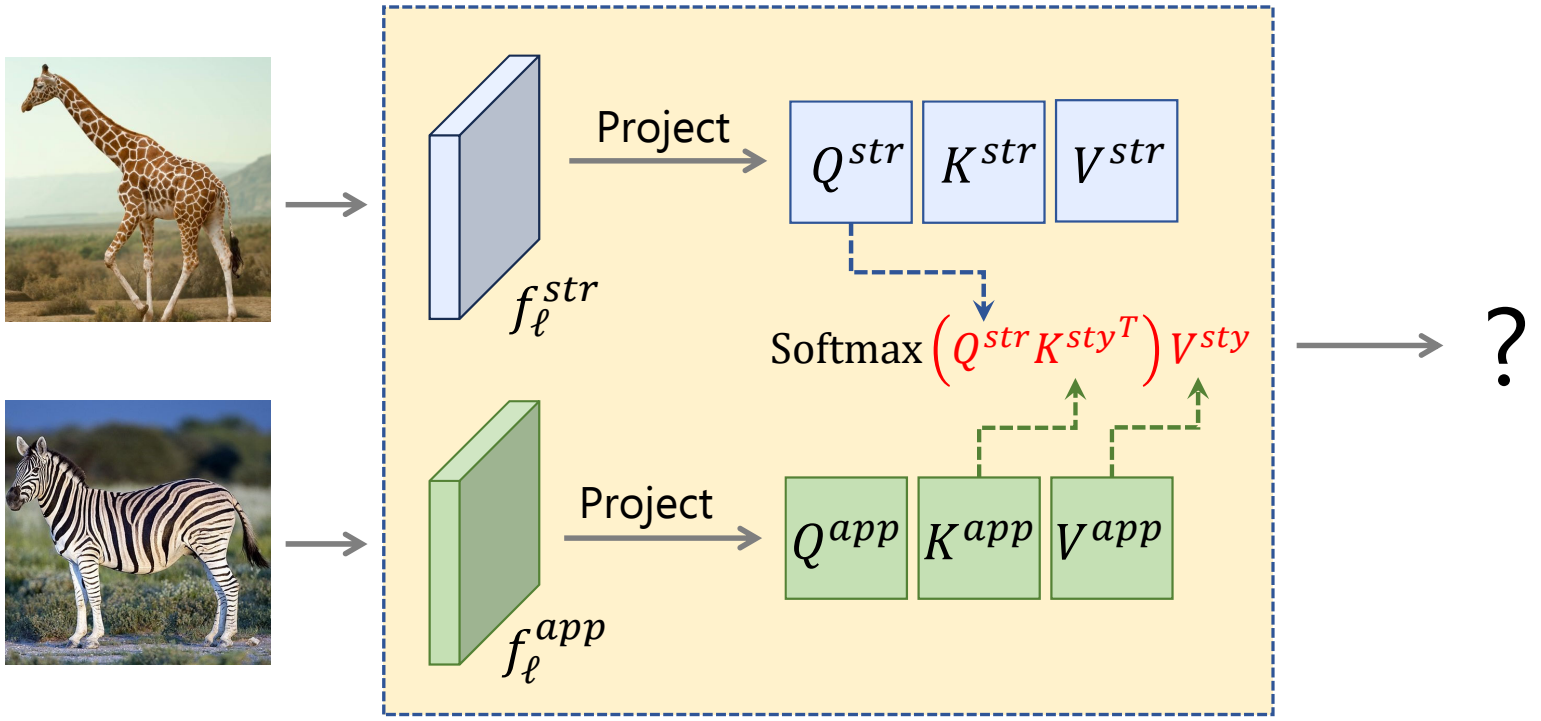
$$Q_{struct} \cdot K_{struct}^T$$



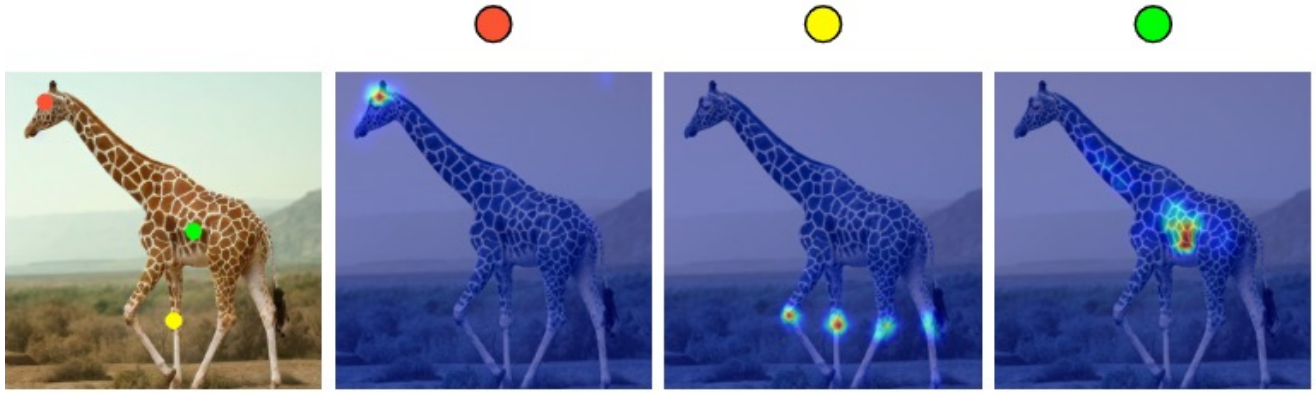
$$Q_{app} \cdot K_{app}^T$$

Self-attention maps, which focus on semantically similar regions in the image.

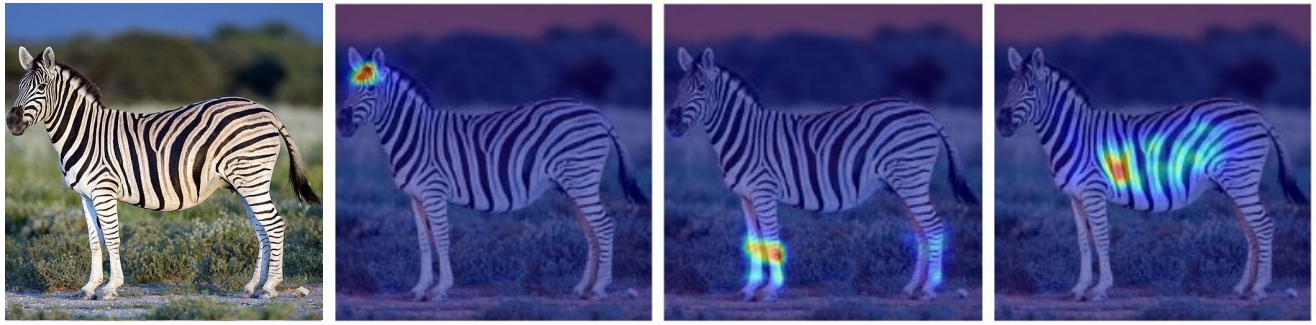
What If we Swapped the Queries, Keys, and Values Between Different Images?



The Roles of the Queries, Keys, and Values



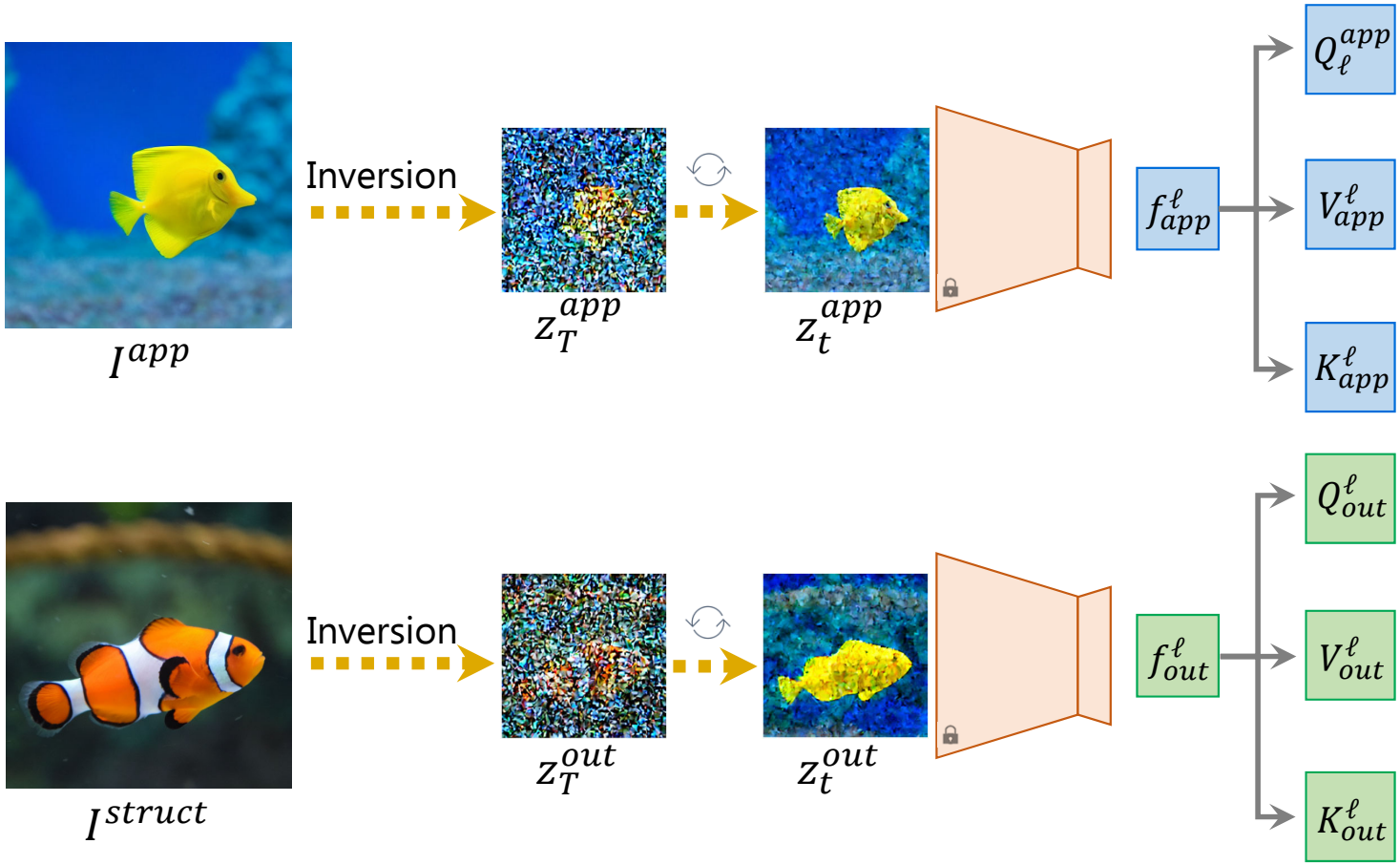
$$Q_{struct} \cdot K_{struct}^T$$



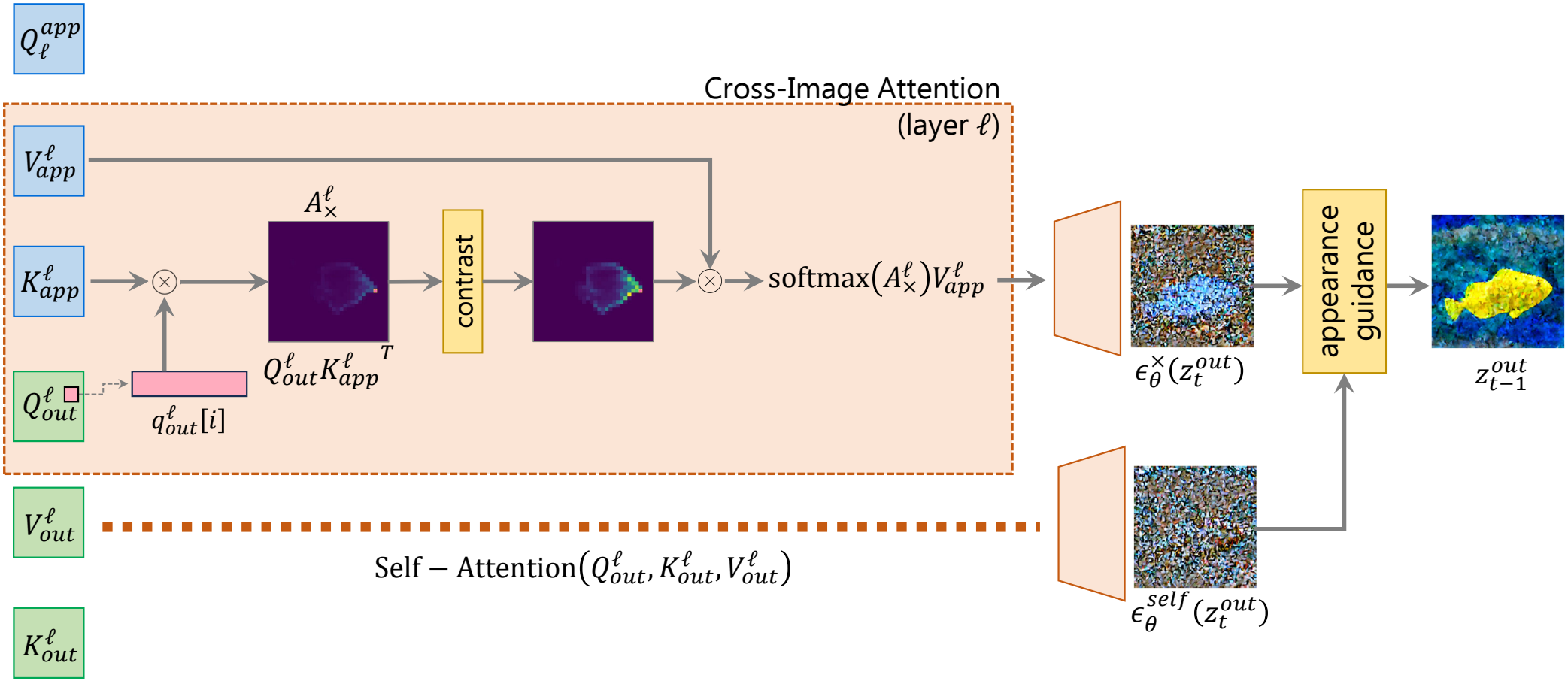
$$Q_{struct} \cdot K_{app}^T$$

Taking the **queries** from the structure image and the **keys** from the appearance image gives semantic correspondences between objects!

The Cross-Image Attention



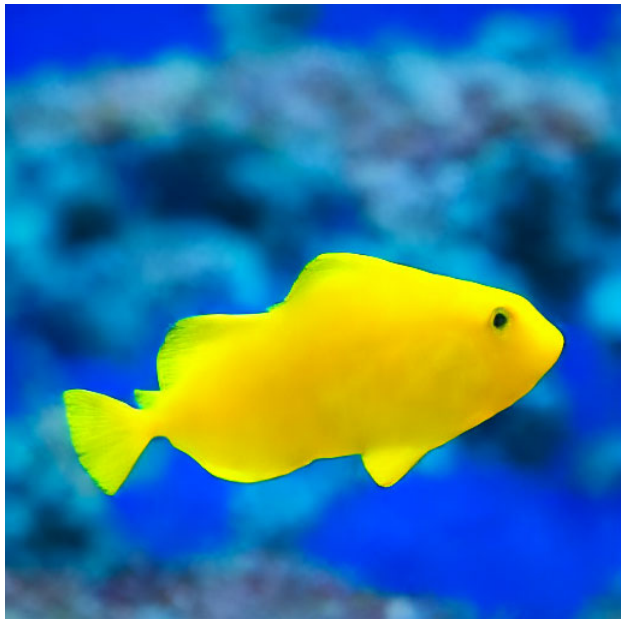
The Cross-Image Attention



The Cross-Image Attention

Structure

Appearance



Output

Appearance Transfer Results

Structure

Appearance

Output

Appearance Transfer Results

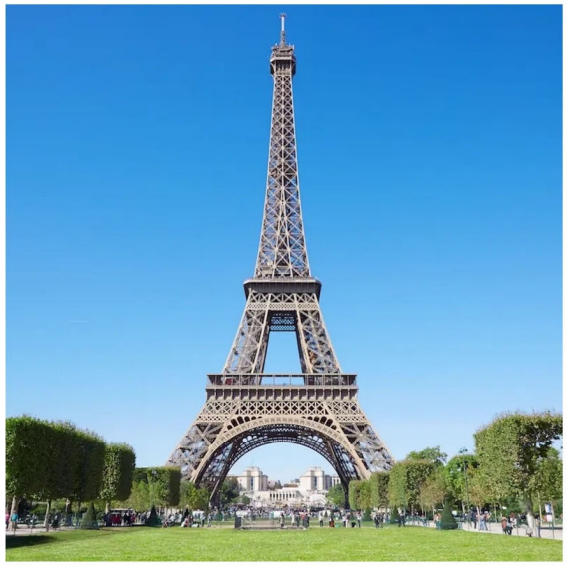
Structure

Appearance

Output

Appearance Transfer Results

Eiffel Tower



Structure

Sagrada Família

Appearance

Output

Appearance Transfer Results

Structure

Appearance

Output

Appearance Transfer Results

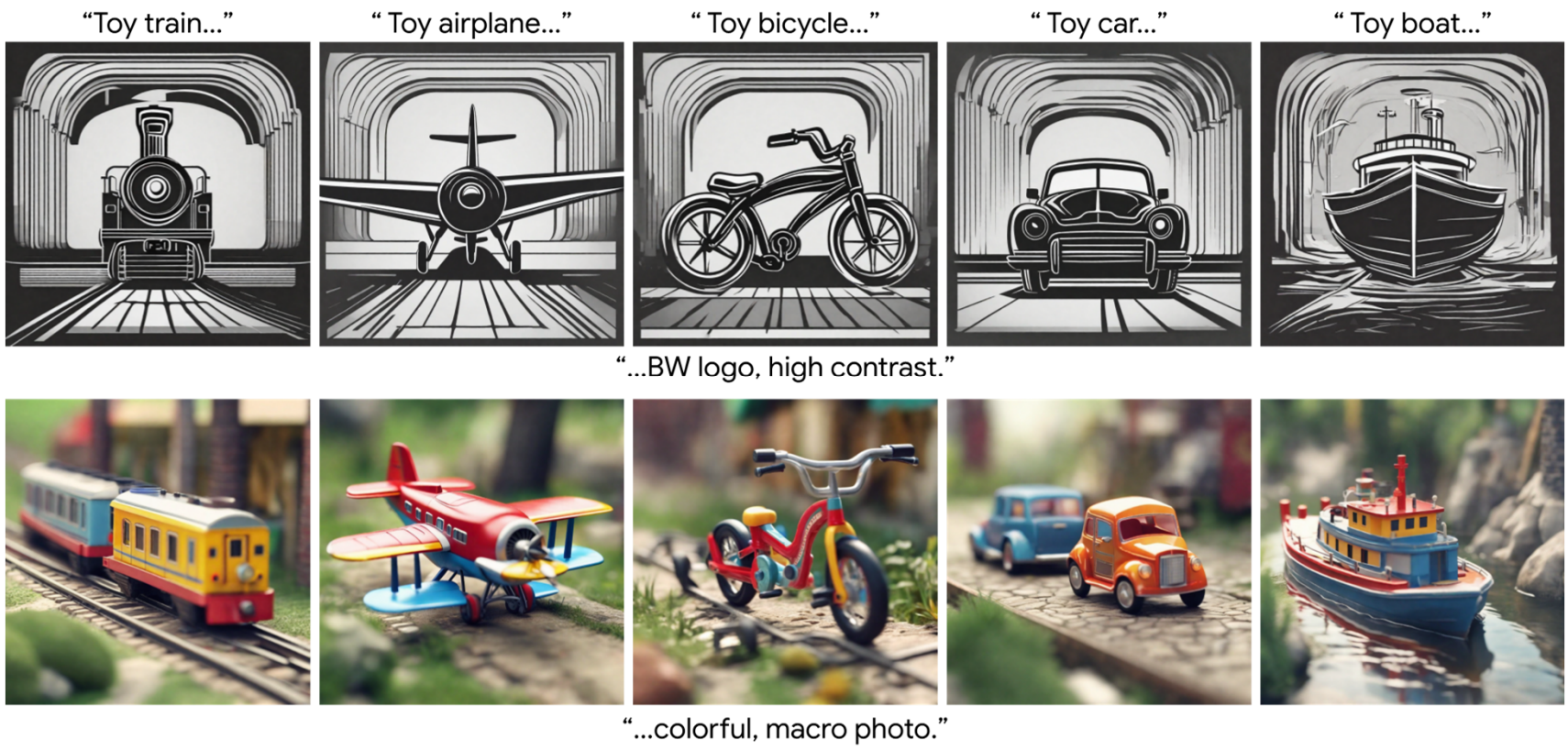
Structure

Appearance

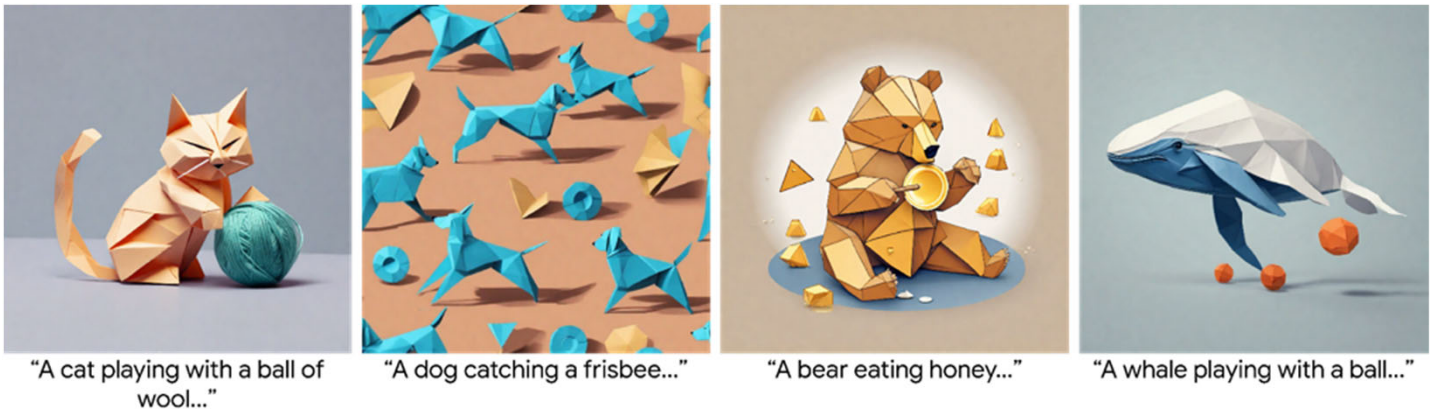
Output

StyleAlign

(link)

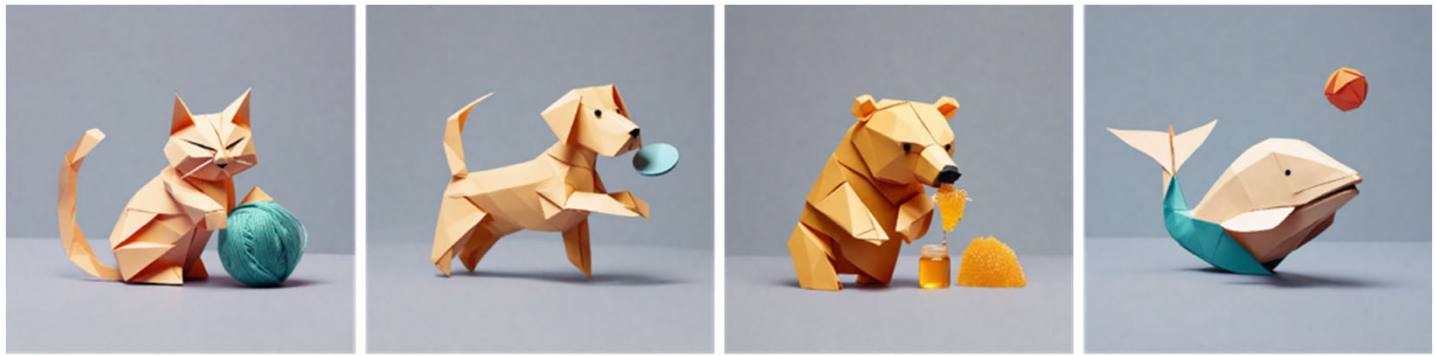


Text-to-Image Generation



"... in minimal origami style."

Text-to-Image Generation with StyleAligned

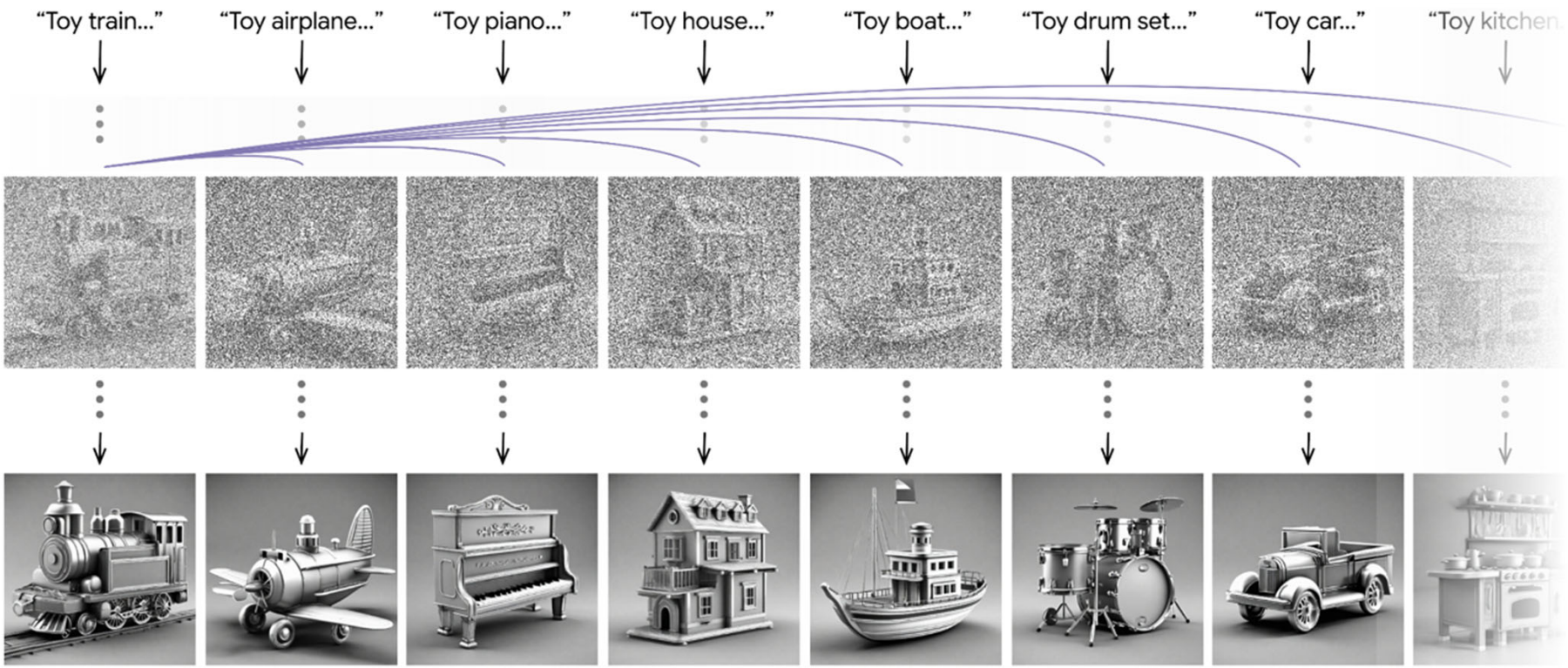


"A cat playing with a ball of wool..." | "A dog catching a frisbee..." | "A bear eating honey..." | "A whale playing with a ball..."

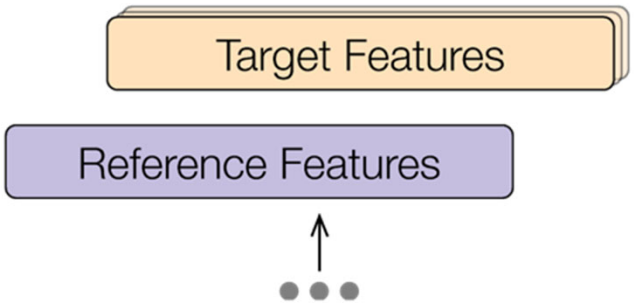
"A woman working in the office..." | "A temple..." | "A person riding a bike..." | "A cactus..."

"... in minimal origami style."

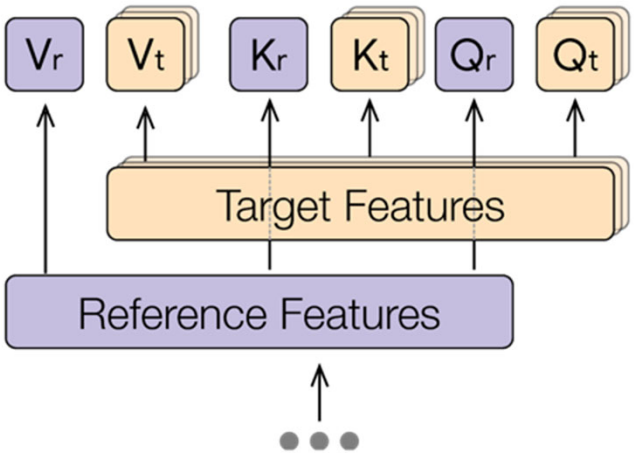
Shared attention during the diffusion process



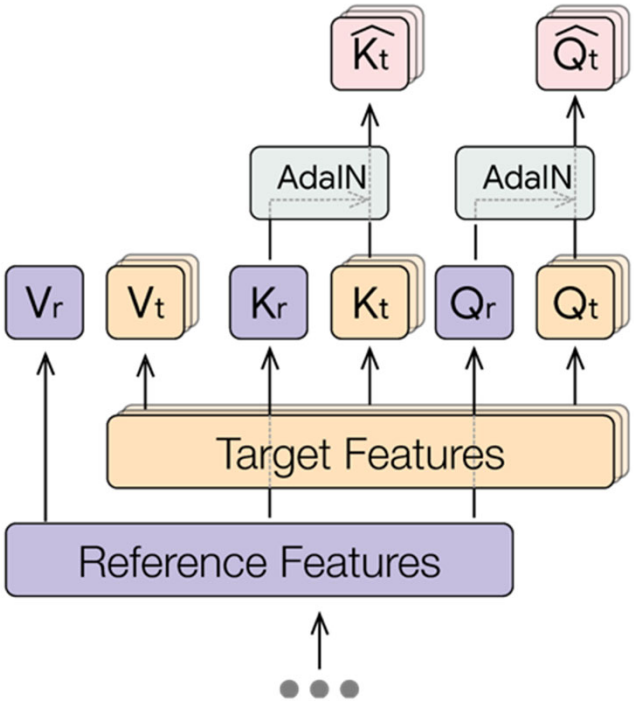
Shared Attention Layer



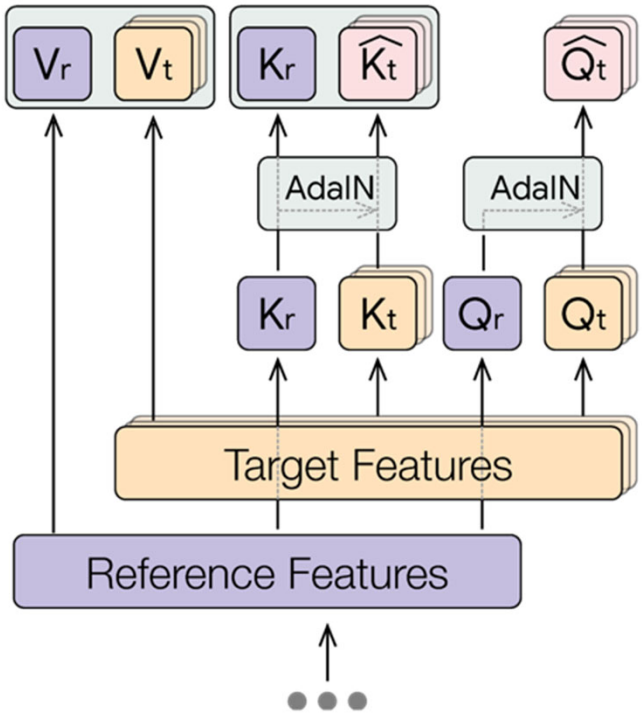
Shared Attention Layer



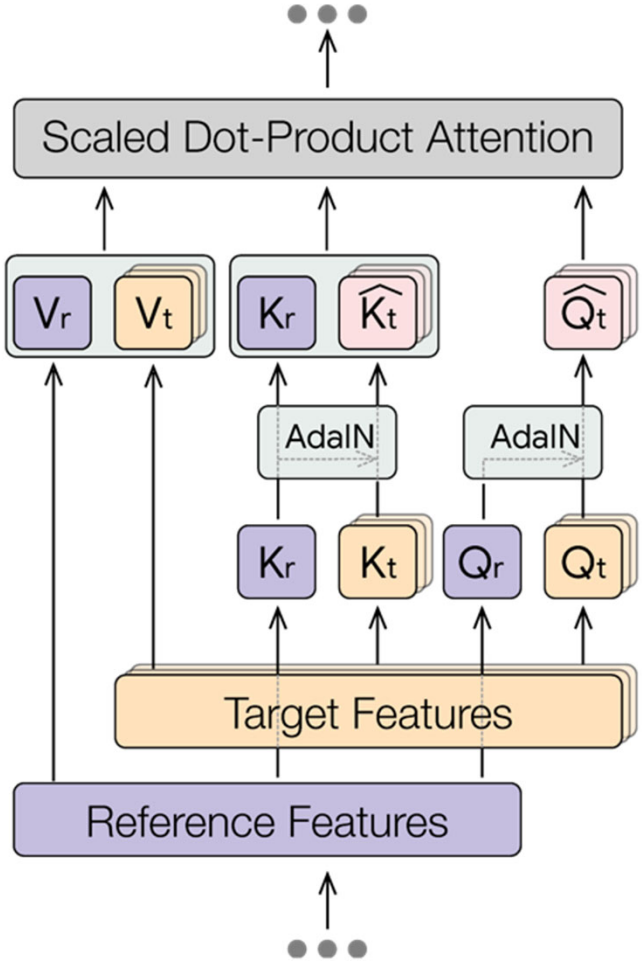
Shared Attention Layer



Shared Attention Layer

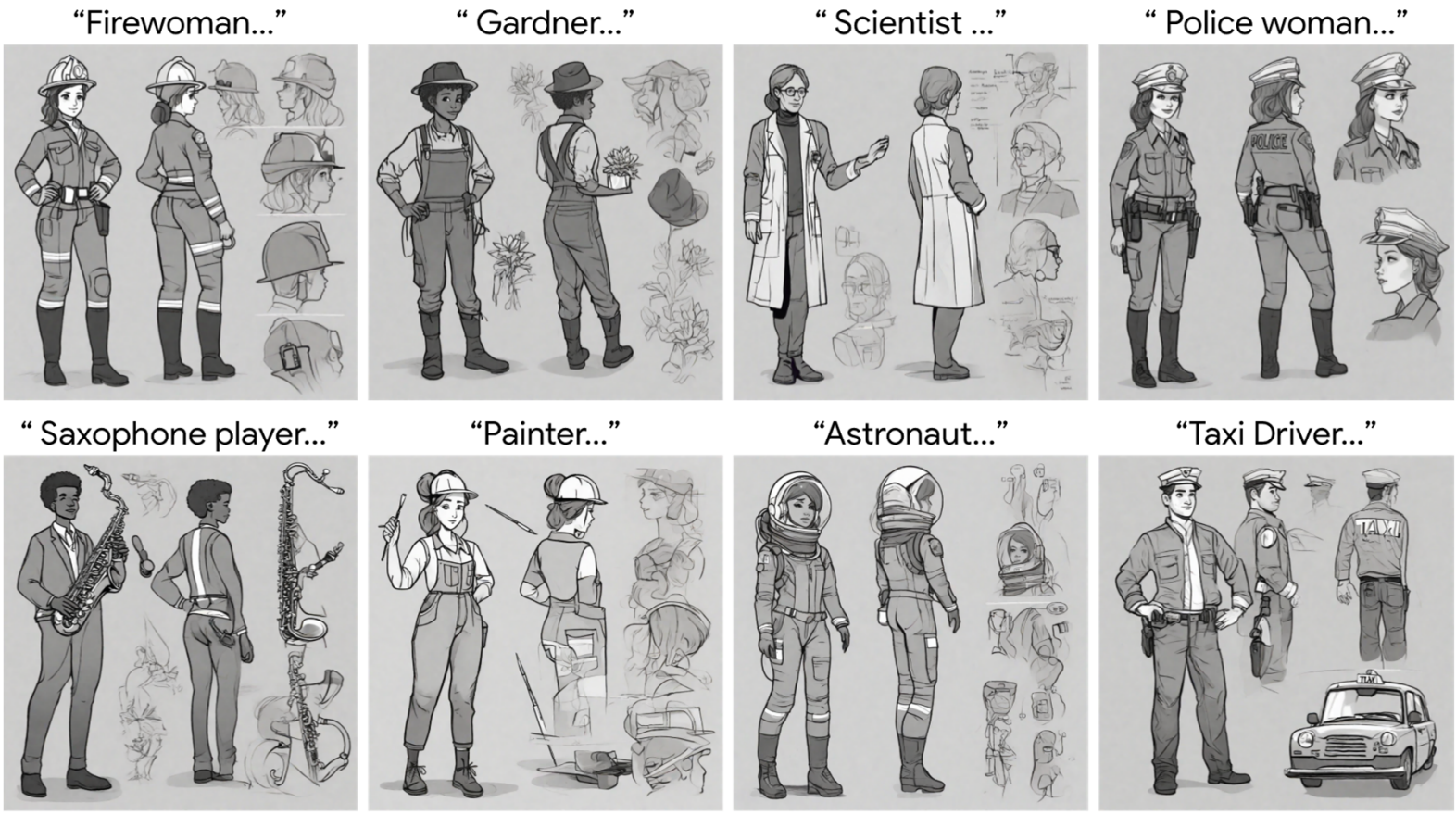


Shared Attention Layer



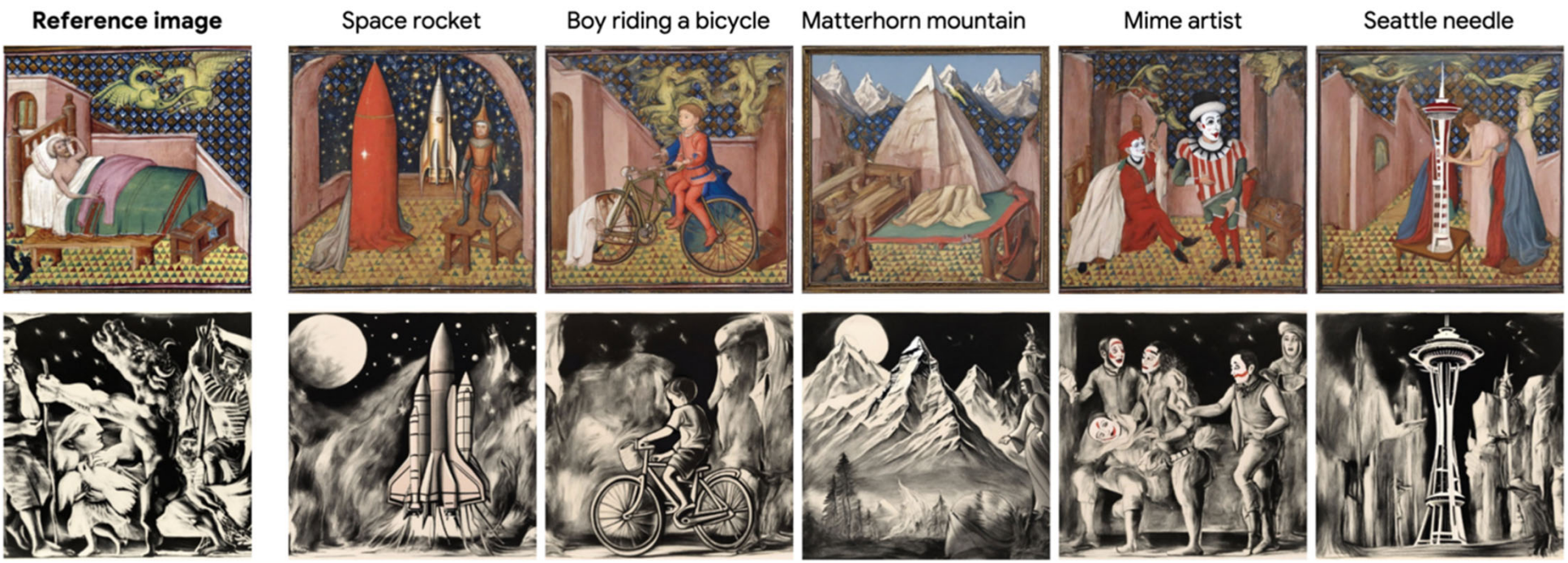
Style Aligned generation of Synthetic Images

Style Aligned generation of Synthetic Images



Style Aligned generation of Synthetic Images

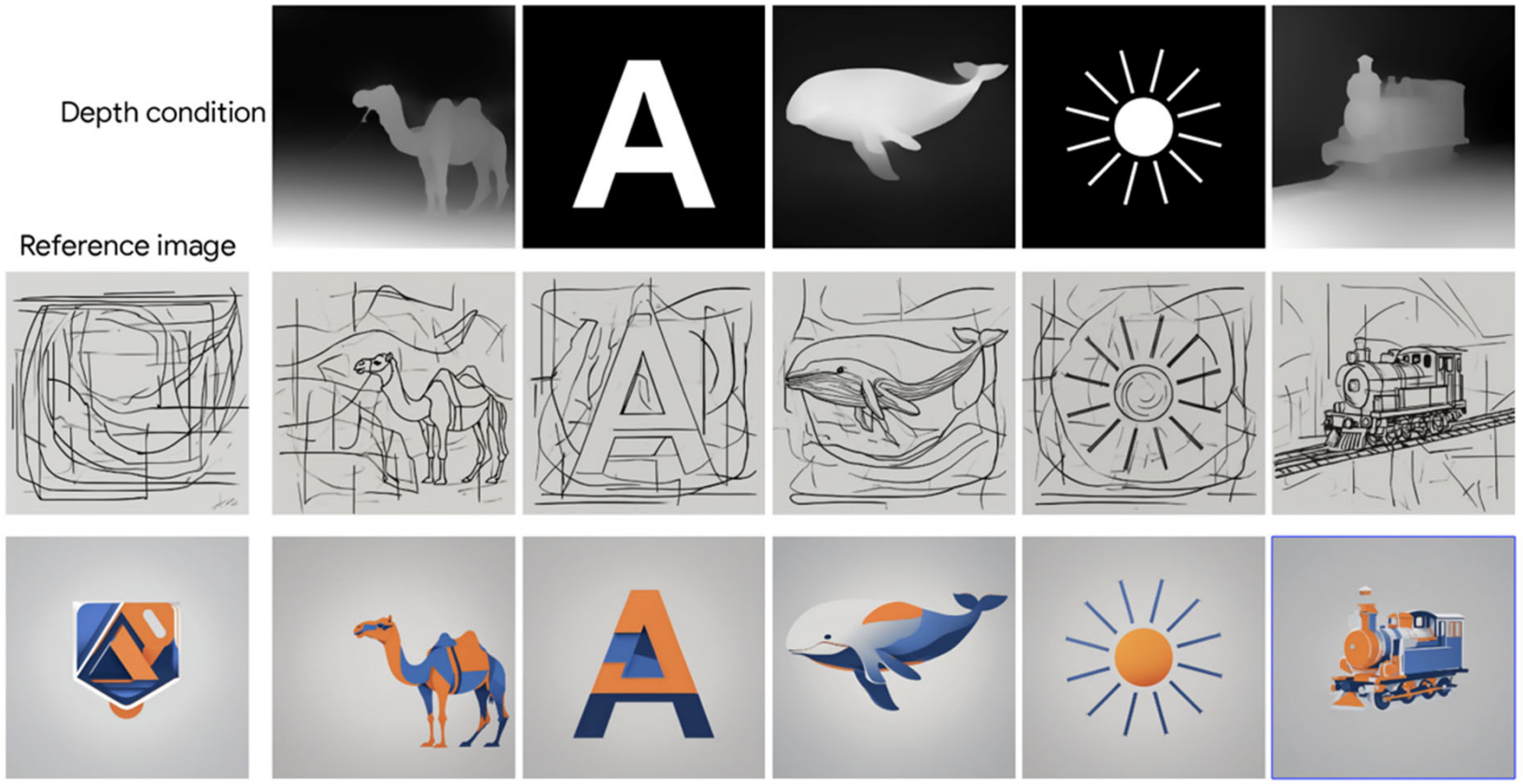
Style Aligned generation from an Input Image



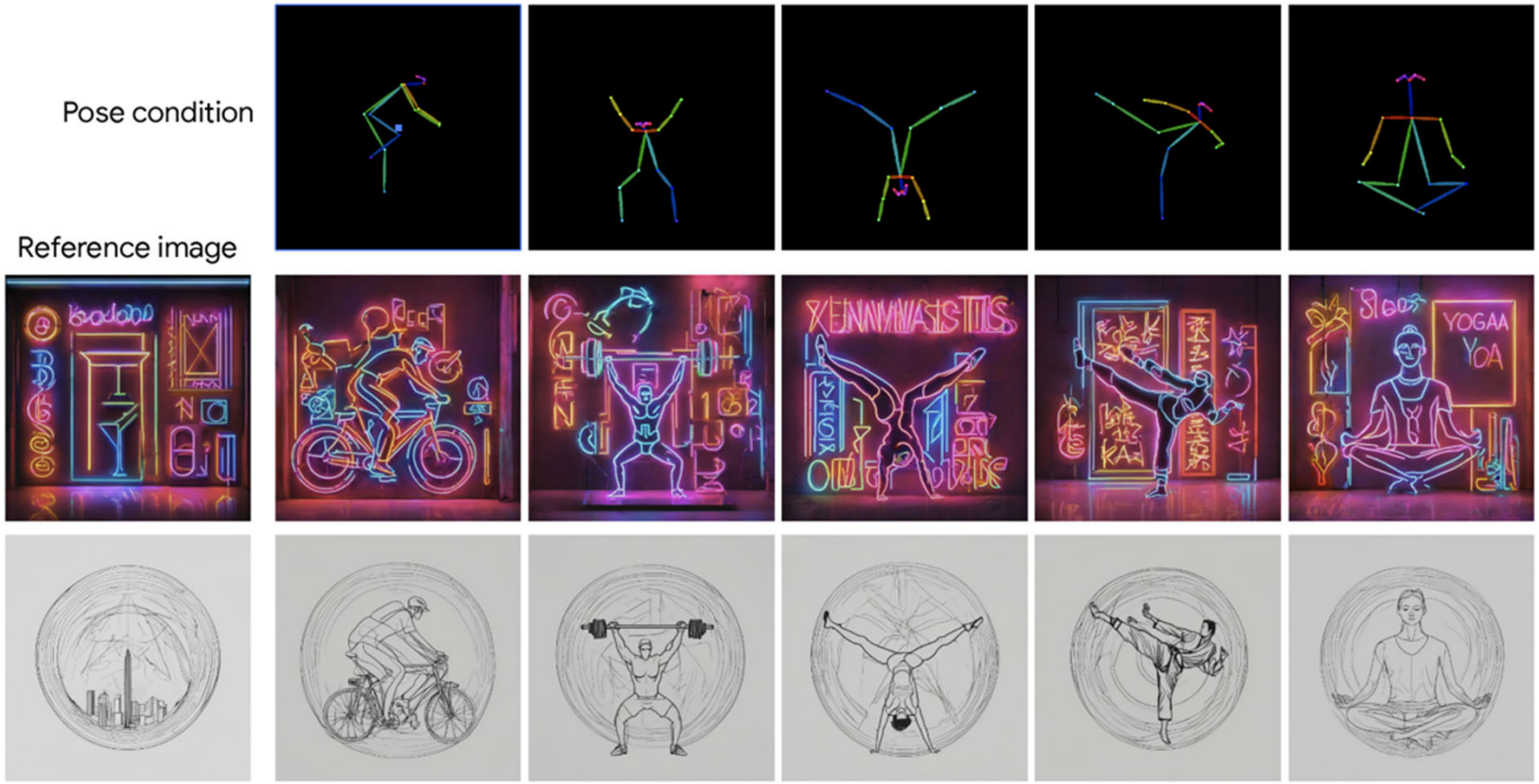
Style Aligned generation from an Input Image

StyleAligned with other methods

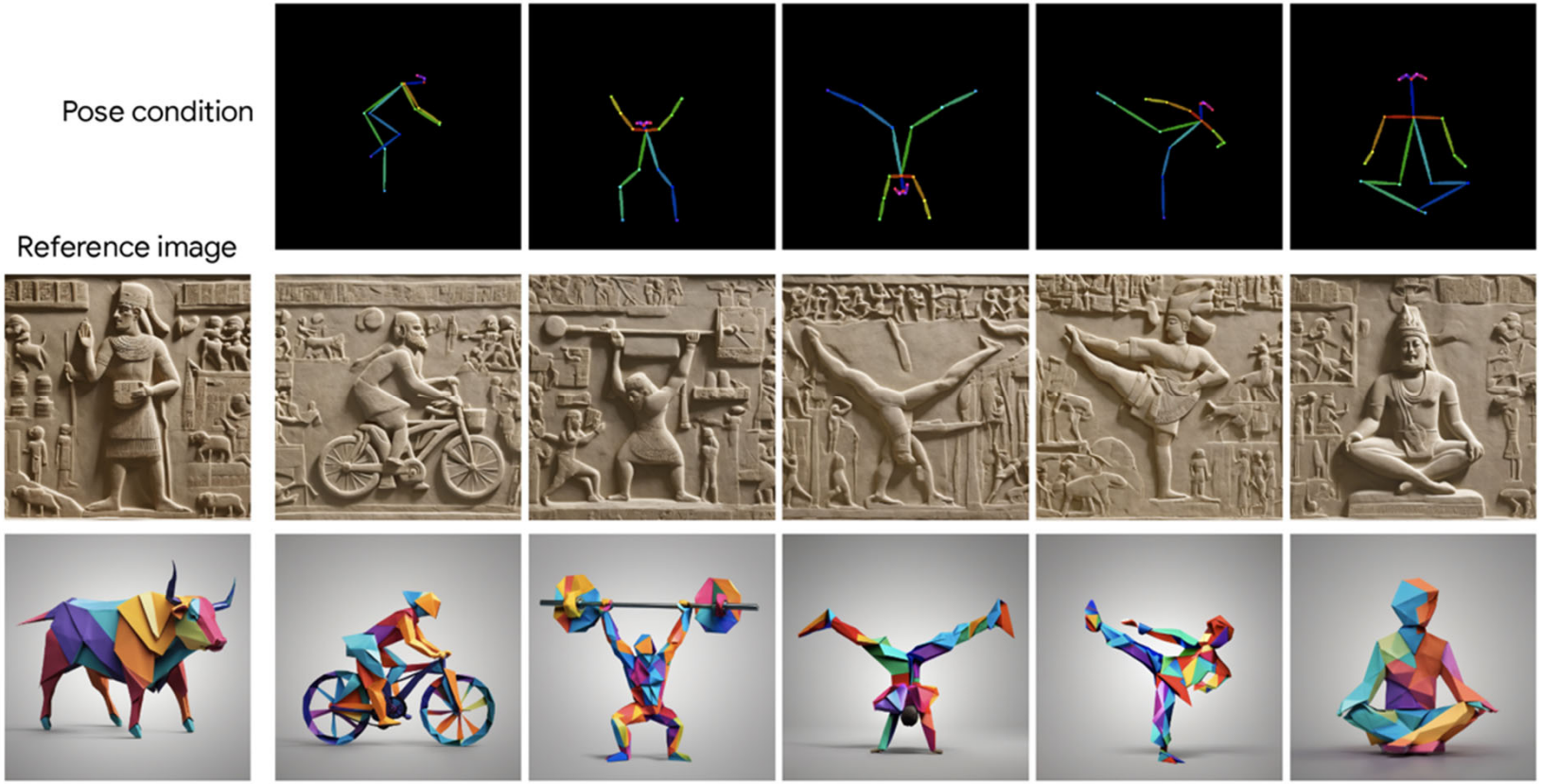
ControlNet + StyleAligned



ControlNet + StyleAligned

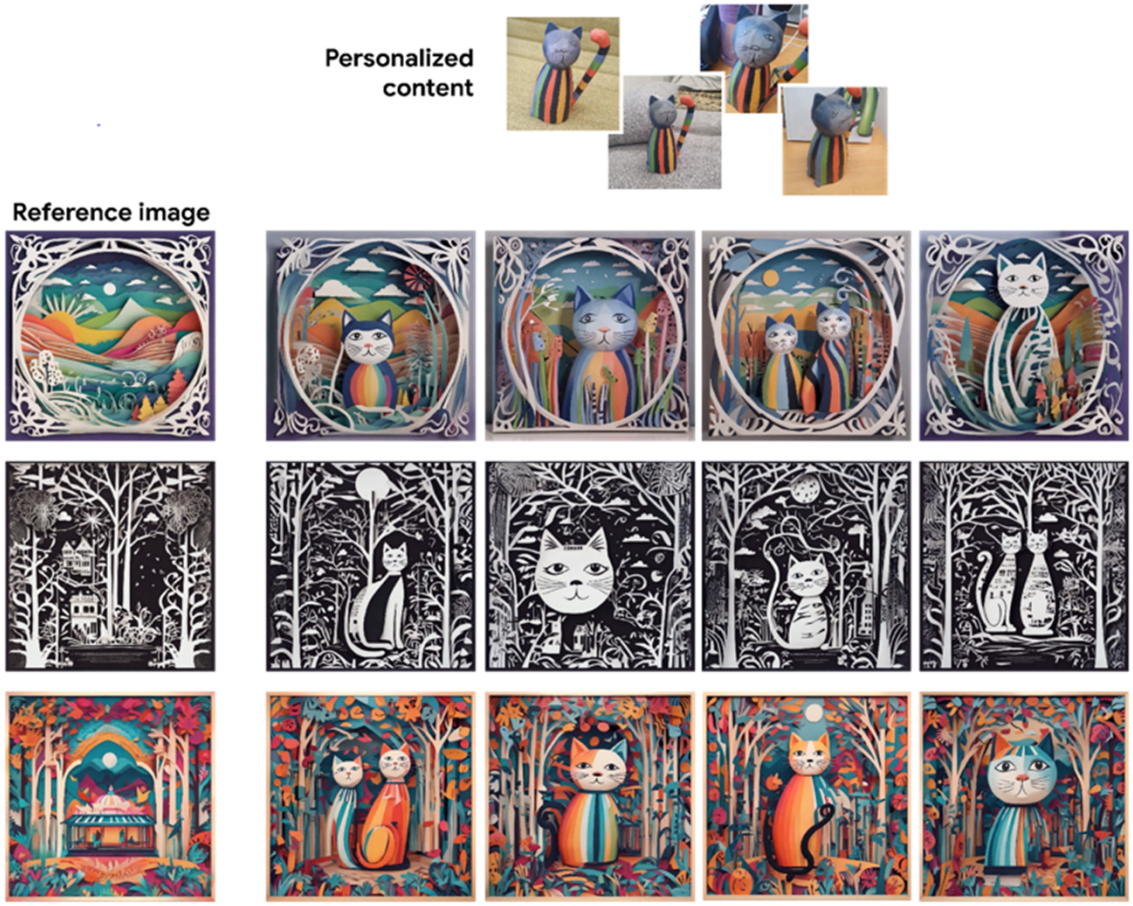


ControlNet + StyleAligned



Textual Inversion+Dreambooth

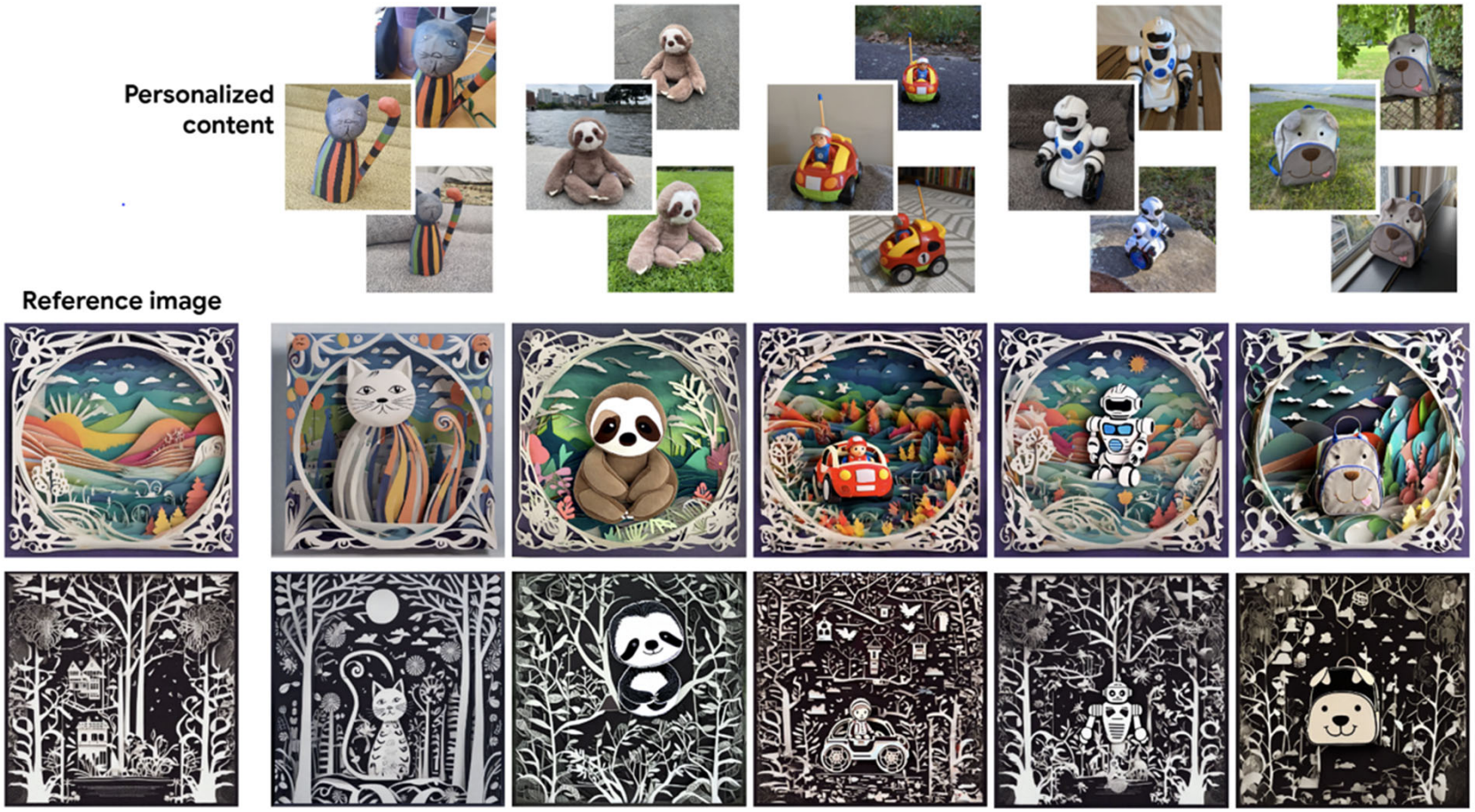
+ StyleAligned



W.O AdaIN


Personalized content
Reference image
“<V object> in the style of a beautiful paper-cut art.”

DreamBooth + StyleAligned

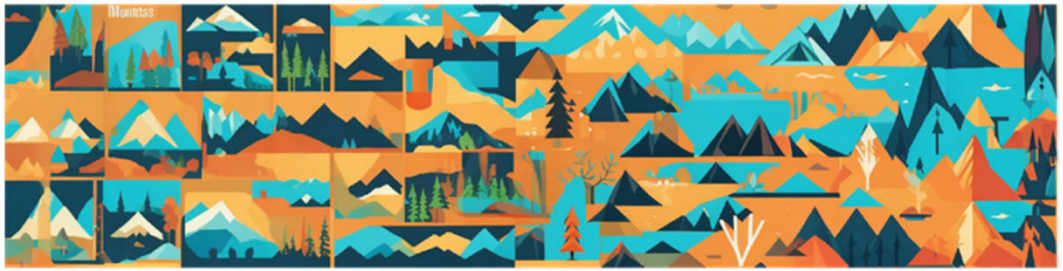


MultiDiffusion + StyleAligned

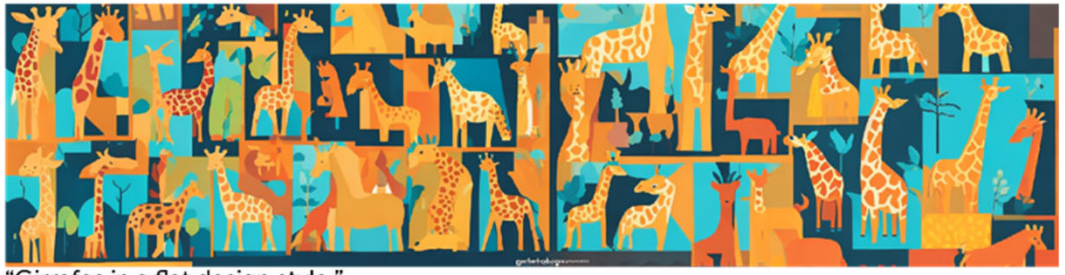
Reference image

"A poster in a flat design style."

"Houses in a flat design style."



"Mountains in a flat design style."



"Girrafes in a flat design style."

MultiDiffusion + StyleAligned

Reference image

"A poster in a flat design style."

W,O shared attention

W,O Attnetion AdaIN

StyleAligned full

MultiDiffusion in Multi Styles

Left Reference

Right Reference

MultiDiffusion in Multi Styles

Left reference

Right reference

Left Reference

Right Reference

Diffusion Models for Visual Computing

Paul Guerrero

Niloy Mitra, Daniel Cohen-Or, Minhyuk Sung, Chun-Hao Huang, Duygu Ceylan

Part 4: Personalization & Editing

https://geometry.cs.ucl.ac.uk/courses/diffusion4VC_eg24/

Presentation Schedule

Introduction to Diffusion Models

Guidance and Conditioning Sampling

Attention

Break

Personalization and Editing

Beyond Single Images

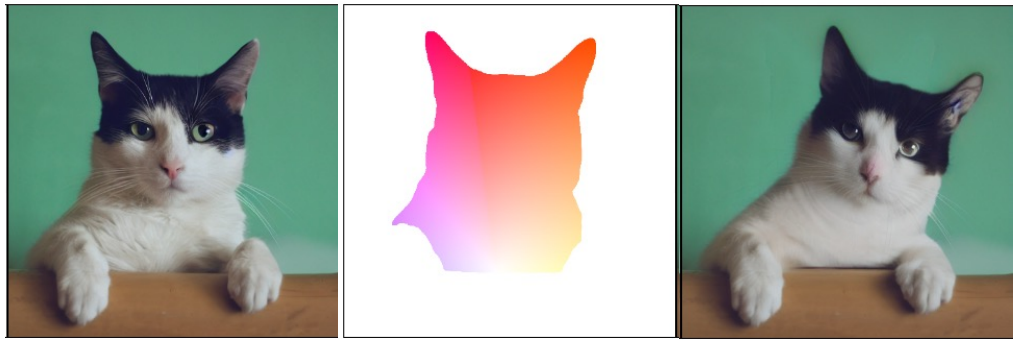
Diffusion Models for 3D Generation

Personalization

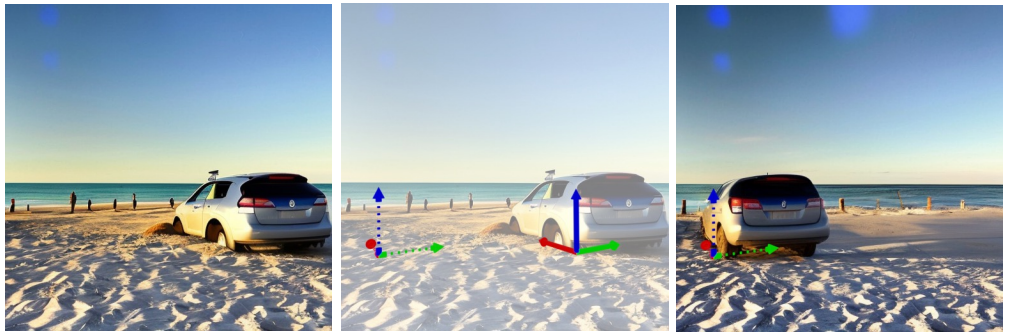
Input images

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Ruiz et al., CVPR 2023

Editing



Motion Guidance: Diffusion-Based Image Editing with Differentiable Motion Estimators, Geng and Owens, ICLR 2024



Diffusion Handles: Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D Pandey et al., CVPR 2024

Personalization

"a hyper-realistic digital painting of a happy girl, with brown eyes"

Without Personalization

Generated with StabelDiffusion 2.1

With Personalization

ConsiStory: Training-Free Consistent Text-to-Image Generation
Tewel et al., ArXiv Feb. 2024

Personalization

With Personalization

Same subject in different settings.

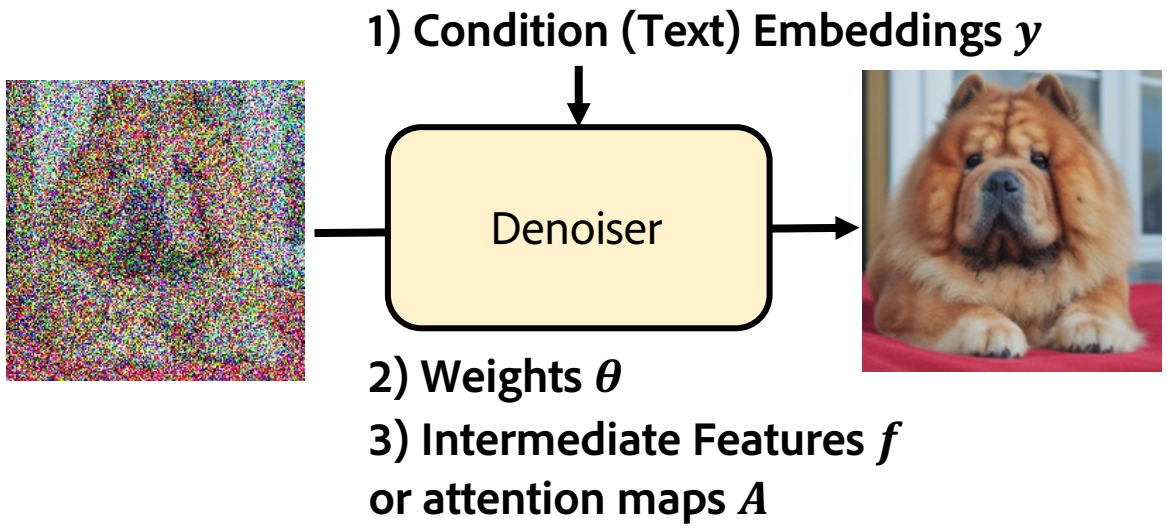
Personalization:

Generative Model
+ Identity Preservation

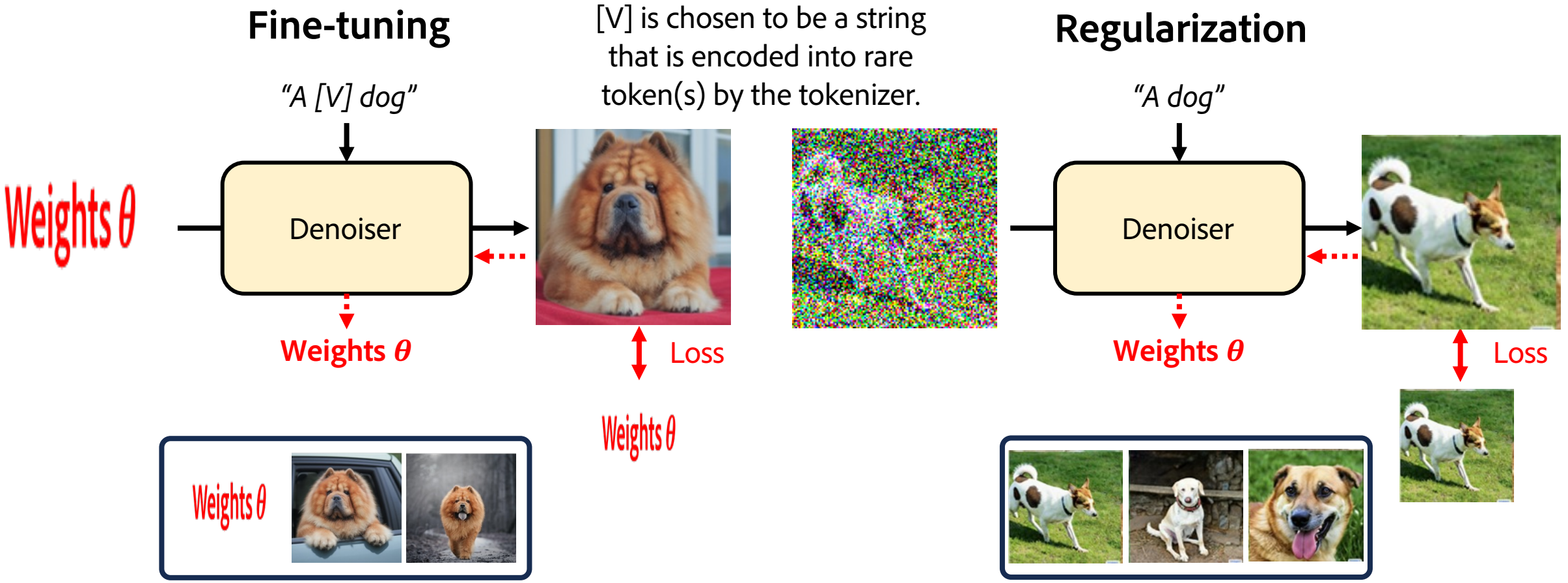
ConsiStory: Training-Free Consistent Text-to-Image Generation
Tewel et al., ArXiv Feb. 2024

Identity Preservation

How can we represent the identity of a subject?



ID Preservation by Fine-Tuning Denoiser Weights



3-5 example images showing the identity.

Regularization data can be generated by the original, non-finetuned model, or can come from a large dataset.

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Ruiz et al., CVPR 2023

ID Preservation by Fine-Tuning Denoiser Weights

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation, Ruiz et al., CVPR 2023

ID Preservation by Fine-Tuning Denoiser Weights

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation,
Ruiz et al., CVPR 2023

Code

https://github.com/google/dreambooth

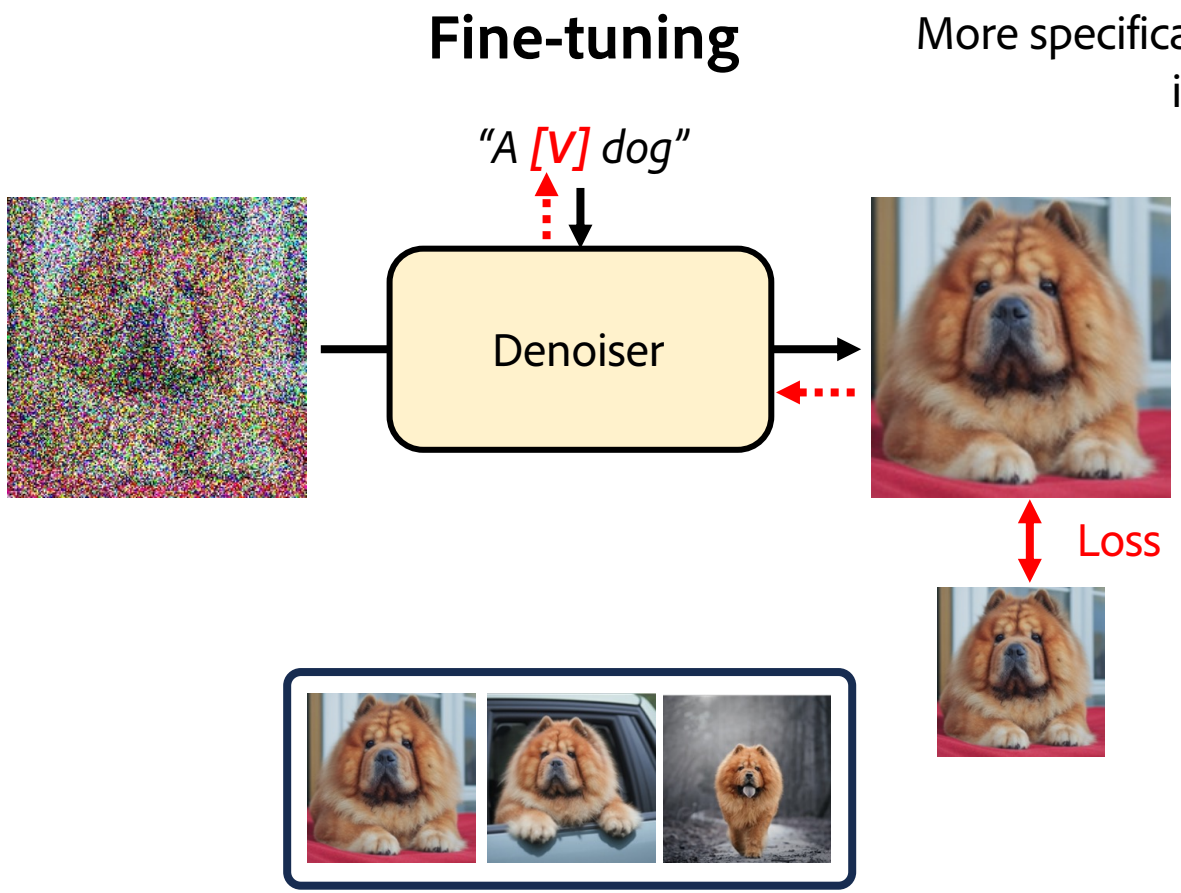
Code

https://huggingface.co/docs/diffusers/en/training/dreambooth

Demo

https://huggingface.co/spaces/multimodalart/dreambooth-training

ID Preservation by Fine-Tuning Text Embeddings



An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion, Gal et al., ICLR 2023

ID Preservation by Fine-Tuning Text Embeddings

Compared to fine-tuning denoiser weights:

- Only the optimized embedding needs to be stored, not the full denoiser weights.
- ID preservation is a bit weaker.

Input Images

"A photo of [V]"

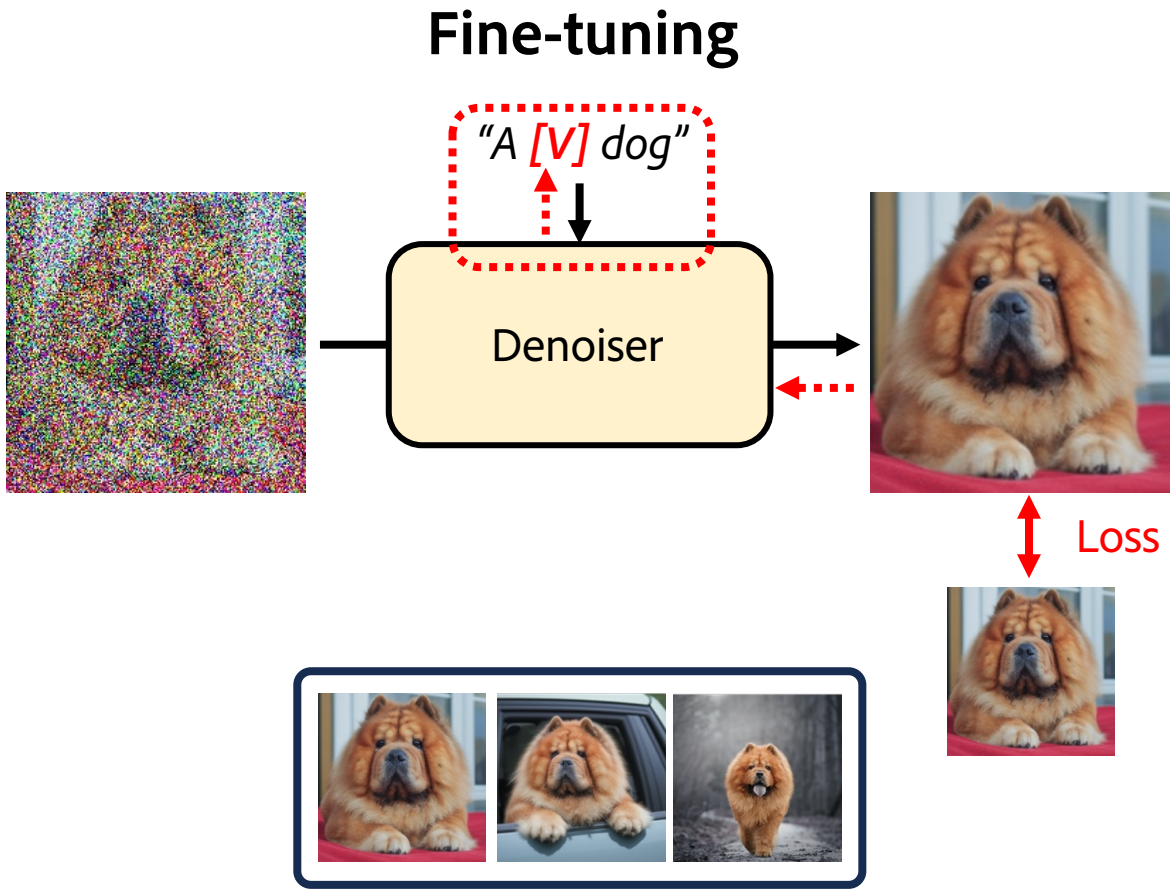
"A photo of S_*" "A photo of S_* on the beach" "A photo of S_* on the moon" "Elmo holding a S_*"

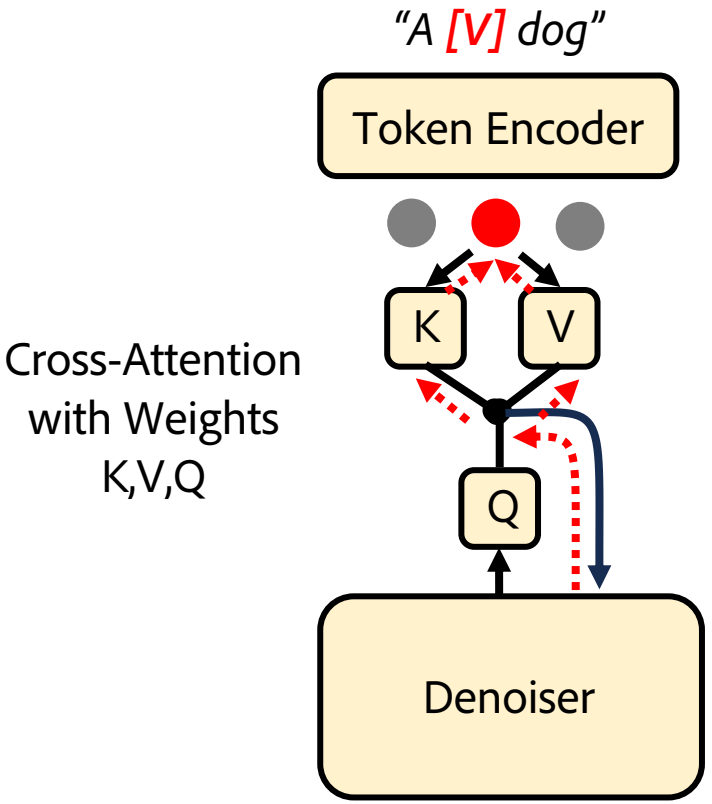
An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion, Gal et al., ICLR 2023

Code

https://github.com/rinongal/textual_inversion

Fine-Tuning Text Embeddings & Cross-Att. Weights





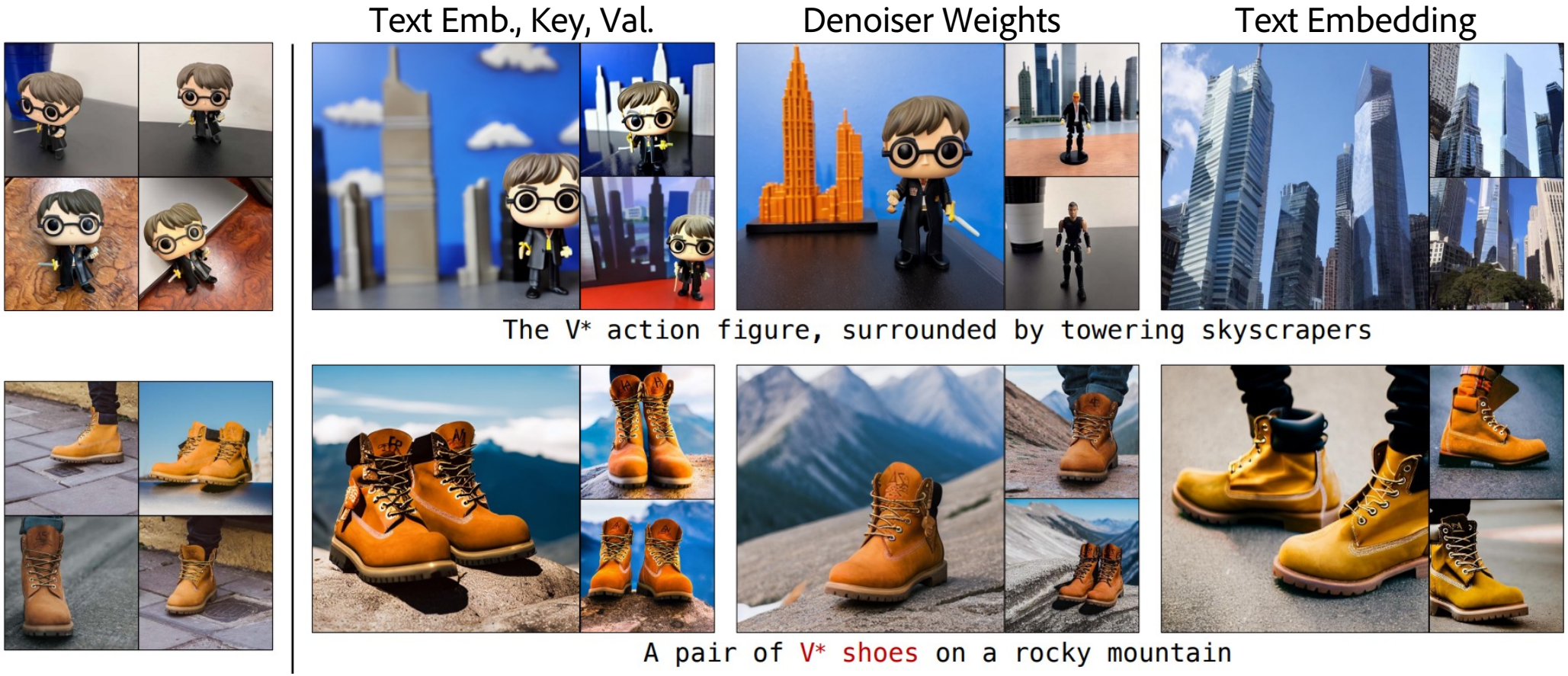
Cross-Attention with Weights K,V,Q

Multi-Concept Customization of Text-to-Image Diffusion, Kumari et al., CVPR 2023

Key-Locked Rank One Editing for Text-to-Image Personalization, Tewel et al., SIGGRAPH 2023

Fine-Tuning Text Embeddings & Cross-Att. Weights

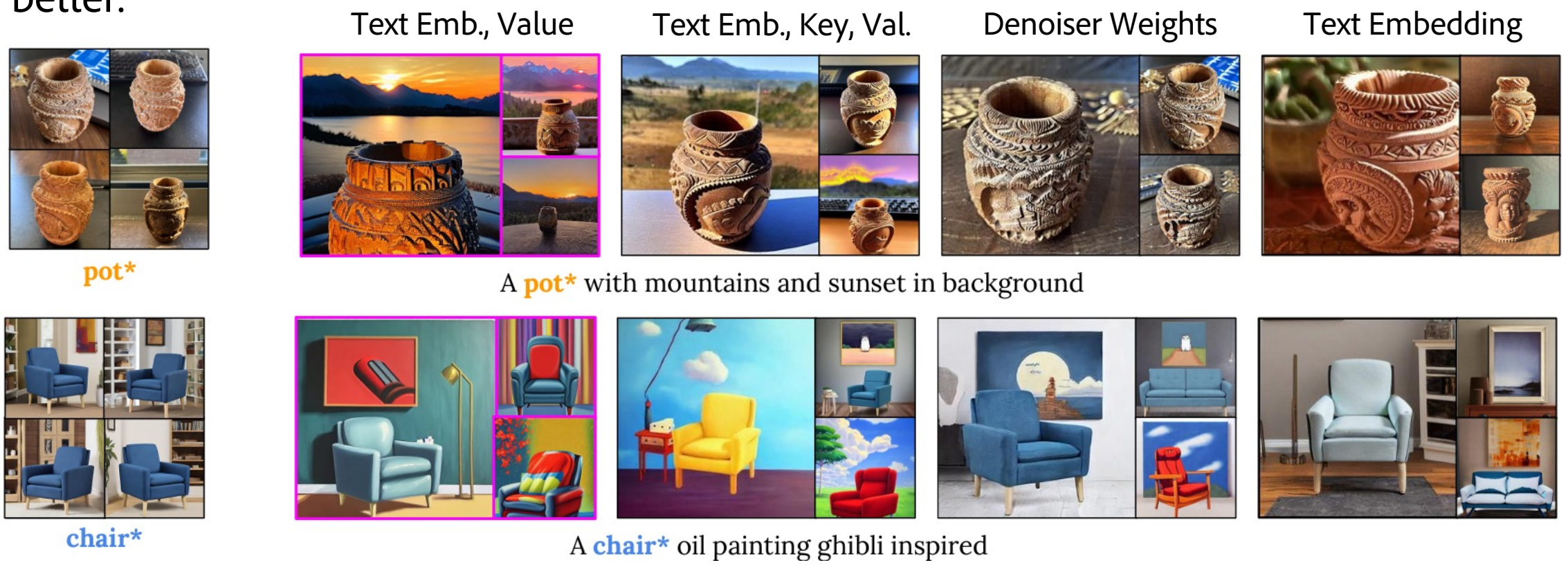
Fine-tuning text embeddings, keys and values.
ID preservation is close to tuning denoiser weights, while requiring less storage.



Multi-Concept Customization of Text-to-Image Diffusion, Kumari et al., CVPR 2023

Fine-Tuning Text Embeddings & Cross-Att. Weights

Fine-tuning text embeddings and values only.
ID preservation is slightly worse than tuning denoiser weights but follows the prompt better.



Key-Locked Rank One Editing for Text-to-Image Personalization, Tewel et al., SIGGRAPH 2023

Custom Diffusion

Multi-Concept Customization of Text-to-Image Diffusion

(fine-tuning keys, values and text embeddings)

Code

https://github.com/adobe-research/custom-diffusion

Code

https://huggingface.co/docs/diffusers/en/training/custom_diffusion

Perfusion

Key-Locked Rank One Editing for Text-to-Image Personalization

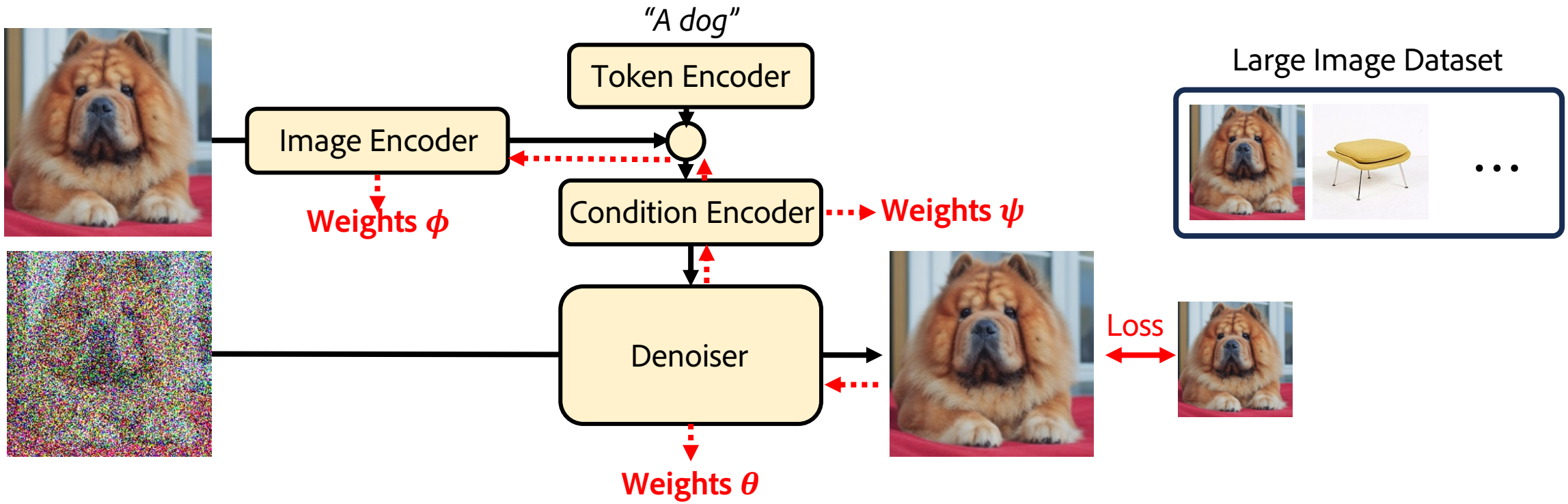
(fine-tuning values and text embeddings only)

Code

https://github.com/ChenDarYen/Key-Locked-Rank-One-Editing-for-Text-to-Image-Personalization

ID Preservation with Learned Condition Encoders

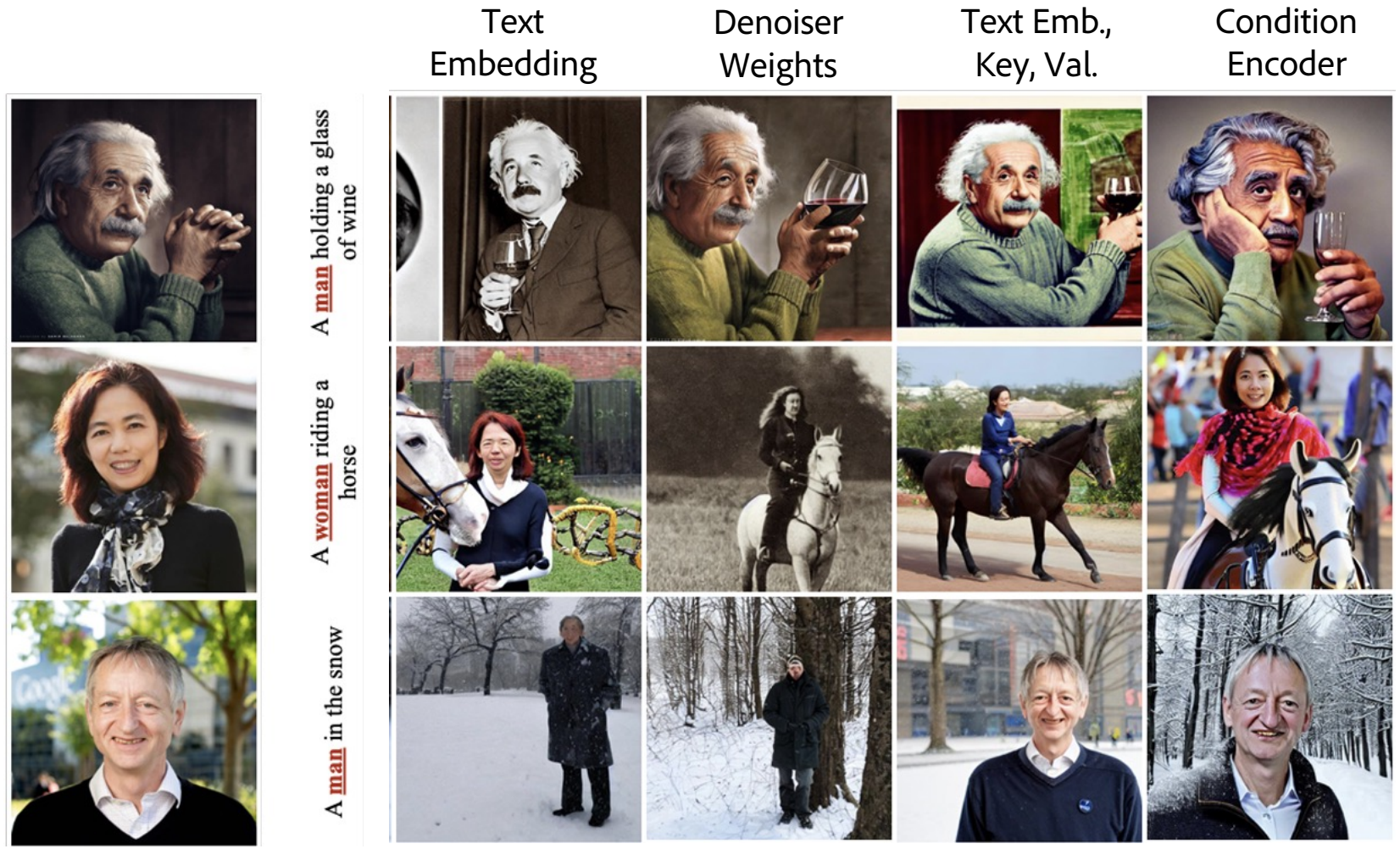
Motivation: avoid the need to fine-tune for each object identity.



FastComposer: Tuning-Free Multi-Subject Image Generation with Localized Attention, Gal et al., ArXiv May 2023

BLIP-Diffusion: Pre-trained Subject Representation for Controllable Text-to-Image Generation and Editing, Li et al., NeurIPS 2024

ID Preservation with Learned Condition Encoders



FastComposer: Tuning-Free Multi-Subject Image Generation with Localized Attention, Gal et al., ArXiv May 2023

ID Preservation with Learned Condition Encoders

BLIP-Diffusion: Pre-trained Subject Representation for Controllable Text-to-Image Generation and Editing,
Li et al., NeurIPS 2024

FastComposer

uning-Free Multi-Subject Image Generation with Localized Attention

(CLIP-based image encoder)

Code

https://github.com/mit-han-lab/fastcomposer/tree/main

Demo

https://replicate.com/cjwbw/fastcomposer

BLIP-Diffusion

Pre-trained Subject Representation for Controllable Text-to-Image Generation and Editing

(BLIP-based image encoder)

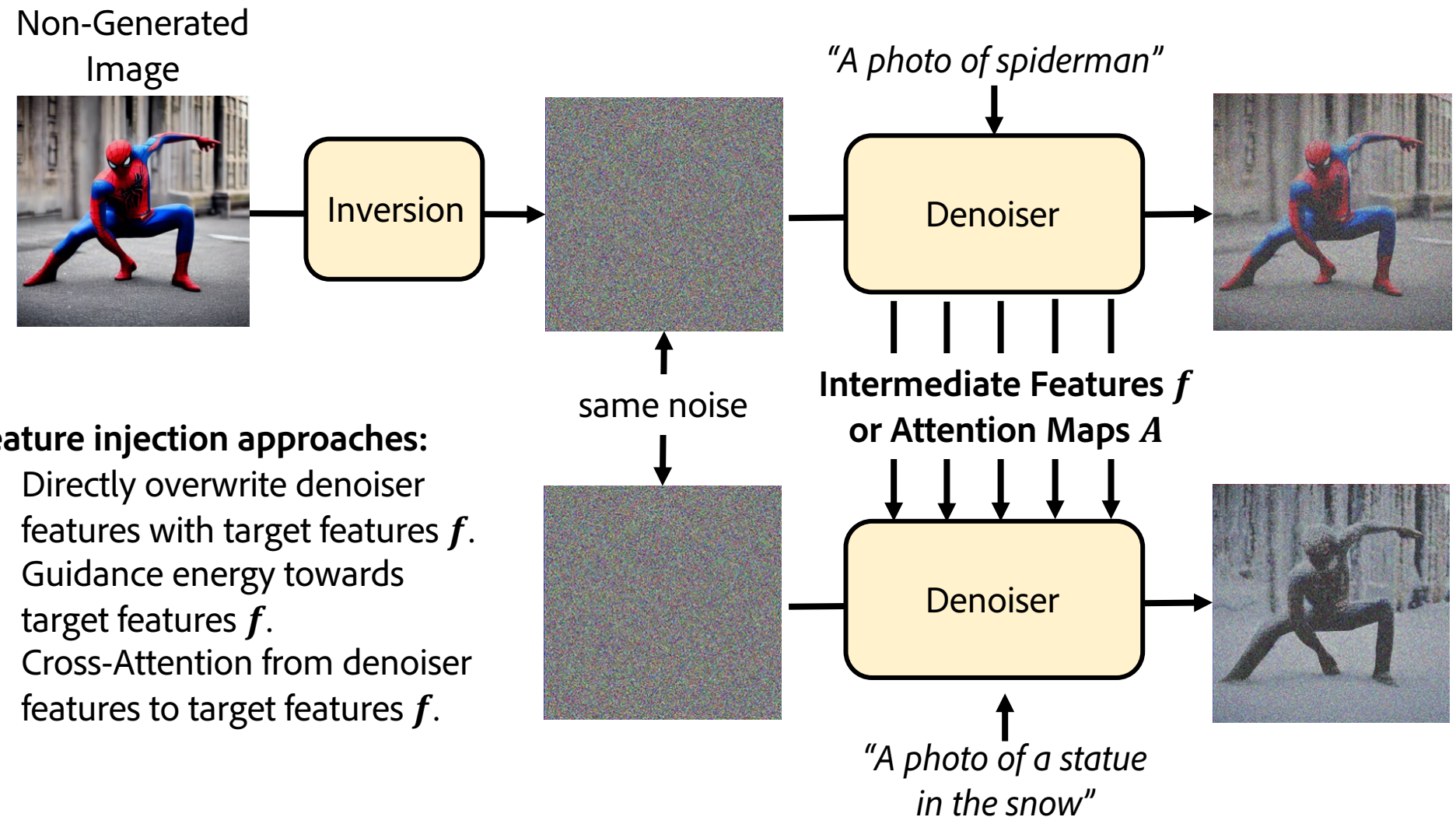
Code

https://huggingface.co/docs/diffusers/en/api/pipelines/blip_diffusion

Code

https://github.com/salesforce/LAVIS/tree/main/projects/blip-diffusion

ID Preservation through Intermediate Features



Feature injection approaches:

1) Directly overwrite denoiser features with target features f.
2) Guidance energy towards target features f.
3) Cross-Attention from denoiser features to target features f.

Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation, Tumanyan et al., CVPR 2023

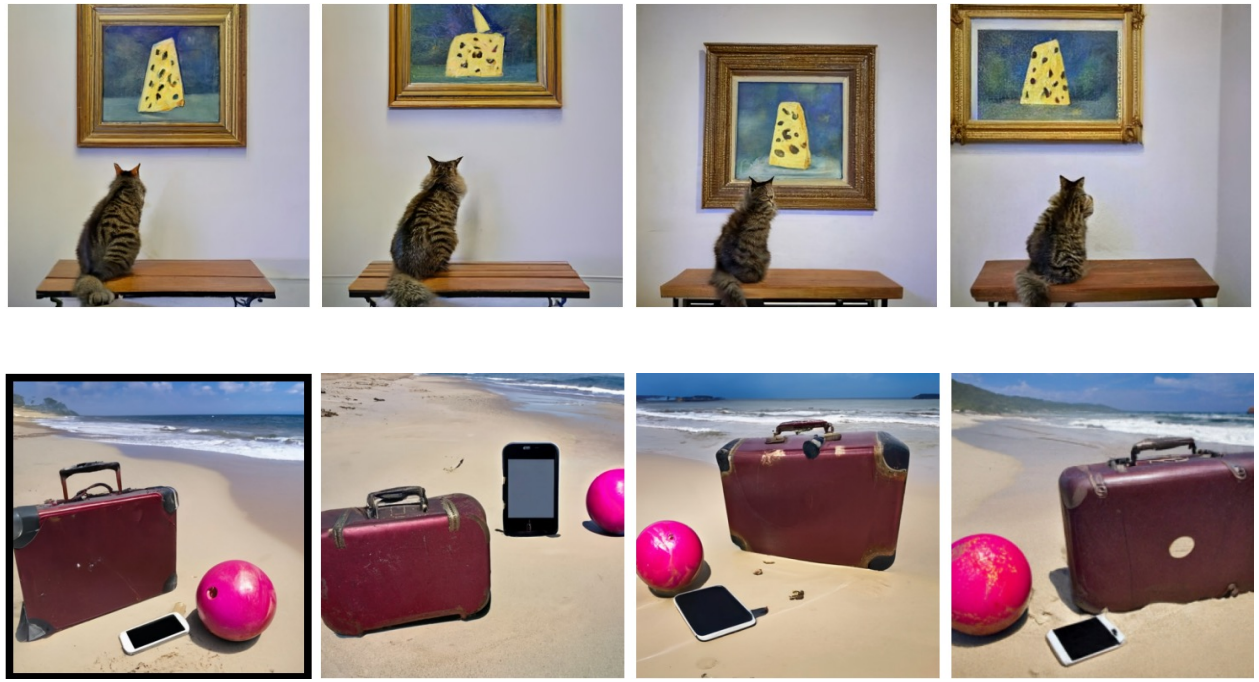
Diffusion Self-Guidance for Controllable Image Generation, Epstein et al., NeurIPS 2023

MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing, Cao et al., ICCV 2023

ID Preservation through Intermediate Features

- No training or fine-tuning needed.
- Intermediate features entangle identity with location and context, thus it requires additional work to allow for large changes in locations & contexts.

Guidance energy towards target features f.



Diffusion Self-Guidance for Controllable Image Generation, Epstein et al., NeurIPS 2023

Cross-Attention from denoiser features to target features f.

MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing, Cao et al., ICCV 2023

ID Preservation through Intermediate Features

- No training or fine-tuning needed.
- Intermediate features entangle identity with location and context, thus it requires additional work to allow for large changes in locations & contexts.

Directly overwrite denoiser features with target features f.

Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation, Tumanyan et al., CVPR 2023

MasaCtrl

Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing
(Cross-Attention-Based Feature Injection)

Code

https://github.com/TencentARC/MasaCtrl

PnP-Diffusers

Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation
(Overwrite-Based Feature Injection)

Demo

https://huggingface.co/spaces/hysts/PnP-diffusion-features

Code

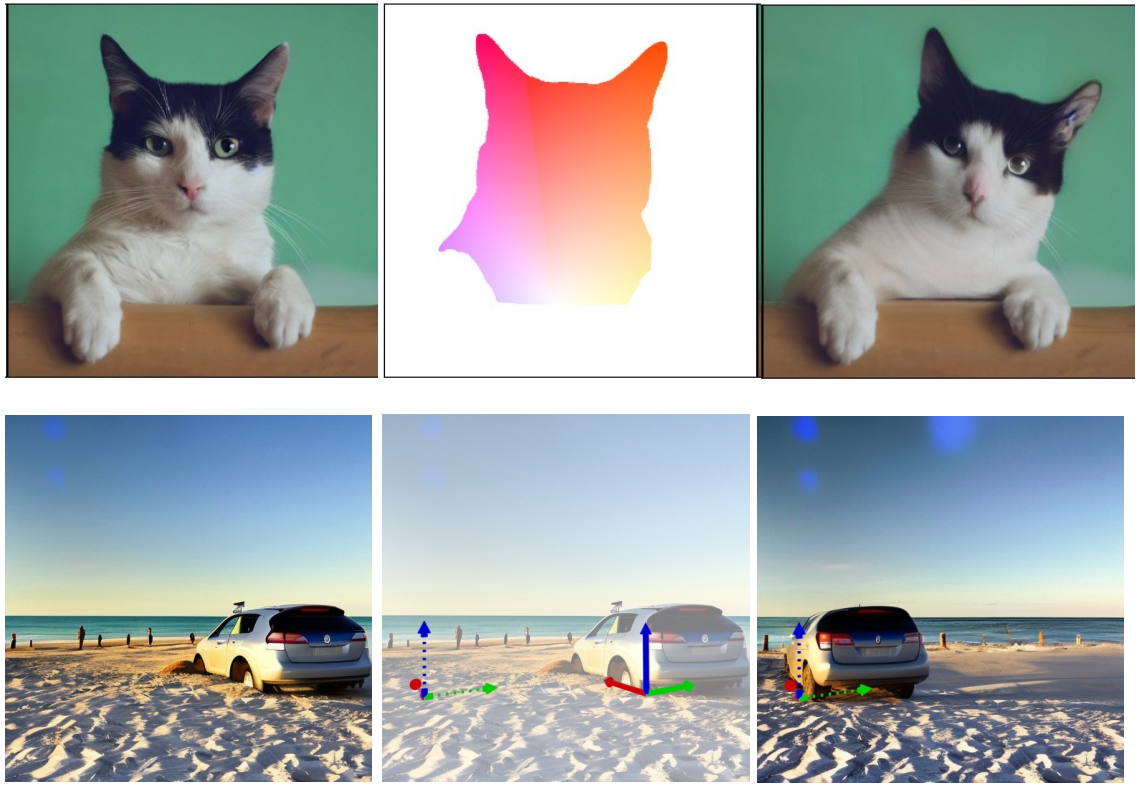
https://github.com/MichalGeyer/pnp-diffusers

Editing with Generative Models

Personalization

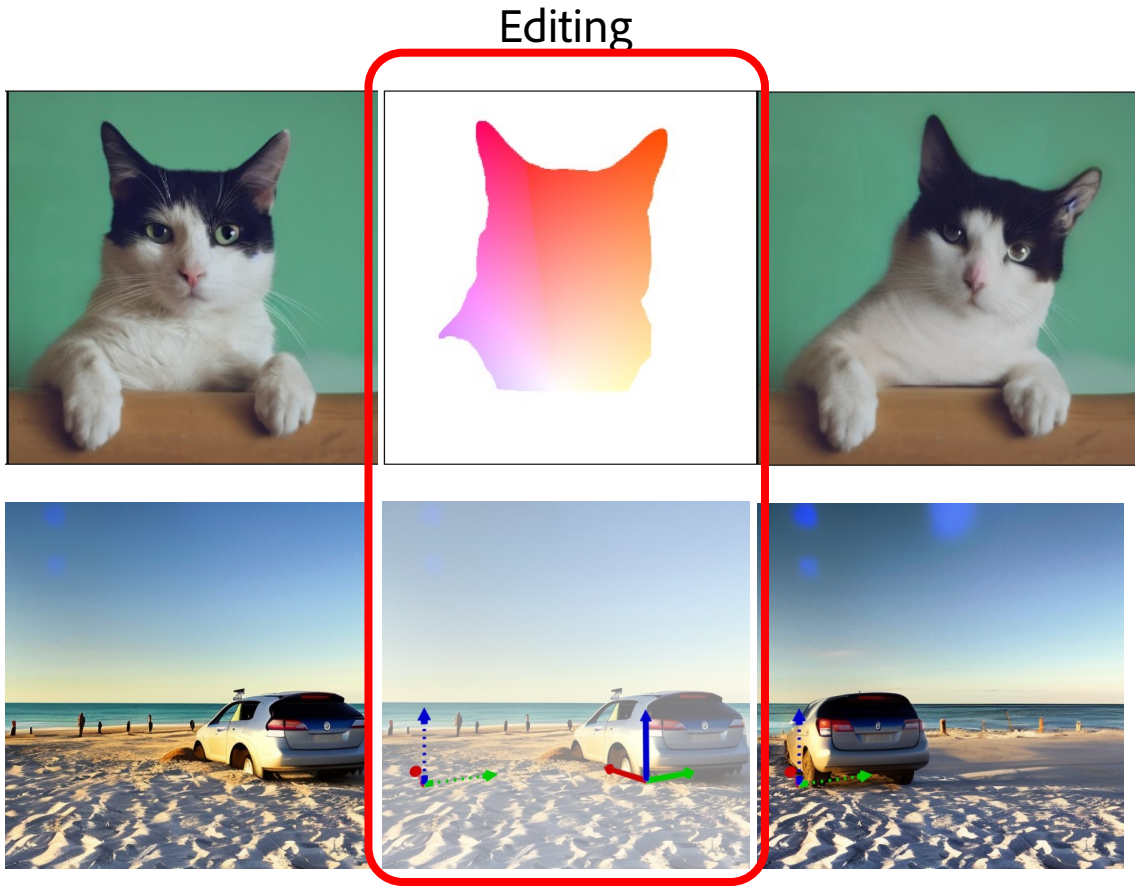
ConsiStory: Training-Free Consistent Text-to-Image Generation
Tewel et al., ArXiv Feb. 2024

Editing



Diffusion Handles: Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D Pandey et al., CVPR 2024
Motion Guidance: Diffusion-Based Image Editing with Differentiable Motion Estimators, Geng and Owens, ICLR 2024

Editing with Generative Models



Same subject, same scene.
Subject property changed by user **edit**.
(Property such as position, pose, etc.)

Editing:

Generative Model
+ Identity Preservation
+ Edit Control

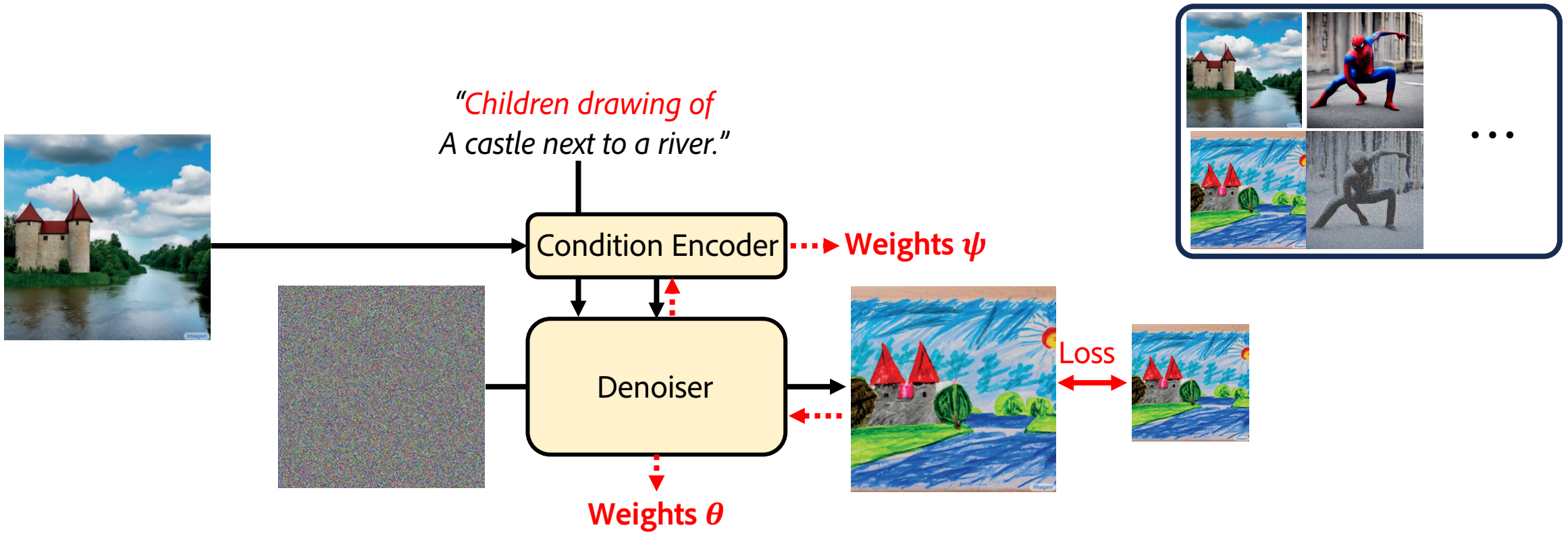
Diffusion Handles: Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D Pandey et al., CVPR 2024
Motion Guidance: Diffusion-Based Image Editing with Differentiable Motion Estimators, Geng and Owens, ICLR 2024

Image-to-Image Translation

Edit Control: Text Prompt

Identity Preservation: Condition Encoder

Large Paired Image Dataset

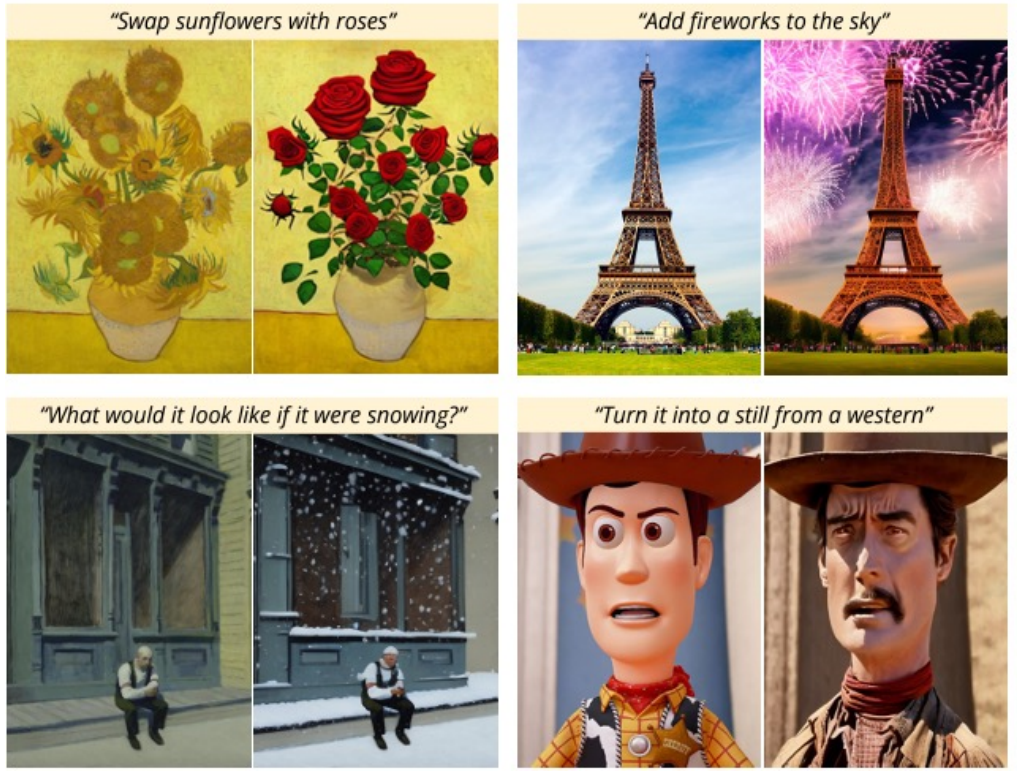


InstructPix2Pix: Learning to Follow Image Editing Instructions,
Brooks et al., CVPR 2023

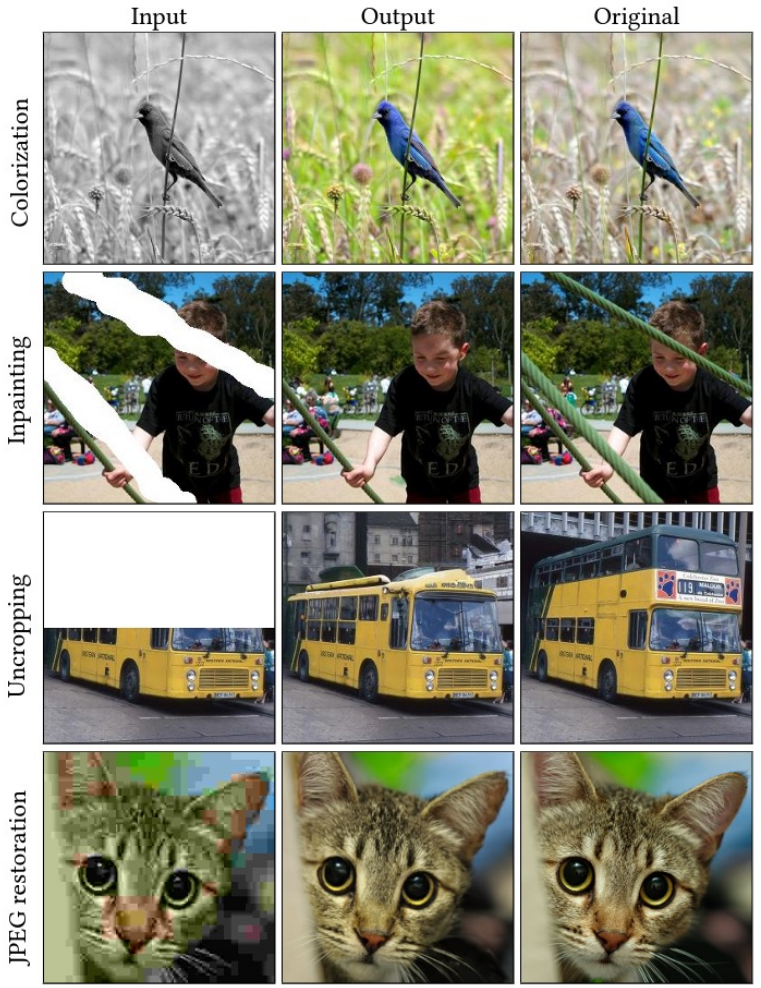
Palette: Image-to-Image Diffusion Models,
Saharia et al., SIGGRAPH 2022

Image-to-Image Translation

- Requires paired image dataset.
- Text prompt provides only coarse control.



InstructPix2Pix: Learning to Follow Image Editing Instructions,
Brooks et al., CVPR 2023



Palette: Image-to-Image Diffusion Models,
Saharia et al., SIGGRAPH 2022

InstructPix2Pix

Learning to Follow Image Editing Instructions

(Edit Instructions)

Code

https://github.com/timothybrooks/instruct-pix2pix

Demo

https://huggingface.co/spaces/timbrooks/instruct-pix2pix

Palette

Image-to-Image Diffusion Models

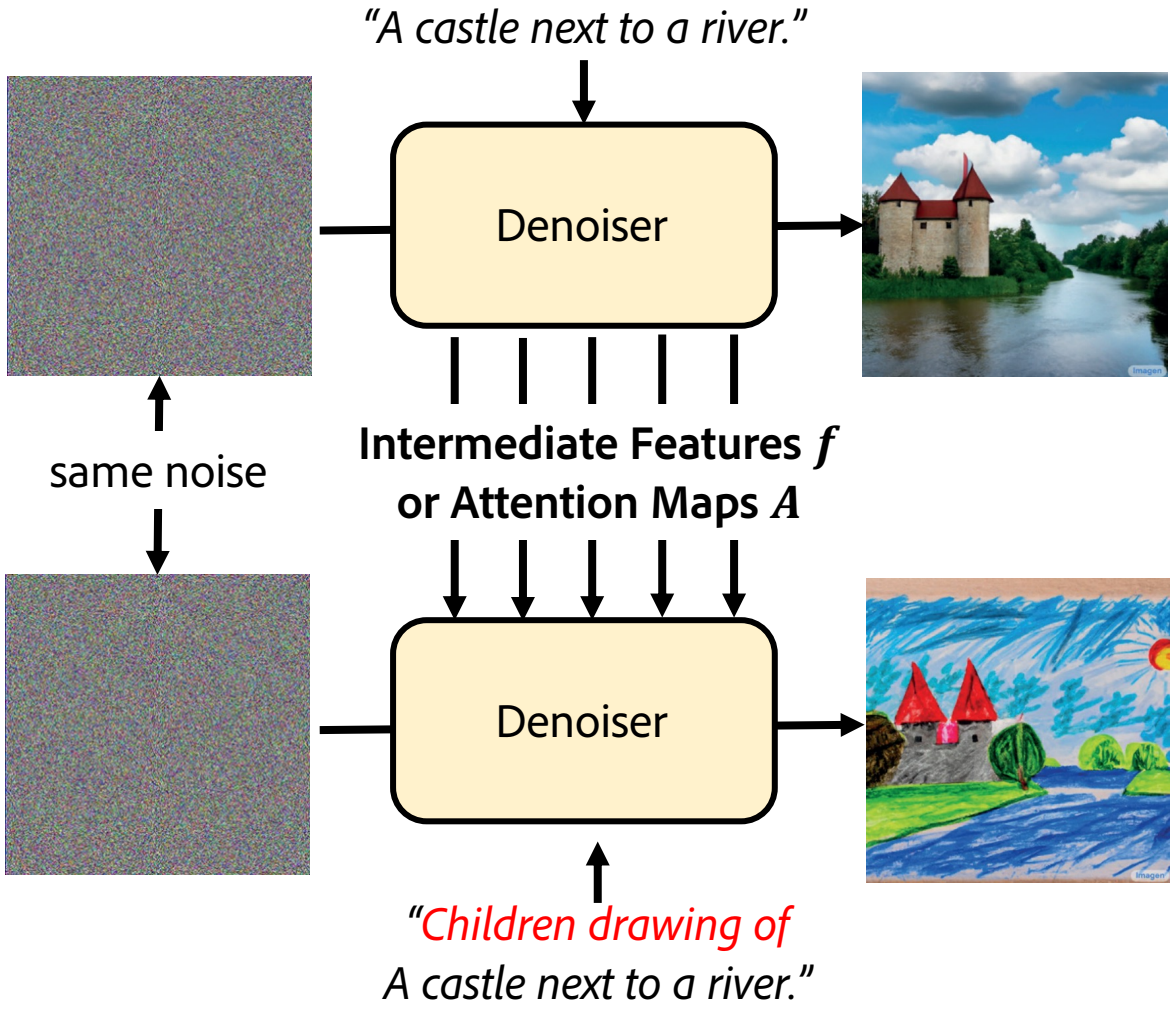
(Per-Task Fine-tuning)

Code

https://github.com/Janspiry/Palette-Image-to-Image-Diffusion-Models

Image-to-Image Translation

Edit Control:
Text Prompt
Identity Preservation:
Intermediate Features / Attention Maps



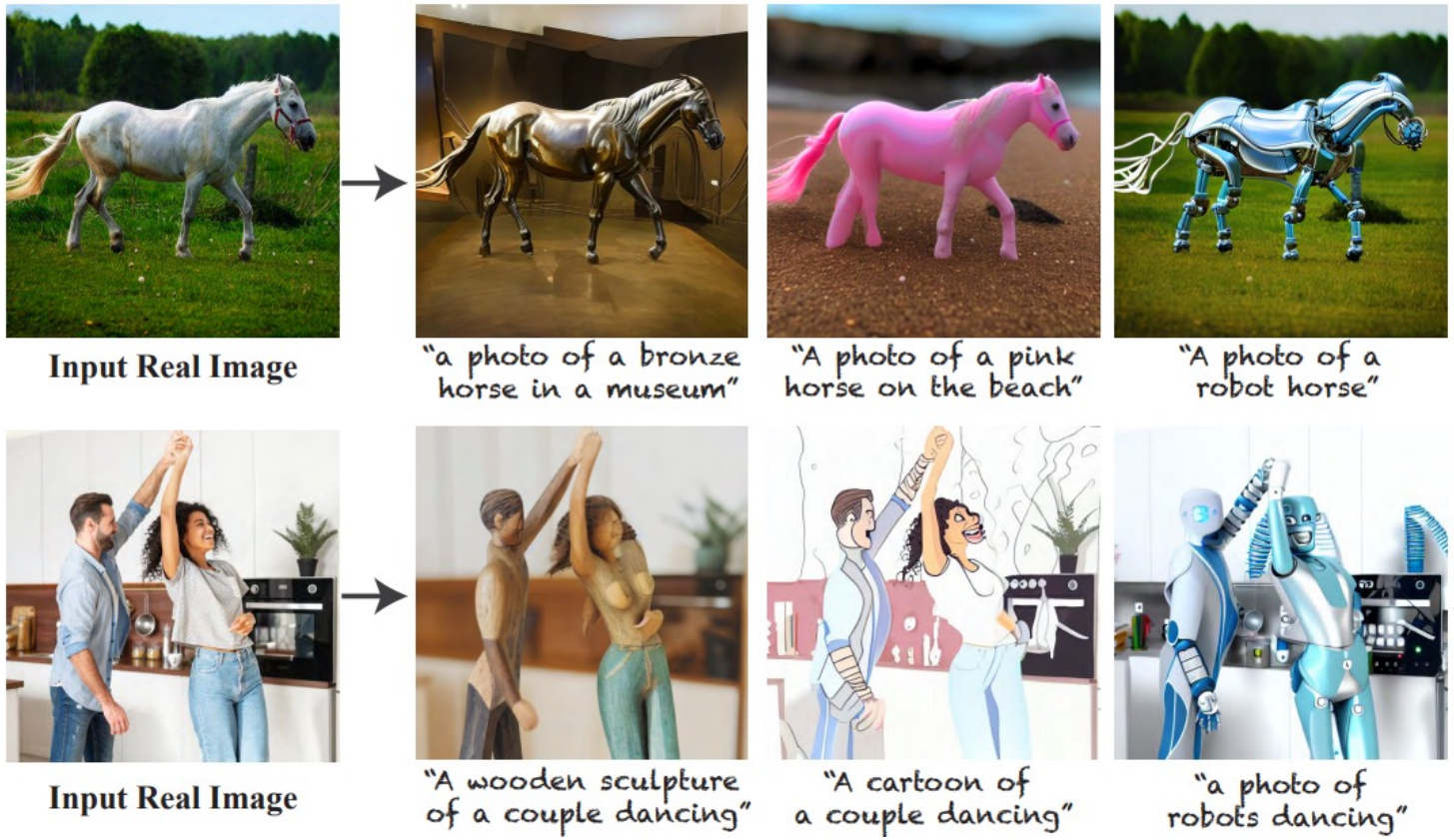
Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation, Tumanyan et al., CVPR 2023

Prompt-to-Prompt Image Editing with Cross Attention Control, Hertz et al., ArXiv Aug. 2022

MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing, Cao et al., ArXiv Aug. 2022

Image-to-Image Translation

- No training or fine-tuning needed.
- Cannot strongly change scene layout (object positions in the image remain roughly the same)
- Text prompt provides only coarse control.



Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation, Tumanyan et al., CVPR 2023

Prompt-to-Prompt Image Editing with Cross Attention Control, Hertz et al., ArXiv Aug. 2022

Image-to-Image Translation

- No training or fine-tuning needed.
- Cannot strongly change scene layout (object positions in the image remain roughly the same)
- Text prompt provides only coarse control.

MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing, Cao et al., ICCV 2023

PnP-Diffusers

Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation

(Overwrite-Based Feature Injection)

Demo

https://huggingface.co/spaces/hysts/PnP-diffusion-features

Code

https://github.com/MichalGeyer/pnp-diffusers

Prompt-to-Prompt

Image Editing with Cross Attention Control

(Overwrite-Based Feature Injection)

Code

https://github.com/google/prompt-to-prompt/

MasaCtrl

Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing

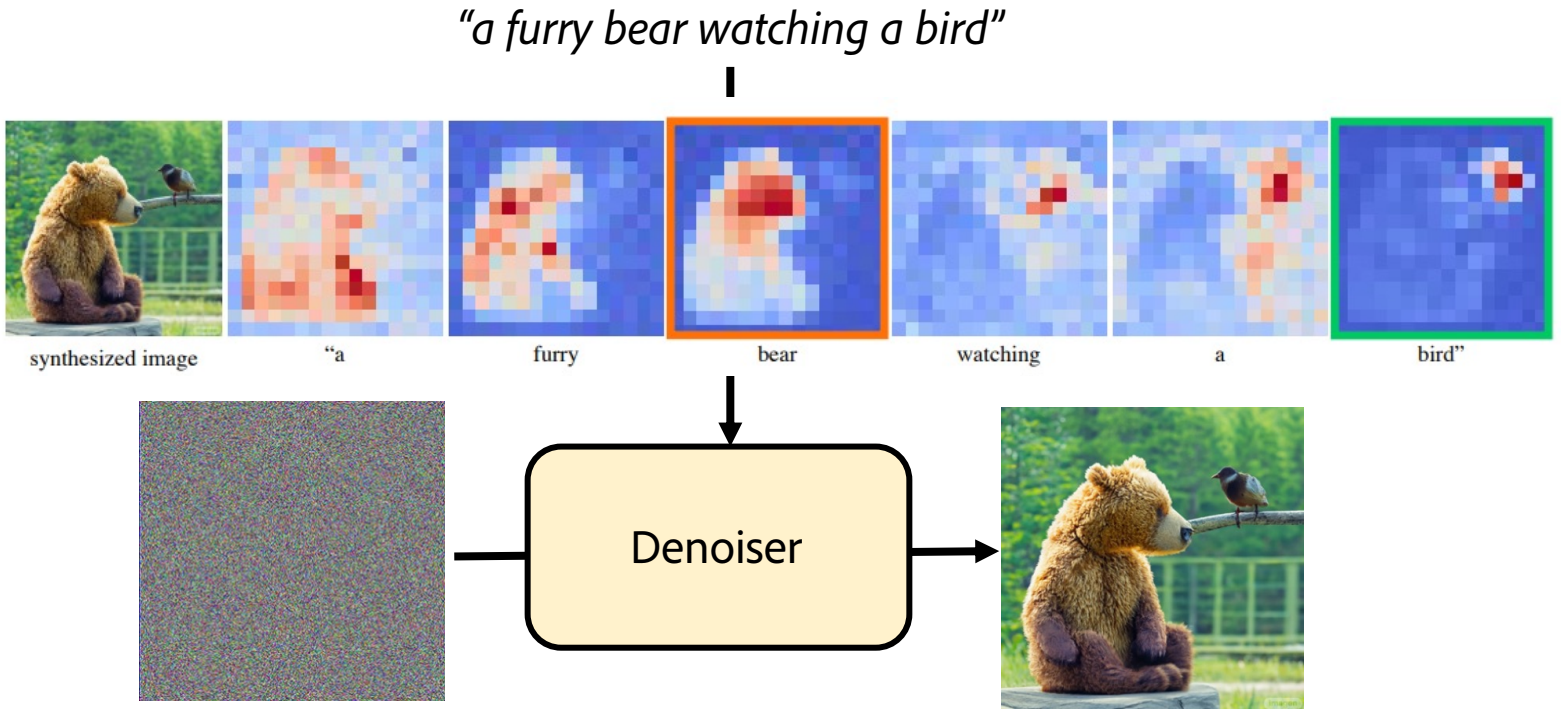
(Cross-Attention-Based Feature Injection)

Code

https://github.com/TencentARC/MasaCtrl

Using Attention Maps & Intermediate Features

Edit Control: Transform Intermediate Features / Attention Maps
Identity Preservation: Intermediate Features / Attention Maps



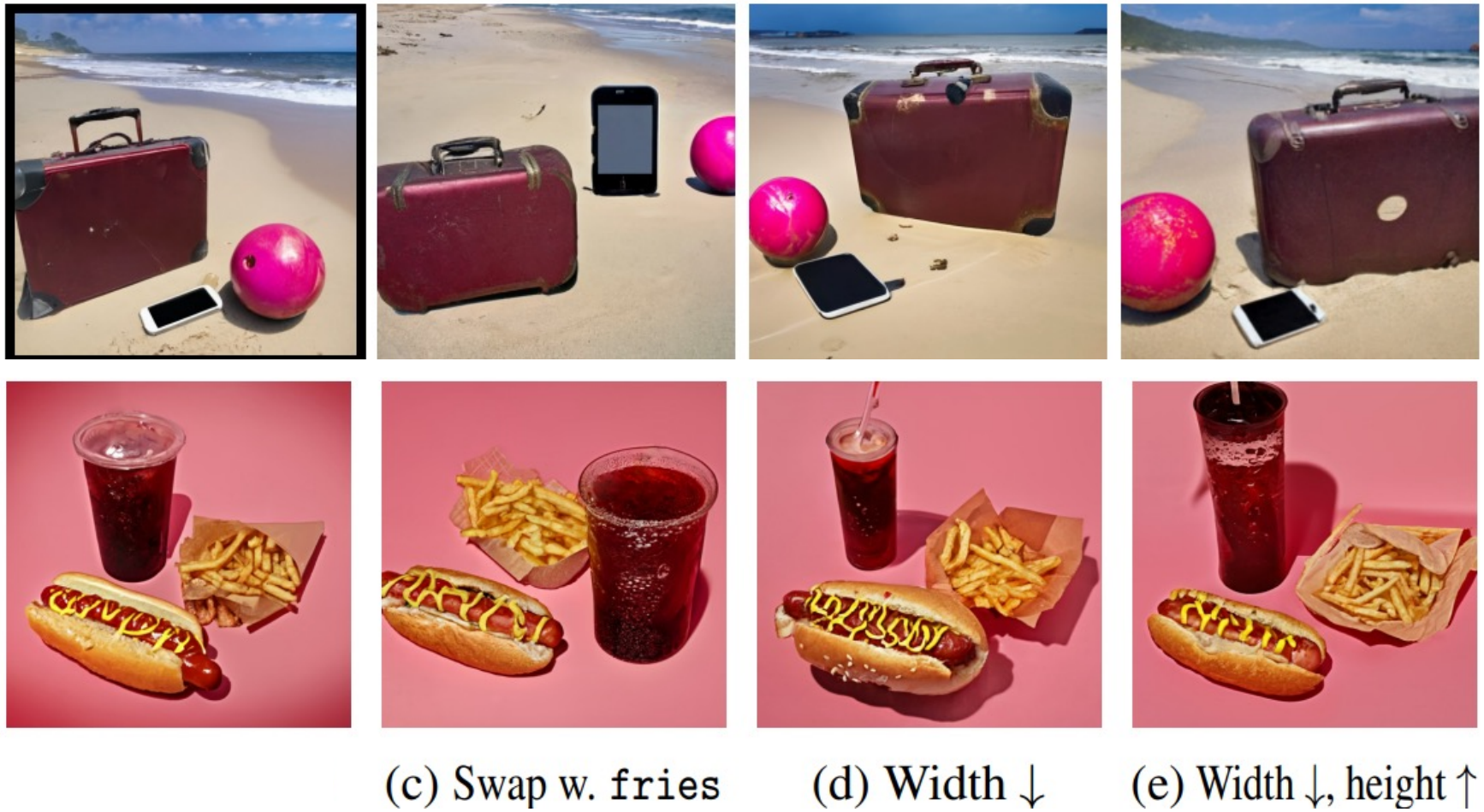
Attention maps from pixels to text tokens.

Prompt-to-Prompt Image Editing with Cross Attention Control, Hertz et al., ArXiv Aug. 2022

Diffusion Self-Guidance for Controllable Image Generation, Epstein et al., NeurIPS 2023

Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D, CVPR 2024

Using Attention Maps & Intermediate Features



Diffusion Self-Guidance for Controllable Image Generation,
Epstein et al., NeurIPS 2023

Prompt-to-Prompt

Image Editing with Cross Attention Control

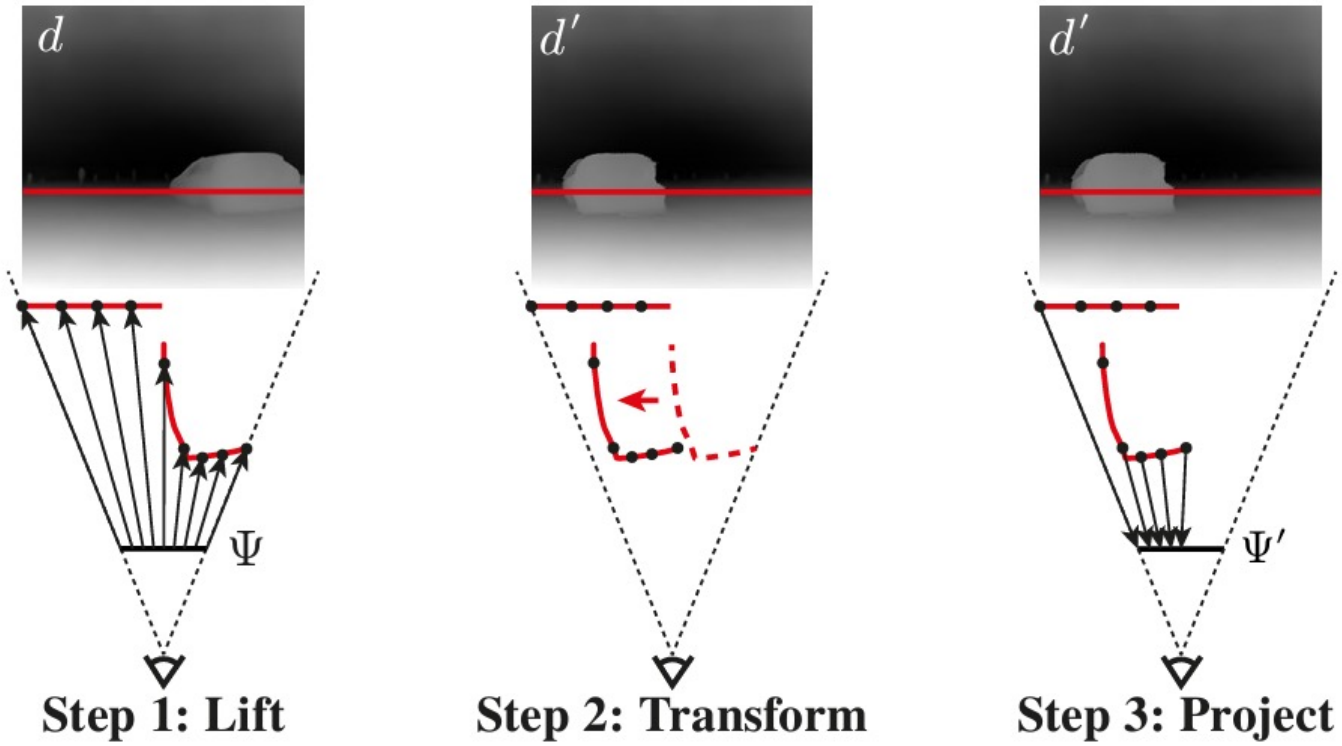
(Overwrite-Based Feature Injection)

Code

https://github.com/google/prompt-to-prompt/

Editing with Attention Maps and Intermediate Features

Attention maps / intermediate features can be 3D-transformed using monocular depth estimates.



Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D, CVPR 2024

Attention Maps & Intermediate Features

Attention maps / intermediate features can be 3D-transformed using monocular depth estimates.

Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D, CVPR 2024

Attention Maps & Intermediate Features

Attention maps / intermediate features can be 3D-transformed using monocular depth estimates.

Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D, CVPR 2024

Attention Maps & Intermediate Features

Attention maps / intermediate features can be 3D-transformed using monocular depth estimates.

Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D, CVPR 2024

Diffusion Models for Visual Computing

Chun-Hao Huang

Niloy Mitra, Daniel Cohen-Or, Minhyuk Sung, Duygu Ceylan, Paul Guerrero

Part 5: Beyond Single Images

https://geometry.cs.ucl.ac.uk/courses/diffusion4VC_eg24/

Presentation Schedule

Introduction to Diffusion Models

Guidance and Conditioning Sampling

Attention

Break

Personalization and Editing

Beyond Single Images

Diffusion Models for 3D Generation

So Far

- Theory and principles of diffusion models
- 2D image generation and editing
  - Editing one single image
  - Identity preservation

This Section

- Editing “multiple images”
  - in temporal dimension → videos
  - in 2D image domain → image montage
  - in 3D spatial domain → multi-views (*without* explicit 3D awareness)

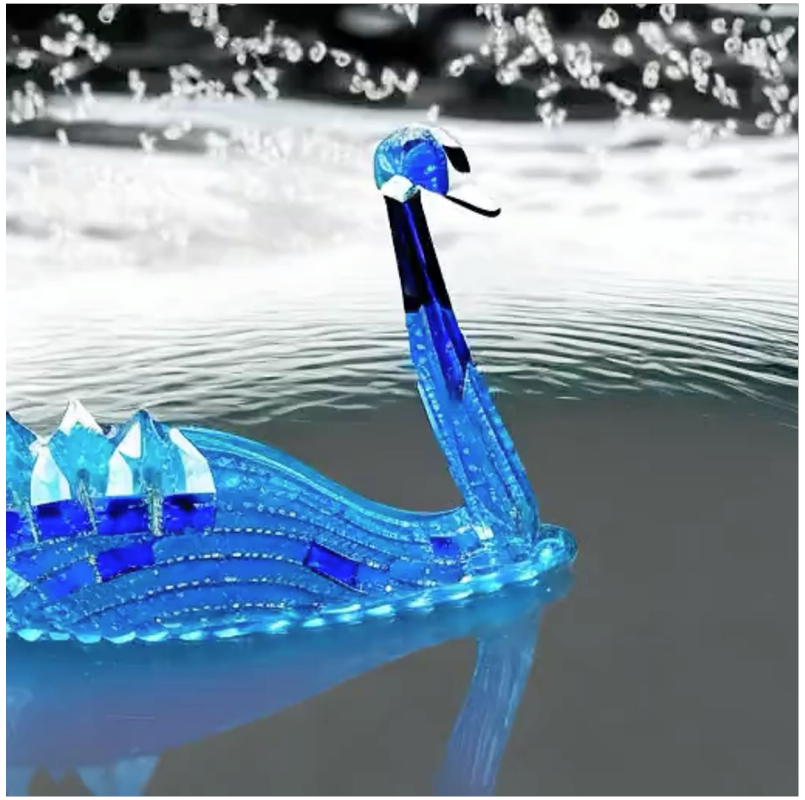
- Training-free / zero-shot setup
  - Repurposing existing pretrained image diffusion model

- Training setup
  - Video diffusion model

Individual Editing – Videos

input

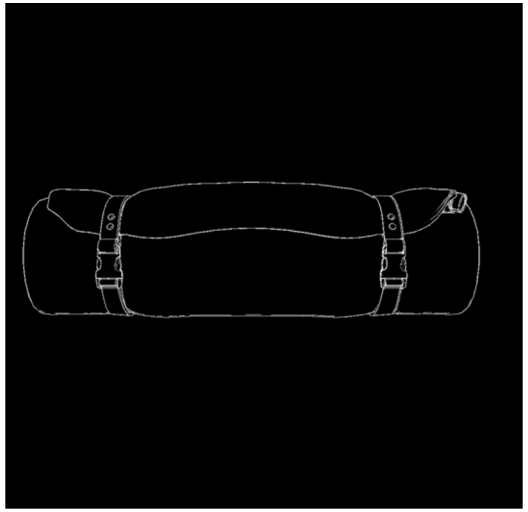
per-frame edit



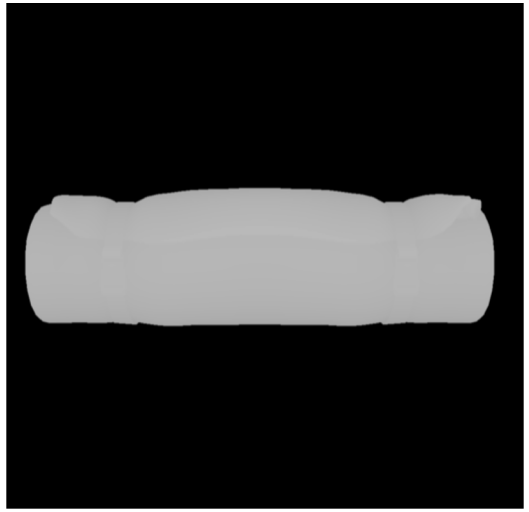
depth-conditioned SD

Individual Editing – Multiple Views

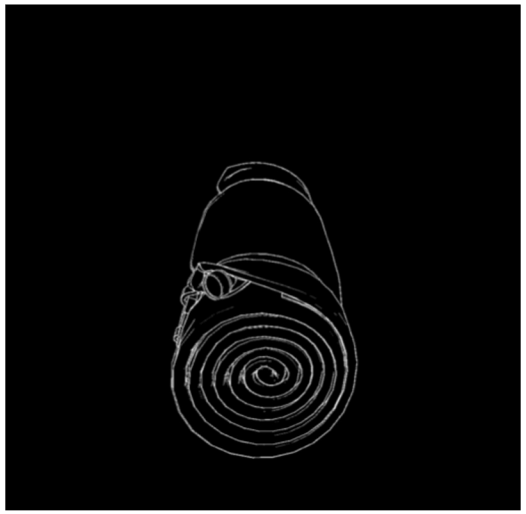
input

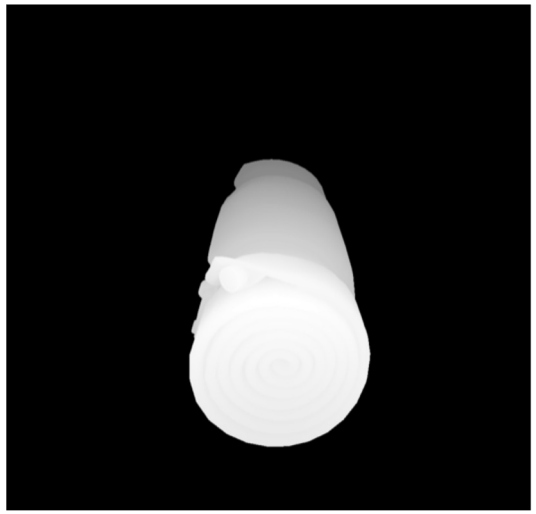


input



per-view edits





Firefly structure match

Design Space for Consistency

- If *correspondences are known*, synchronzing:
  1. Initial noise and/or noisy latents
  2. Features
- Loss-guided denoising.
- Repurposing self attention for cross-view, cross-frame attention.

Noise Model

- Denoting the initial noise of each frame as $\epsilon^1, \epsilon^2, \dots, \epsilon^i, \dots$
- I.I.D.: $\epsilon^1, \epsilon^2, \dots, \epsilon^i, \dots \sim \mathcal{N}(0, \mathbf{I})$
- Re-using the same noise: $\epsilon^1 = \epsilon^2 = \epsilon^i = \cdots \sim \mathcal{N}(0, \mathbf{I})$
- Mixed Noise Model [1]

$$\epsilon_{\text{shared}} \sim \mathcal{N}\left(0, \frac{\alpha^2}{1+\alpha^2}\mathbf{I}\right), \epsilon_{\text{ind}}^i \sim \mathcal{N}\left(0, \frac{\alpha^2}{1+\alpha^2}\mathbf{I}\right)$$
$$\epsilon^i = \epsilon_{\text{shared}} + \epsilon_{\text{ind}}^i$$

- Progressive Noise Model [1]

$$\epsilon^0 \sim \mathcal{N}(0, \mathbf{I}), \epsilon_{\text{ind}}^i \sim \mathcal{N}\left(0, \frac{1}{\sqrt{1+\alpha^2}}\mathbf{I}\right)$$
$$\epsilon^i = \frac{\alpha}{1+\alpha^2}\epsilon^{i-1} + \epsilon_{\text{ind}}^i$$

[1] Ge et al., Preserve Your Own Correlation: A Noise Prior for Video Diffusion Models, *ICCV'23*

Correspondence-guided Noise

Kass and Pesare, Coherent noise for non-photorealistic rendering, *ToG'11*

Correspondence-guided Noise

- $\epsilon^0 = \sqrt{\frac{1-\alpha}{1+\alpha}}\epsilon^0_{\text{ind}}$, where $\epsilon^0_{\text{ind}} \sim \mathcal{N}(0, \mathbf{I})$

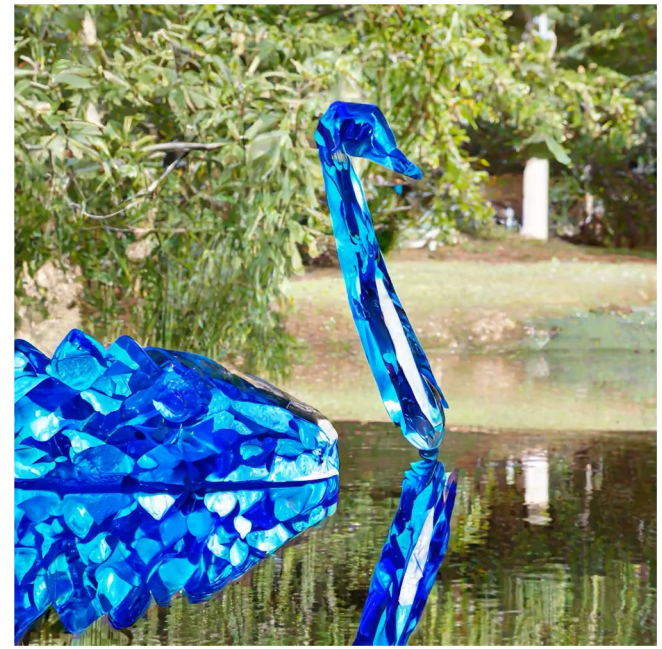
- $$\epsilon^i(x,y) = \begin{cases} \sqrt{\frac{1-\alpha}{1+\alpha}}\epsilon^i_{\text{ind}}(x,y), & \epsilon^i_{\text{ind}} \sim \mathcal{N}(0, \mathbf{I}) \\ \alpha\epsilon^{i-1}(x',y') + (1-\alpha)\epsilon^i_{\text{ind}}(x,y) \end{cases}$$

disocclusion

if correspondence pairs (x', y') & (x, y) exist e.g., obtained by optical flow.

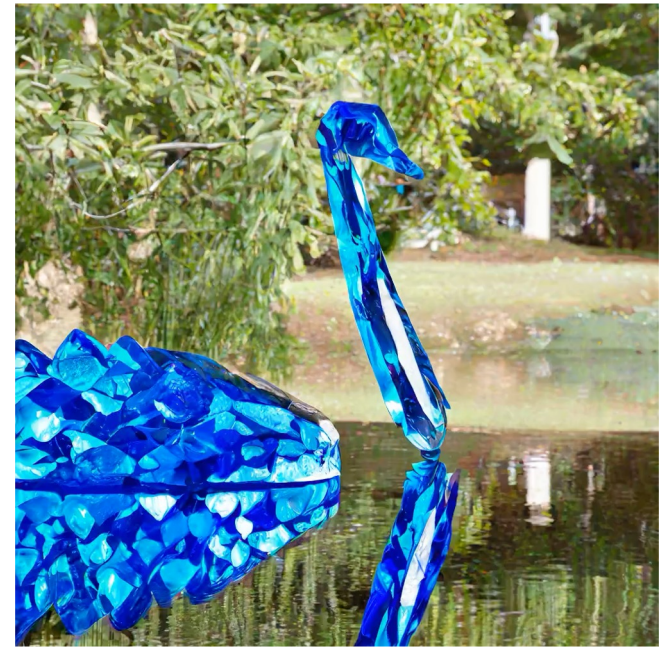
input

results using same rand initialized noise



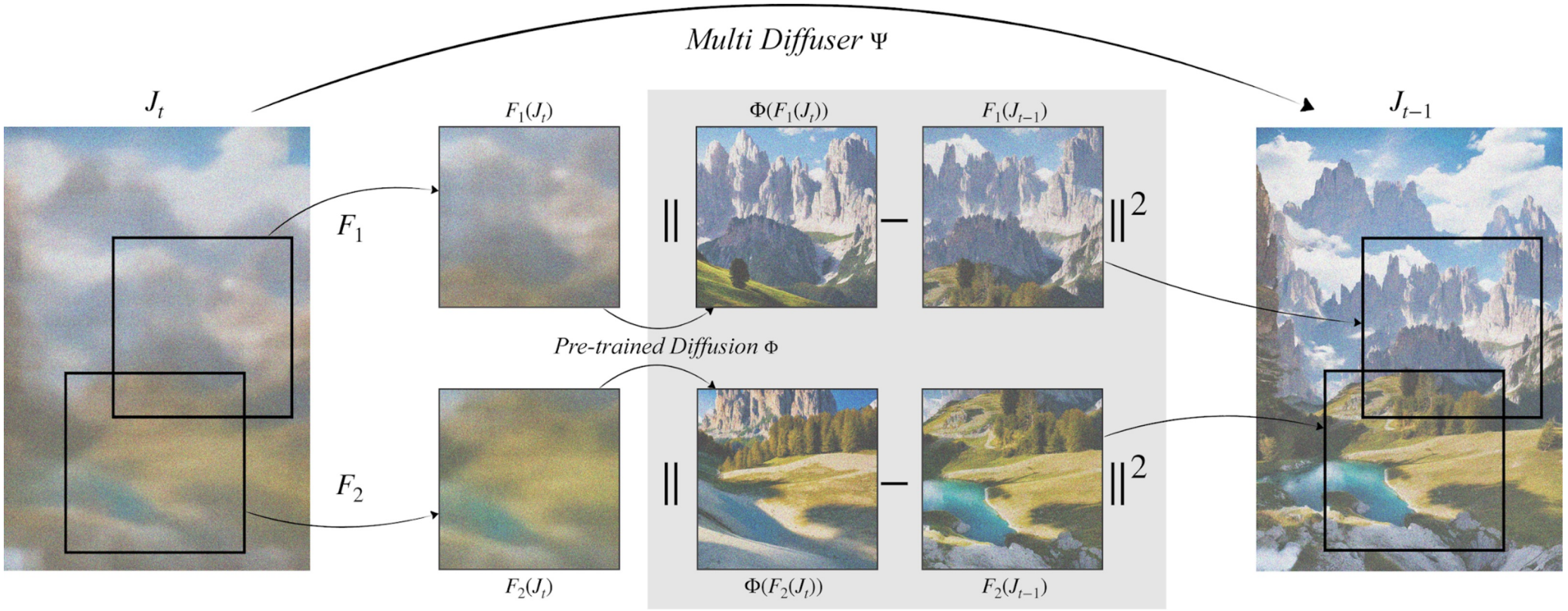
correspondence guided noise

results using correspondence guided noise



Synchronizing Colors or Latent Features

- Averaging noisy latents of the same pixels



Bar-Tal et al., MultiDiffusion: Fusing Diffusion Paths for Controlled Image Generation, *ICML'23*

Loss-guided Denoising

$\mathcal{D}(\mathbf{x}_t)$ noisy image at t=45

0.6485 0.6433

$\mathcal{D}(\phi_\theta(\mathbf{x}_t, t))$ predicted x0 at t=45

0.5416 0.3501

$\mathcal{D}(\mathbf{x}_0)$ final denoised x0

0.5911 0.3704

$$\hat{\mathbf{x}}_t^{(i)} = \mathbf{x}_t^{(i)} - w\nabla_{\mathbf{x}_t^{(i)}}\mathcal{L}\left(\mathcal{D}(\phi_\theta(\mathbf{x}_t^{(i)}, t)), \mathcal{D}(\phi_\theta(\mathbf{x}_t^{(0)}, t))\right)$$

Loss $\mathcal{L}$: LPIPS score on the predicted denoised x0

Lee et al., SyncDiffusion: Coherent Montage via Synchronized Joint Diffusions, *NeurIPS'23*

without loss-guided denoising

with loss-guided denoising

MultiDiffusion

Fusing Diffusion Paths for Controlled Image Generation
(averaging noisy latents)

Code

https://github.com/omerbt/MultiDiffusion

SyncDiffusion

Coherent Montage via Synchronized Joint Diffusions
(loss-guided denoising)

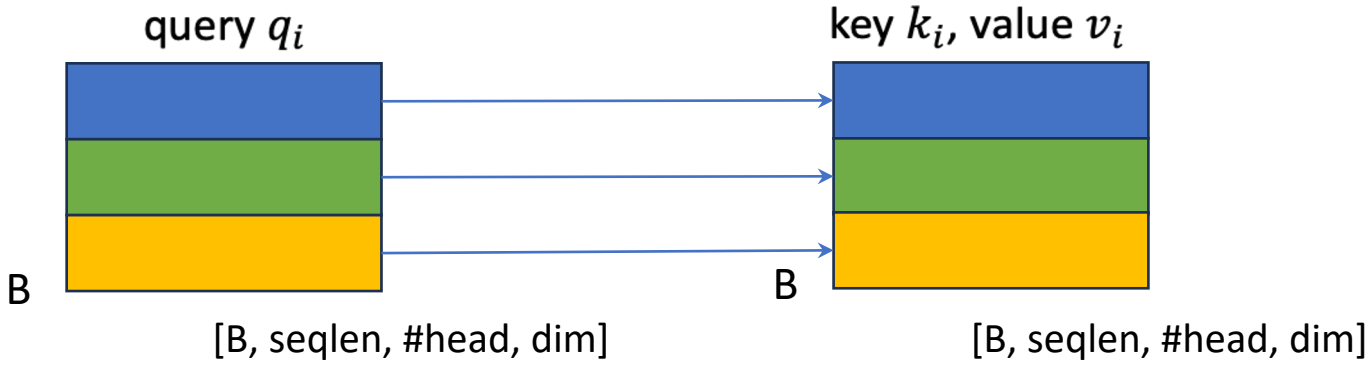
Code

https://github.com/KAIST-Visual-AI-Group/SyncDiffusion

Self-attention in UNet

$$Att(q_i, k_i, v_i) = softmax\left(\frac{q_i k_i^{\mathrm{T}}}{\sqrt{d}}\right) v_i$$

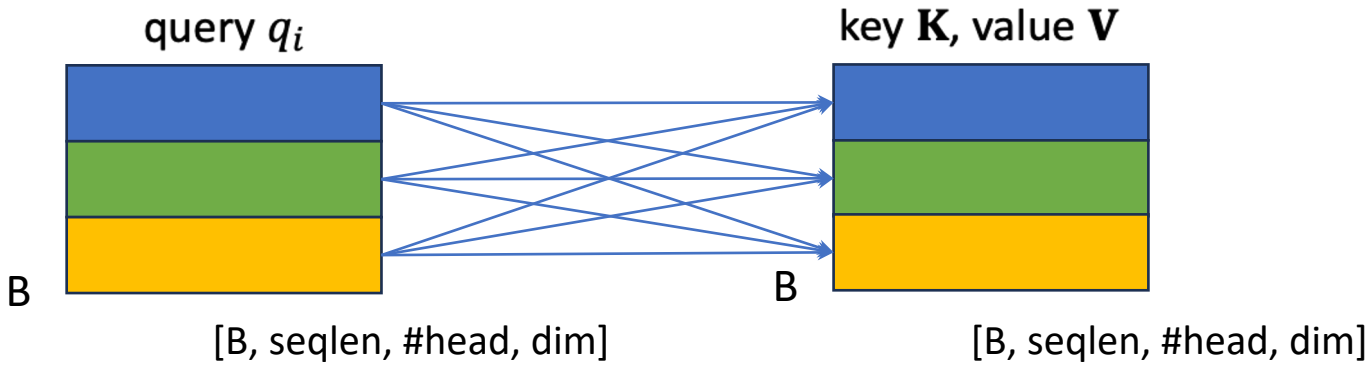
- Independent generation: each sample in the batch attends to only itself.



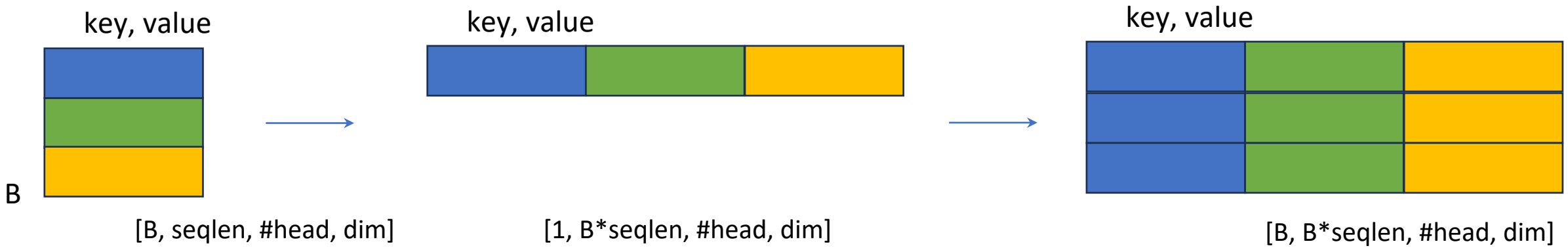
Repurposing Self Attention

$$Att(q_i, \mathbf{K}, \mathbf{V}) = softmax\left(\frac{q_i \mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V},$$
$$\text{where } \mathbf{K} = [k_1, \dots k_i, \dots], \mathbf{V} = [v_1, \dots v_i, \dots]$$

- Batch generation: each sample attends to every samples.

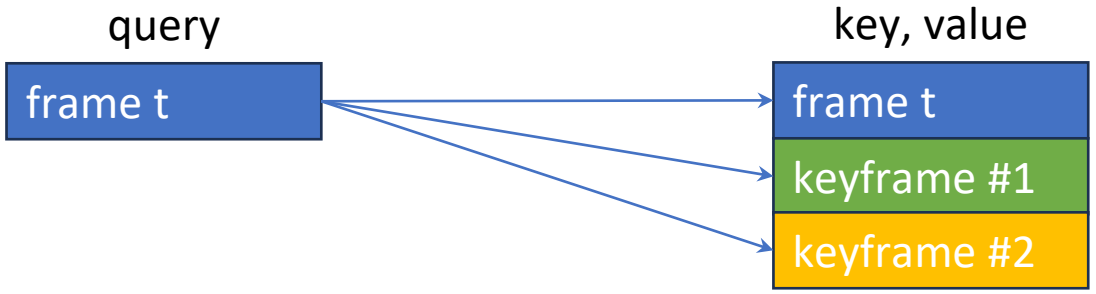


Pseudo Code




```
bs = k.shape[0]
k, v = (t.reshape(1, -1, attn.heads,attn.dim_head) for t in (k, v))
k = k.expand(bs, -1, -1, -1)
v = v.expand(bs, -1, -1, -1)
```

Training-free/zero-shot Video Stylization



- Each frame attends to pre-defined “keyframes”
  - Pix2Video [1]: keyframes = [frame 1, frame t-1]
  - FateZero [2]: keyframes = [middle frame]
  - Text2Video-Zero [3]: keyframes = [frame 1]

[1] Ceylan et al., Pix2Video: Video Editing using Image Diffusion, *ICCV’23*
[2] Qi et al., FateZero: Fusing Attentions for Zero-shot Text-based Video Editing, *ICCV’23*
[3] Khachatryan et al., Text2Video-Zero: Text-to-Image Diffusion Models are Zero-Shot Video Generators, *ICCV’23*

Text-guided Video Stylization

a group of chocolate pigs looking for food

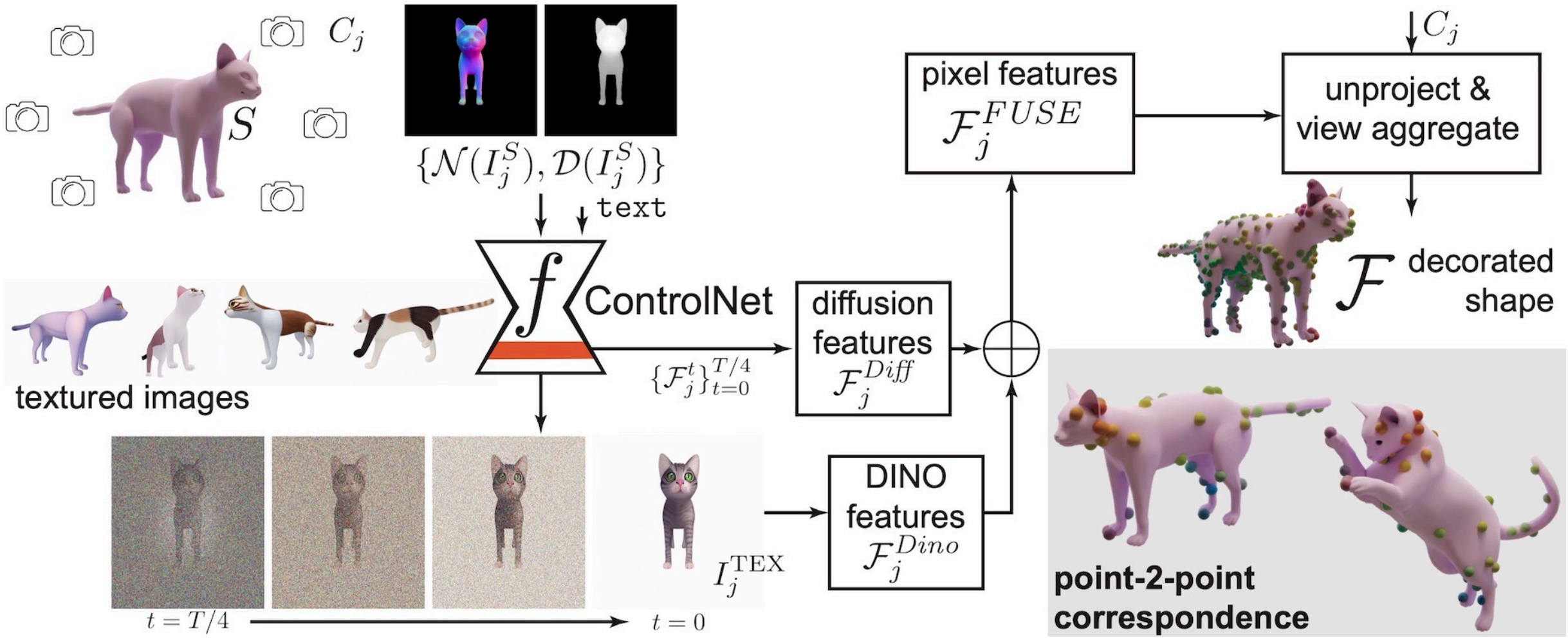
Ceylan et al., Pix2Video: Video Editing using Image Diffusion, *ICCV'23*

Another Example

a Swarovski blue crystal swan on the lake

Ceylan et al., Pix2Video: Video Editing using Image Diffusion, *ICCV'23*

Diffusion Features



Dutt et al., Diffusion 3D Features (Diff3F): Decorating Untextured Shapes with Distilled Semantic Features, *CVPR'24*

Diffusion Features

Dutt et al., Diffusion 3D Features (Diff3F): Decorating Untextured Shapes with Distilled Semantic Features, *CVPR'24*

Pix2Video

Video Editing using Image Diffusion

Code

https://github.com/duyguceylan/pix2video

FateZero

Fusing Attentions for Zero-shot Text-based Video Editing

Code

https://github.com/ChenyangQiQi/FateZero

Text2Video-Zero

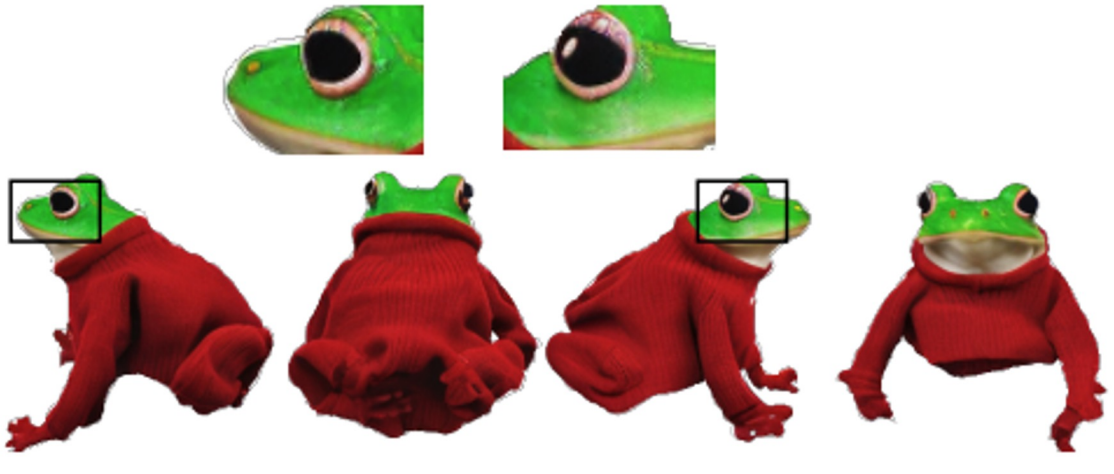
Text-to-Image Diffusion Models are Zero-Shot Video Generators

Code

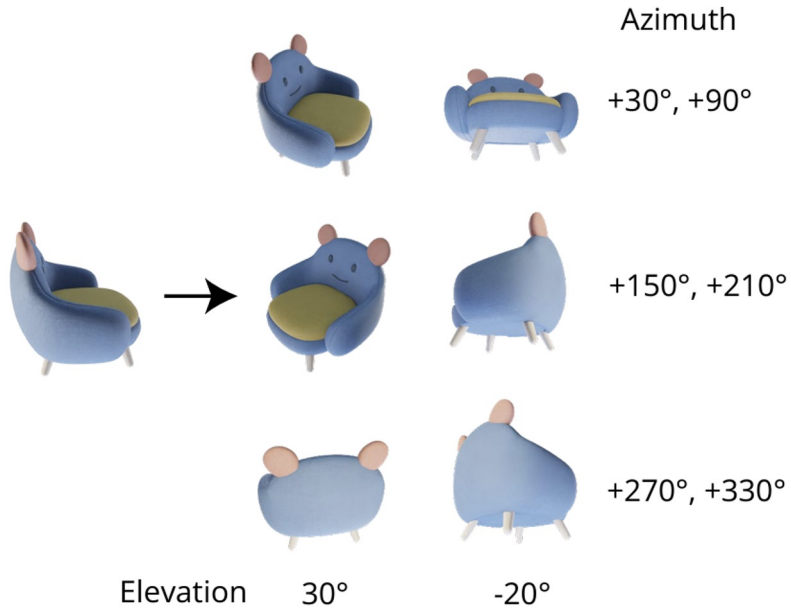
https://github.com/Picsart-AI-Research/Text2Video-Zero

Applied on Multi-view Generation

- Stacking views into an image grid
  a simple yet effective way of repurposing self-attention as cross-view attention.



Tsalicoglou et al., Textmesh: Generation of realistic 3D meshes from text prompts, arXiv'23



Shi et al., Zero123++: a Single Image to Consistent Multi-view Diffusion Base Model, arXiv'23

Zero123++

A Single Image to Consistent Multi-view Diffusion Base Model
(stacking 6 predefined views)

Code

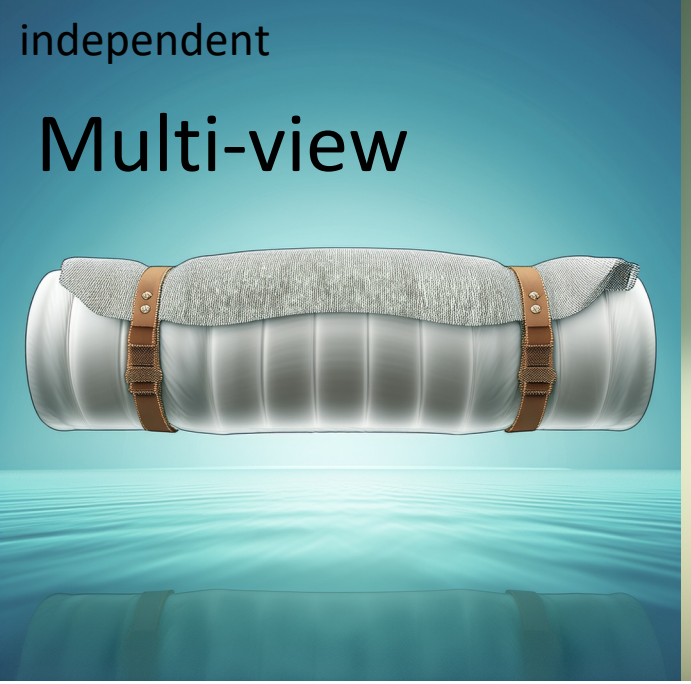
https://github.com/SUDO-AI-3D/zero123plus

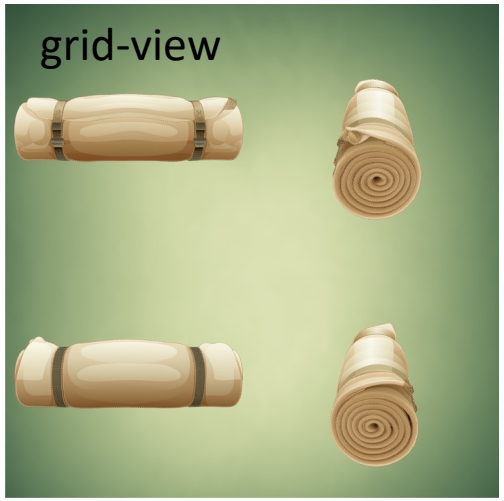
Demo

https://github.com/SUDO-AI-3D/zero123plus

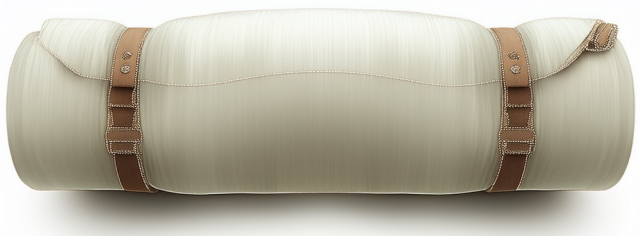
independent

Multi-view



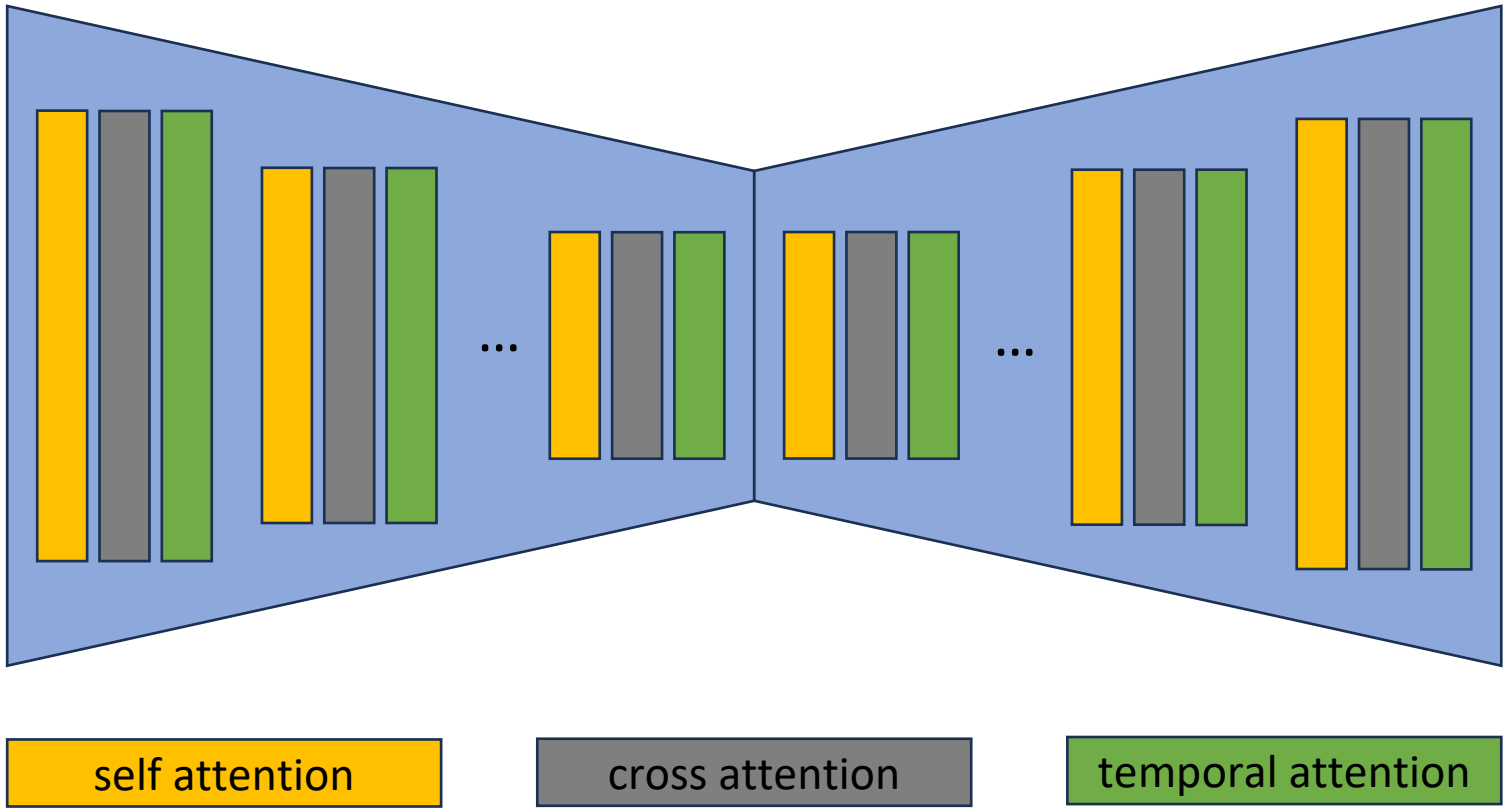


batch

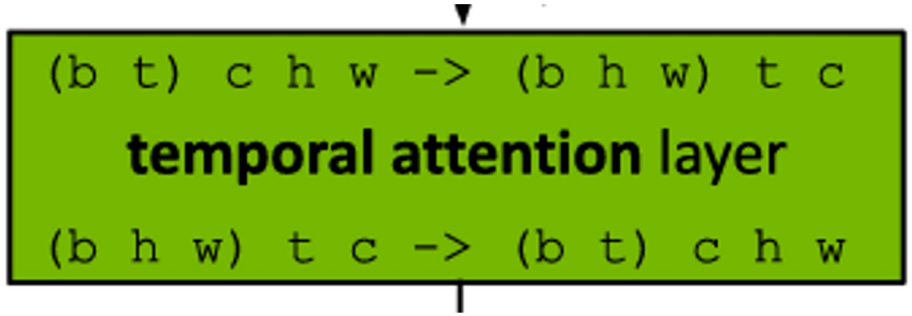


Video Diffusion Model

- Adding a temporal attention layer after spatial cross attention



Inflated Temporal Attention Layer



Blattmann et al., Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models, *CVPR'23*

Blattmann et al., Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets, *arXiv'23*

Guo et al., AnimateDiff: Animate Your Personalized Text-to-Image Diffusion Models without Specific Tuning, *ICLR'24*

Code

https://github.com/huggingface/diffusers/blob/main/src/diffusers/models/transformers/transformer_temporal.py

Applied on Multi-view Generation

Generating 360 turn-table videos of 3D objects

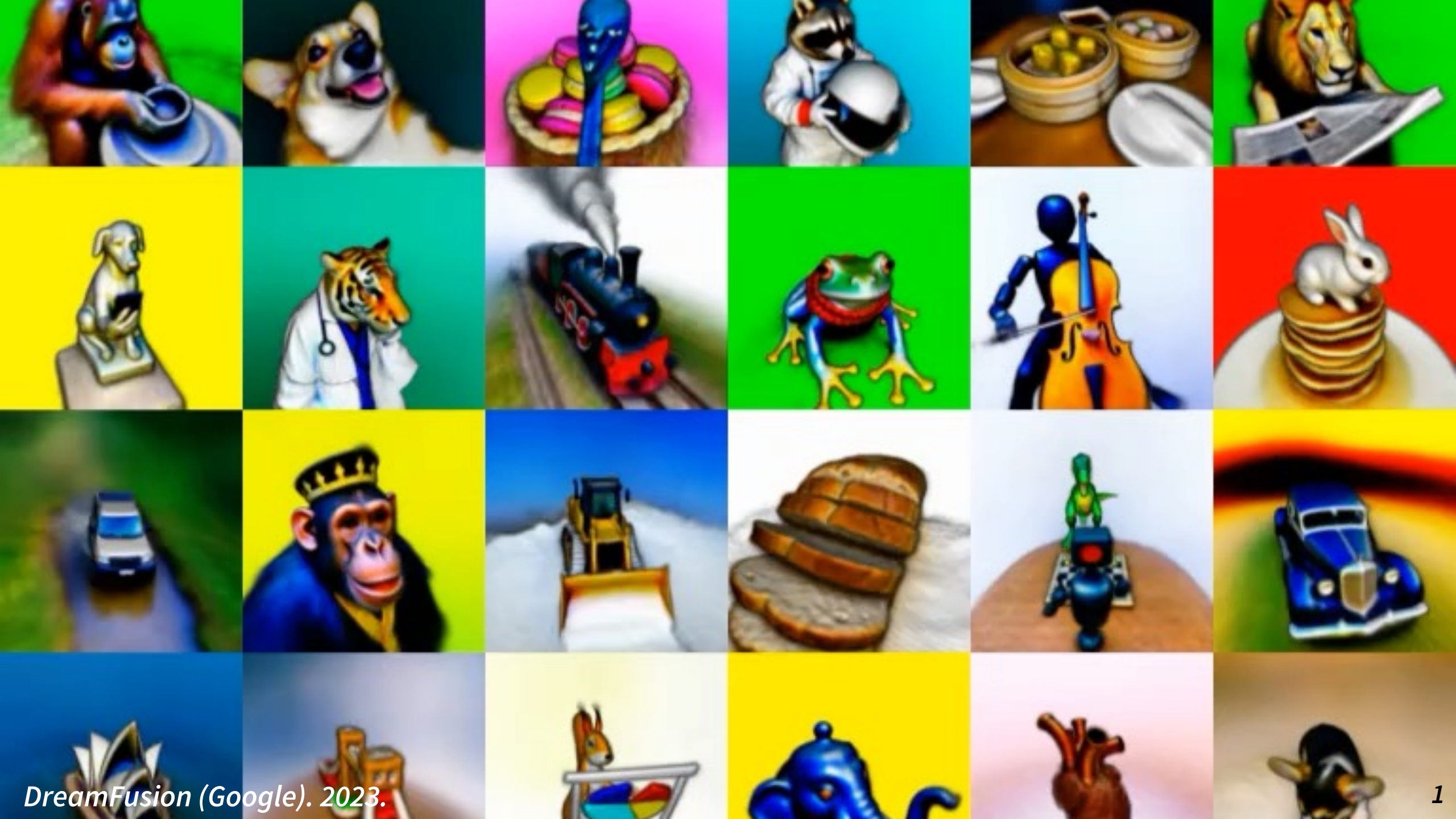
Blattmann et al., Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets, arXiv'23

Diffusion Models for Visual Content Generation

3D Generation

Presenter: Minhyuk Sung

Eurographics 2024 Tutorial



DreamFusion (Google). 2023.

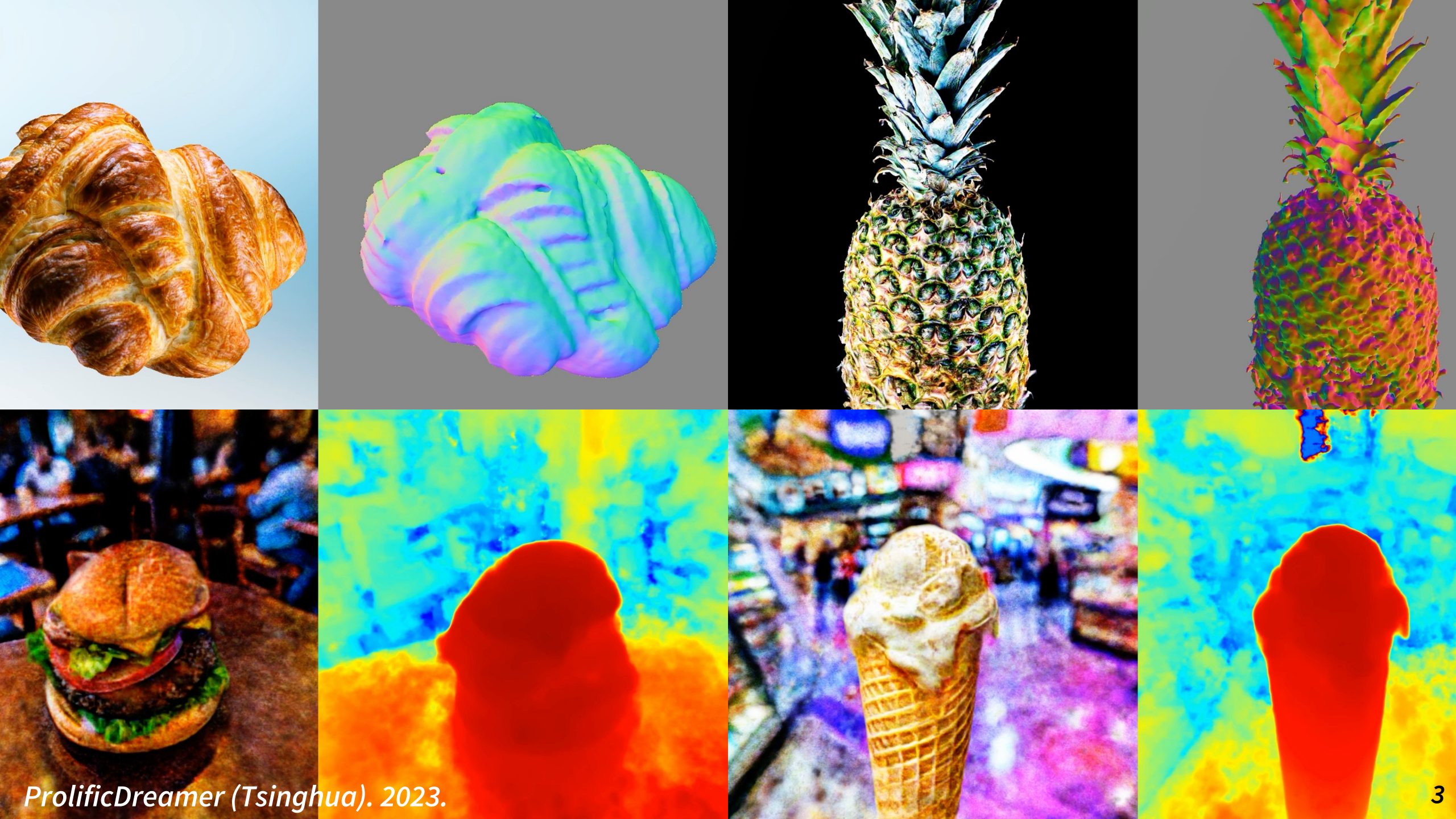
Magic3D : High-Resolution Text-to-3D Content Creation

Chen-Hsuan Lin* Jun Gao* Luming Tang* Towaki Takikawa* Xiaohui Zeng*
Xun Huang Karsten Kreis Sanja Fidler# Ming-Yu Liu# Tsung-Yi Lin

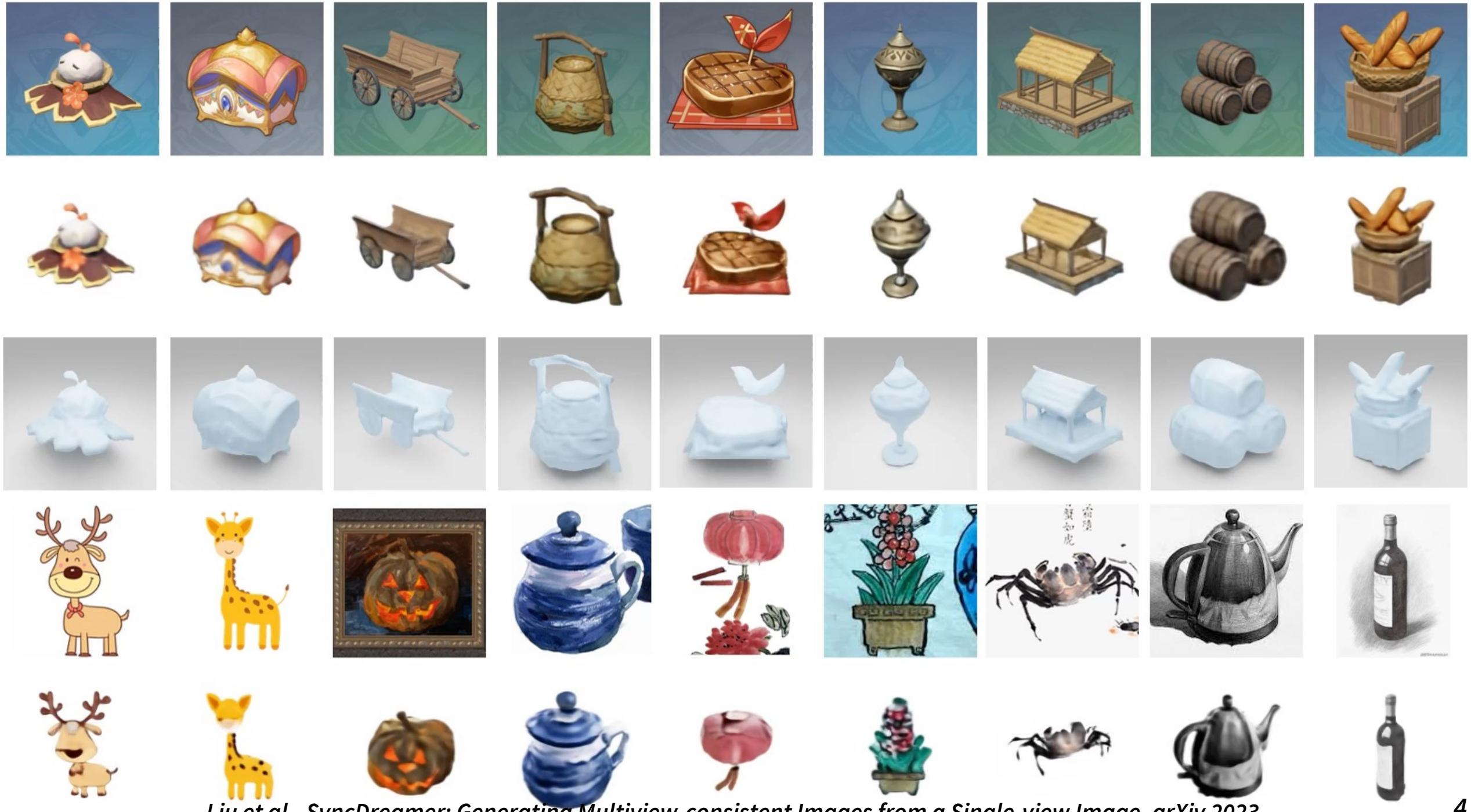
*# : equal contributions

NVIDIA Corporation

Magic3D (NVIDIA). 2023.



ProlificDreamer (Tsinghua). 2023.



Liu et al., SyncDreamer: Generating Multiview-consistent Images from a Single-view Image, arXiv 2023.

Zero123-XL

OpenLRM

Stable Zero123

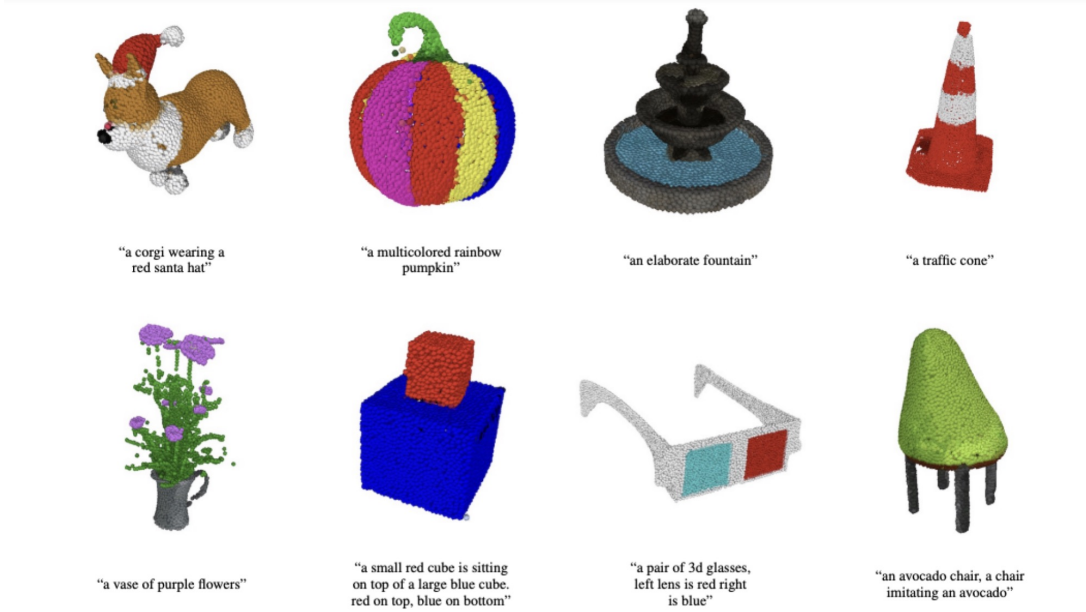
TripoSR (ours)

3D Diffusion Models

3D Diffusion Models

Point clouds

- *ShapeGF (Cai et al., 2020)*
- *DPM (Luo and Hu, 2021)*
- *LION (Nichol et al., 2022)*

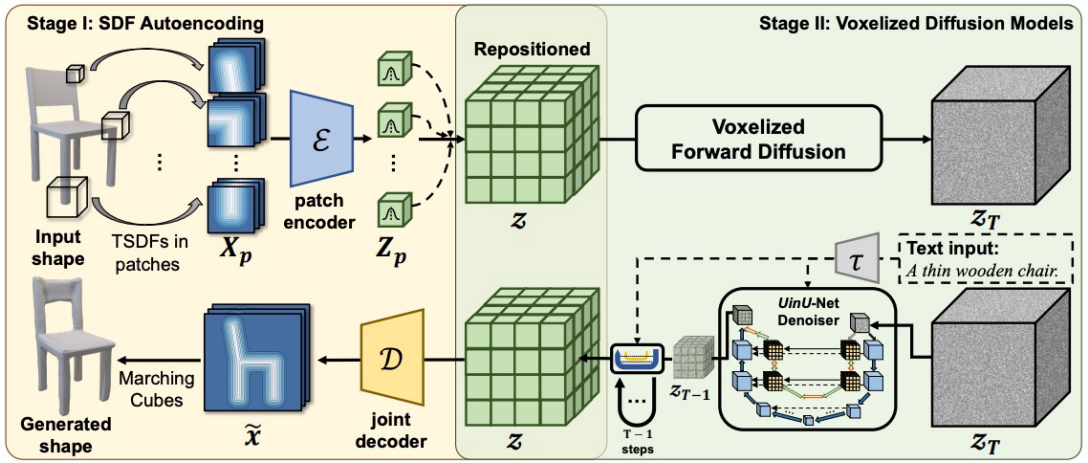


LION, Nichol et al. 2022.

3D Diffusion Models

Voxel representation

- *PVD (Zhou et al., 2022)*
- *Diffusion-SDF (Li et al, 2022)*



Diffusion-SDF, Li et al., 2022.

3D Diffusion Models

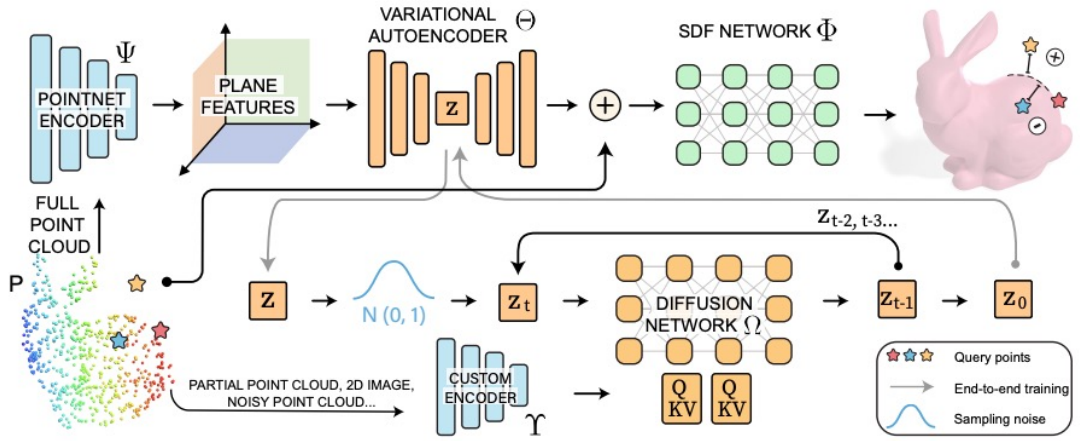
Latent representation

- *SDFusion*

  (Cheng et al., 2022)

- *DiffusionSDF (w/o hyphen,*

  Chou et al., 2023)



DiffusionSDF, ***Chou et al., 2023***

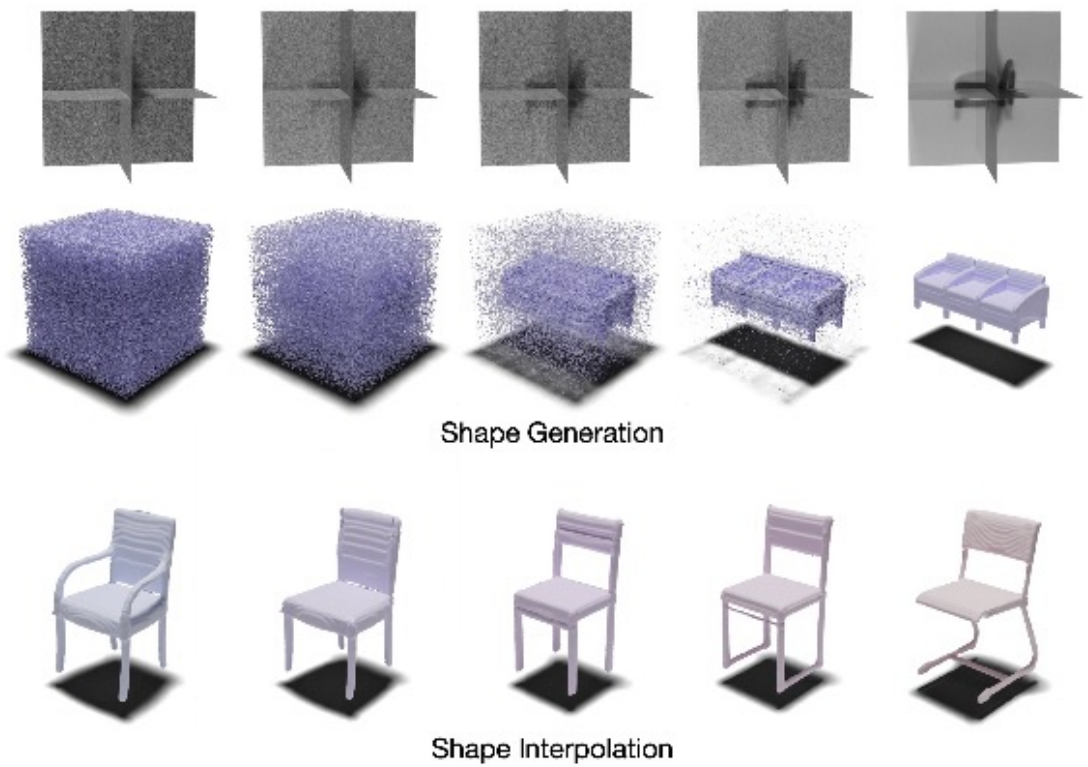
3D Diffusion Models

Triplane representation

- *NFD (Shue et al., 2022)*

Diffusion in the spectral domain:

- *NeuralWavelet (Hui et al., 2022)*



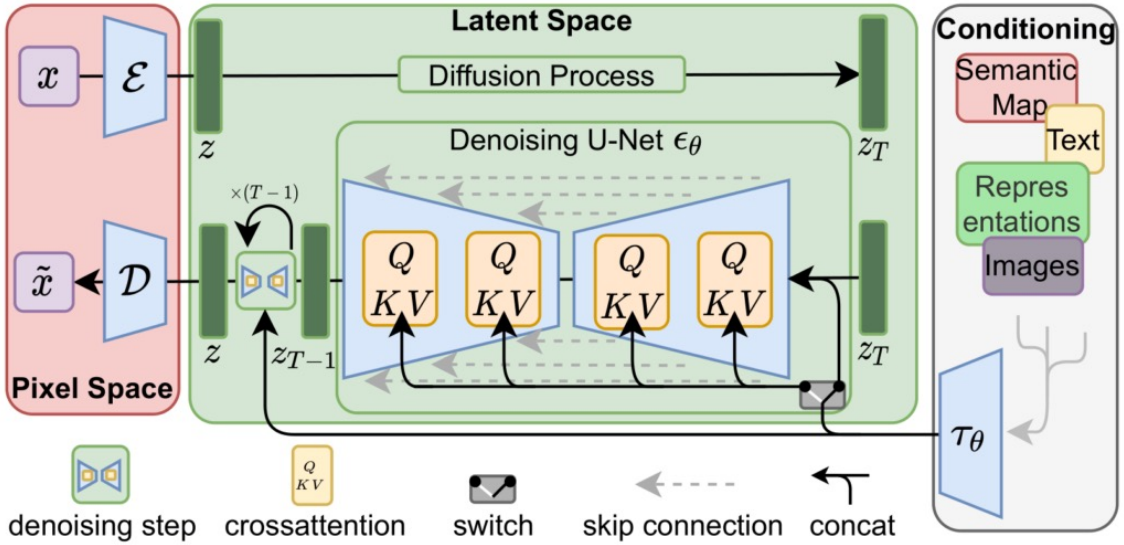
NFD, Shue et al., 2022.

Diffusion w/ Different Representations

- *Implicit representation*
- *Explicit representation*

Diffusion w/ Different Representations

- *Implicit representation (I.e., latent features)*
  - *E.g., Latent diffusion (Rombach et al., 2022)*
  - *(+) Best quality of the generated data.*
  - *(–) Requires retraining for each conditional generation setup.*



Latent Diffusion, Rombach et al., 2022.

Diffusion w/ Different Representations

- *Explicit representation (E.g., pixels in images)*
  - *E.g.: The original DDPM model (Ho et al., 2020) for images.*
  - *(−) Suboptimal performance due to the high dimensionality.*
  - *(−) Cannot change the resolution of the data.*
  - *(+) Can be directly leveraged in conditional generation setups in a zero-shot manner.*

SDEdit [Meng et al., 2022]

Image editing using a pretrained pixel-space diffusion model.

$$\mathbf{x}^{(0)} = m \odot \mathbf{x}_a^{(0)} + (1 - m) \odot \mathbf{x}_b^{(0)}$$

$$\mathbf{x}^{(t)} = \text{denoise}\left(\mathbf{x}^{(0)}, t\right)$$

$$\mathbf{x}'^{(0)} = \text{add\_noise}\left(\mathbf{x}^{(t)}, t\right)$$

RePaint [Lugmayr et al., 2022]

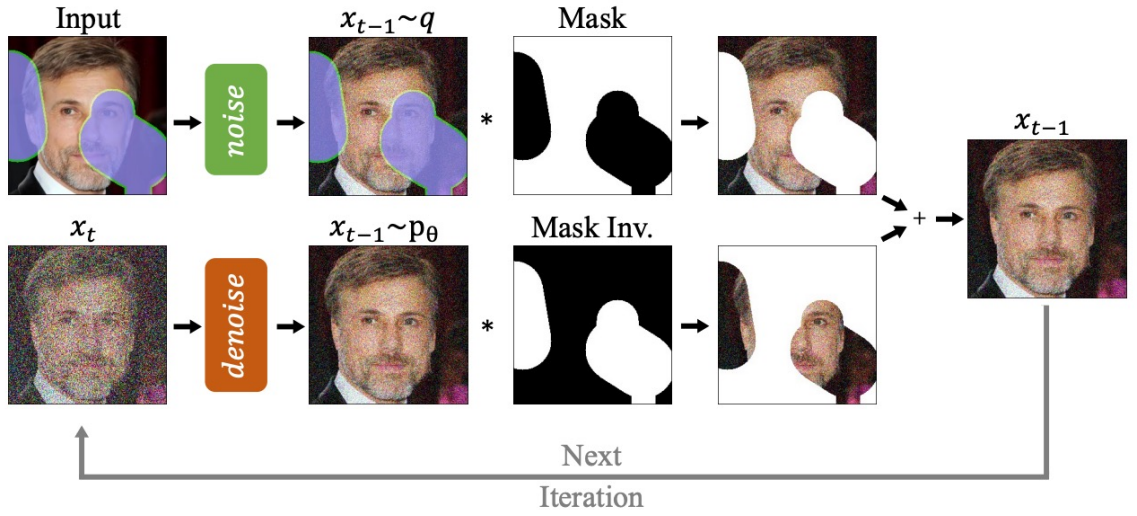
Image inpainting using a pretrained pixel-space diffusion model.

$$\mathbf{x}_f^{(t-1)} = \text{denoise}\left(\mathbf{x}^{(t)}, 1\right)$$

$$\mathbf{x}_b^{(t-1)} = \text{add\_noise}\left(\mathbf{x}^{(0)}, t\right)$$

$$\mathbf{x}^{(t-1)} = m \odot \mathbf{x}_f^{(t-1)} + (1-m) \odot \mathbf{x}_b^{(t-1)}$$

Repeat for $t = T, \cdots, 1.$



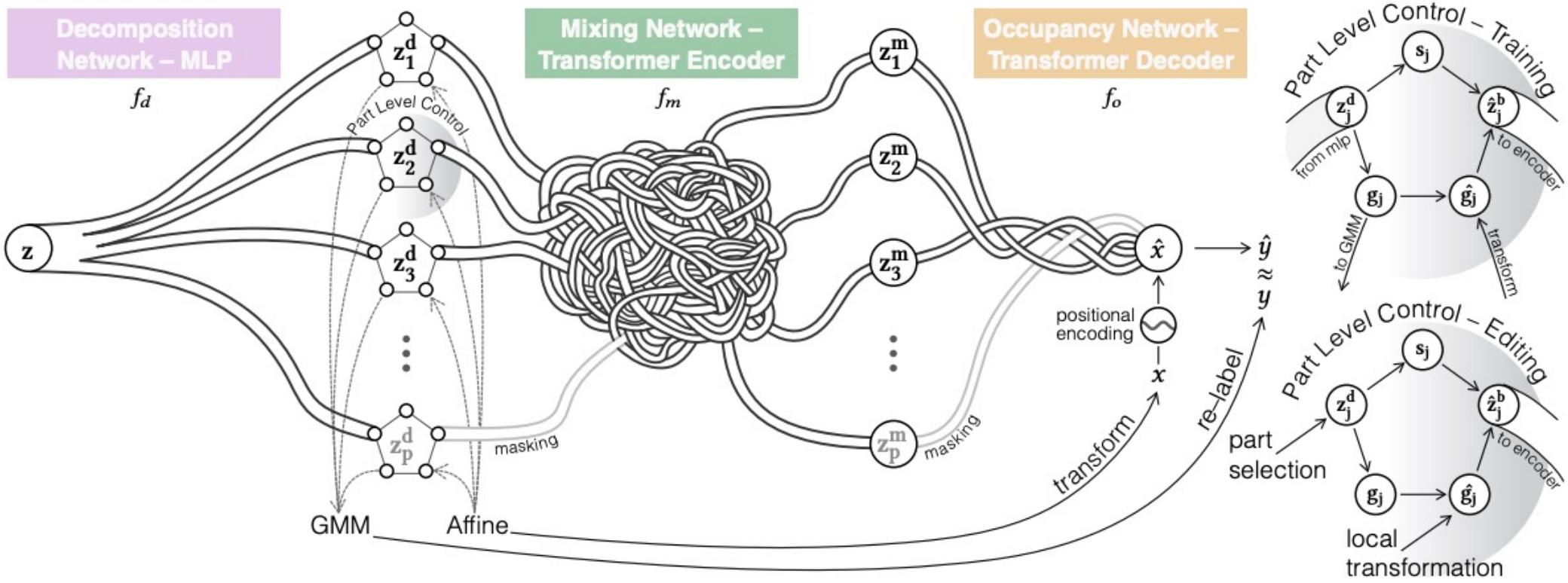
Hybrid Representation

Leverages a novel hybrid representation describing

- global part-level structure explicitly, and
- local geometry implicitly.

SPAGHETTI [Hertz et al., 2022]

The part decomposition and the explicit/implicit representations are learned in an unsupervised way.



Part-Level Representation

For each part of an object learned in an unsupervised way,

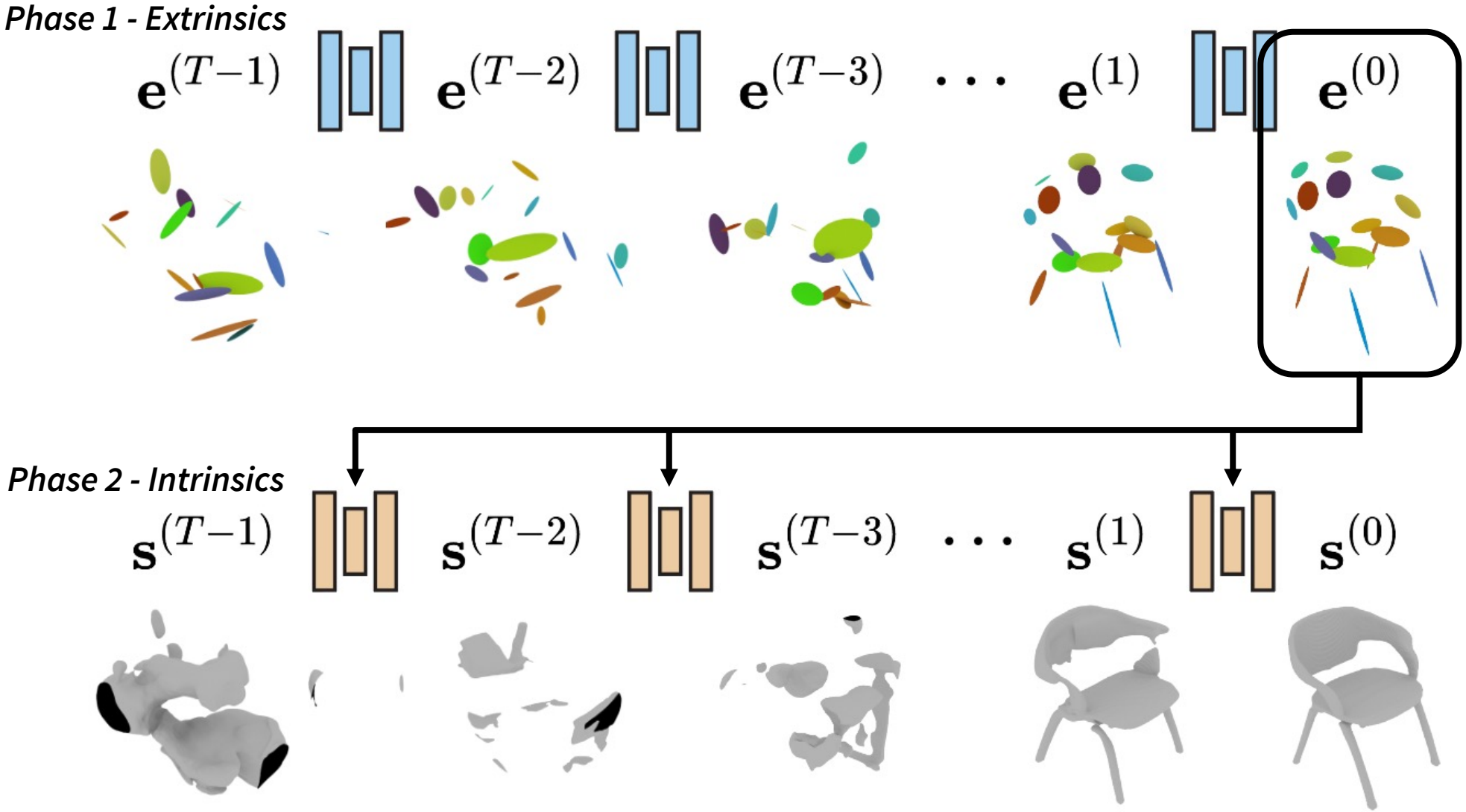
- *Explicit parameters of Gaussian blubs indicate position, scale, and rotation.*

Part-Level Representation

For each part of an object learned in an unsupervised way,

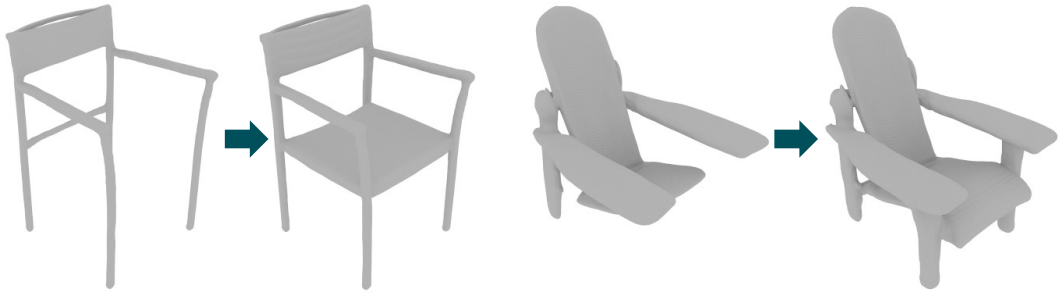
- *Implicit latent feature is decoded into an occupancy function.*

Two-Phase Cascaded Diffusion

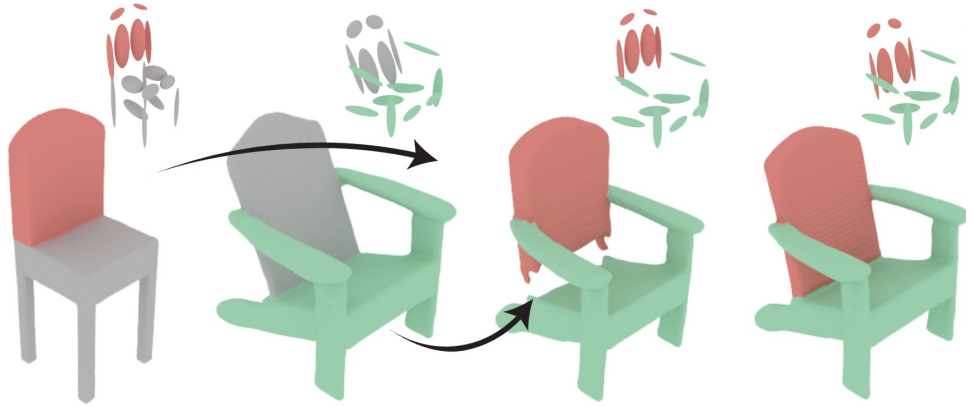


Diversity of 3D Shapes

Applications



Part Completion

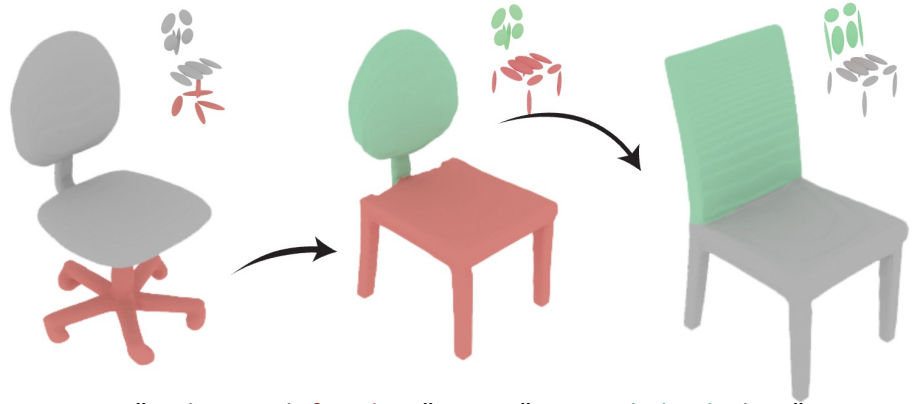


Part Mixing

”Chair has round arms and wheels."

”Its the one with gaps in the back."

Text-to-3D Generation



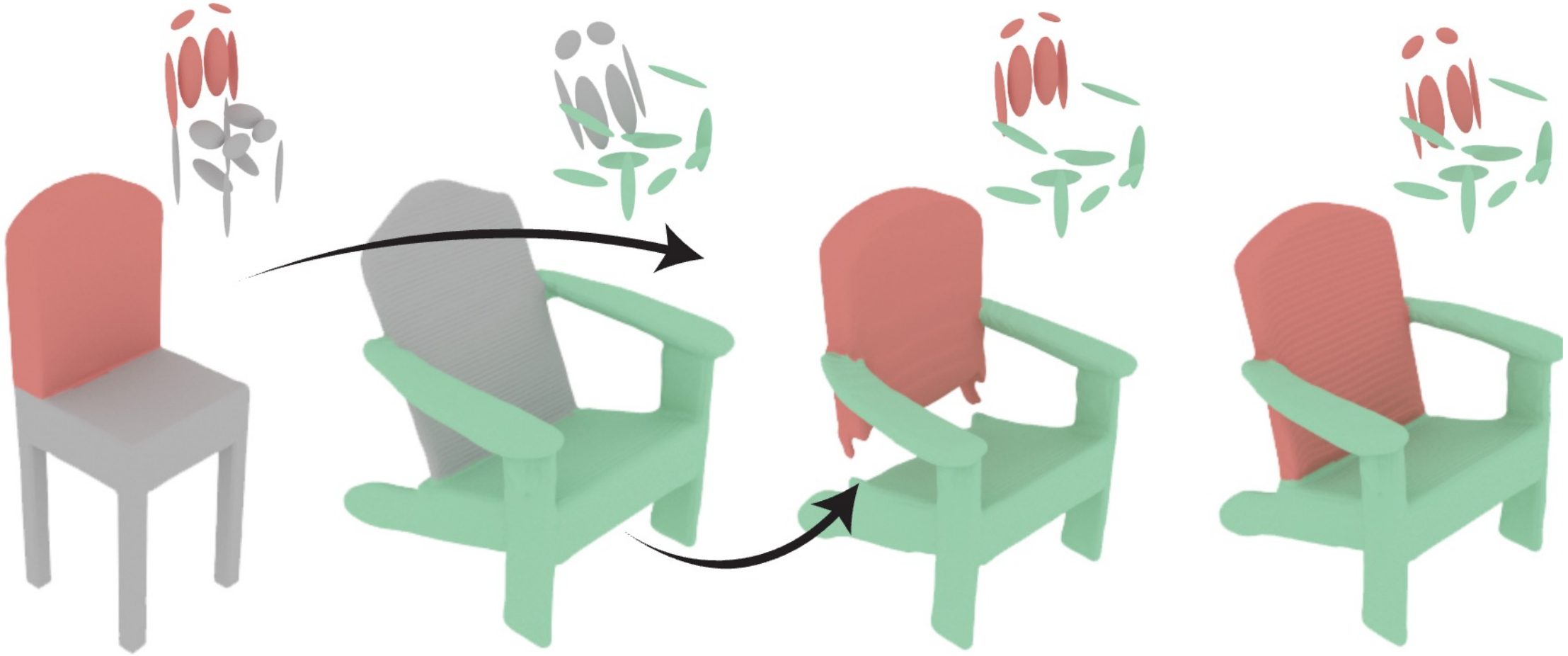
"A chair with four legs"

"rectangle back chair"

Text-Guided Part Editing

Part Mixing

$$\mathbf{x}^{(0)} = m \odot \mathbf{x}_a^{(0)} + (1 - m) \odot \mathbf{x}_b^{(0)}$$
$$\mathbf{x}^{(t)} = \text{denoise}\left(\mathbf{x}^{(0)}, t\right)$$
$$\mathbf{x}'^{(0)} = \text{add\_noise}\left(\mathbf{x}^{(t)}, t\right)$$

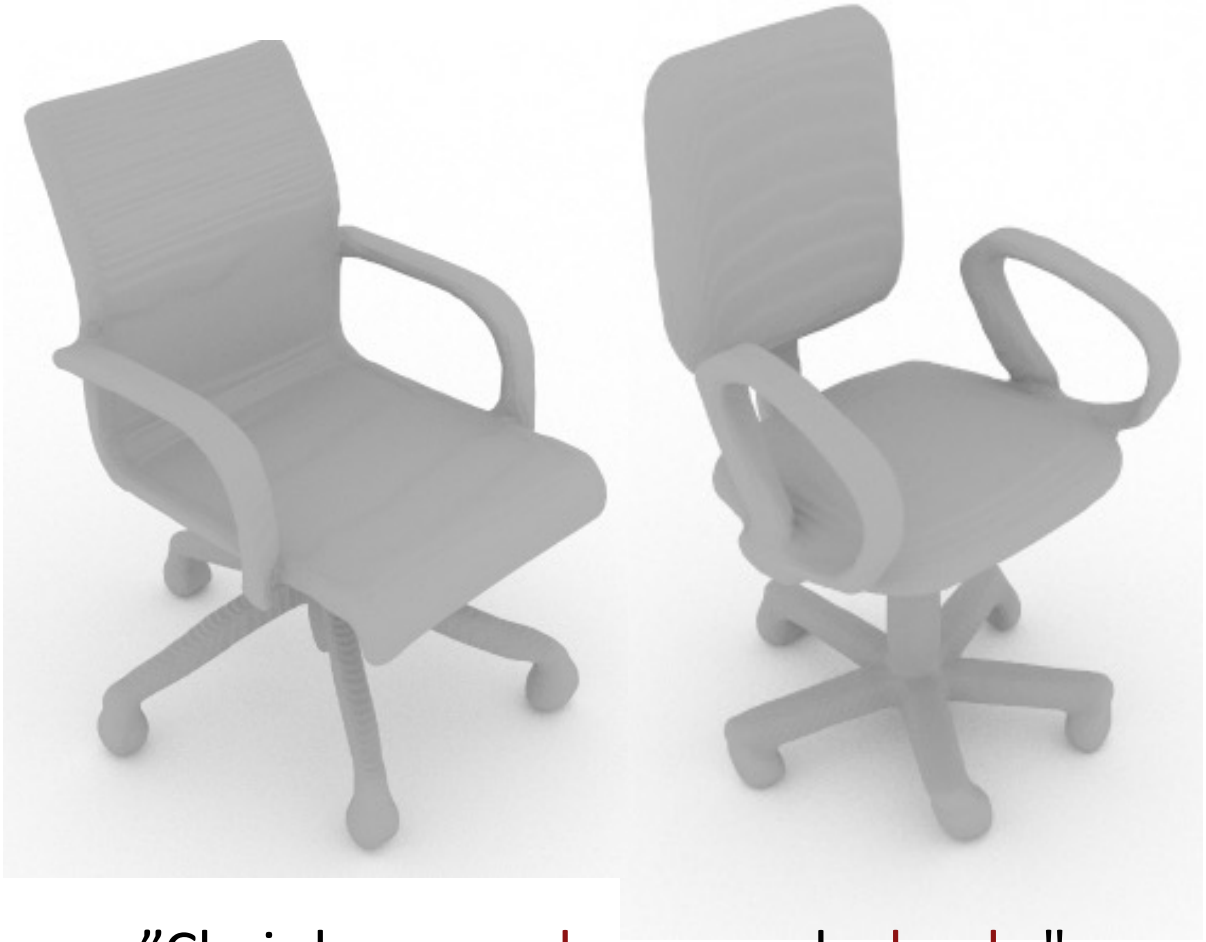


Part Completion

$$\mathbf{x}_f^{(t-1)} = \text{denoise}\left(\mathbf{x}^{(t)}, 1\right)$$
$$\mathbf{x}_b^{(t-1)} = \text{add\_noise}\left(\mathbf{x}^{(0)}, t\right)$$
$$\mathbf{x}^{(t-1)} = m \odot \mathbf{x}_f^{(t-1)} + (1-m) \odot \mathbf{x}_b^{(t-1)}$$

Repeat for $t = T, \cdots, 1$.

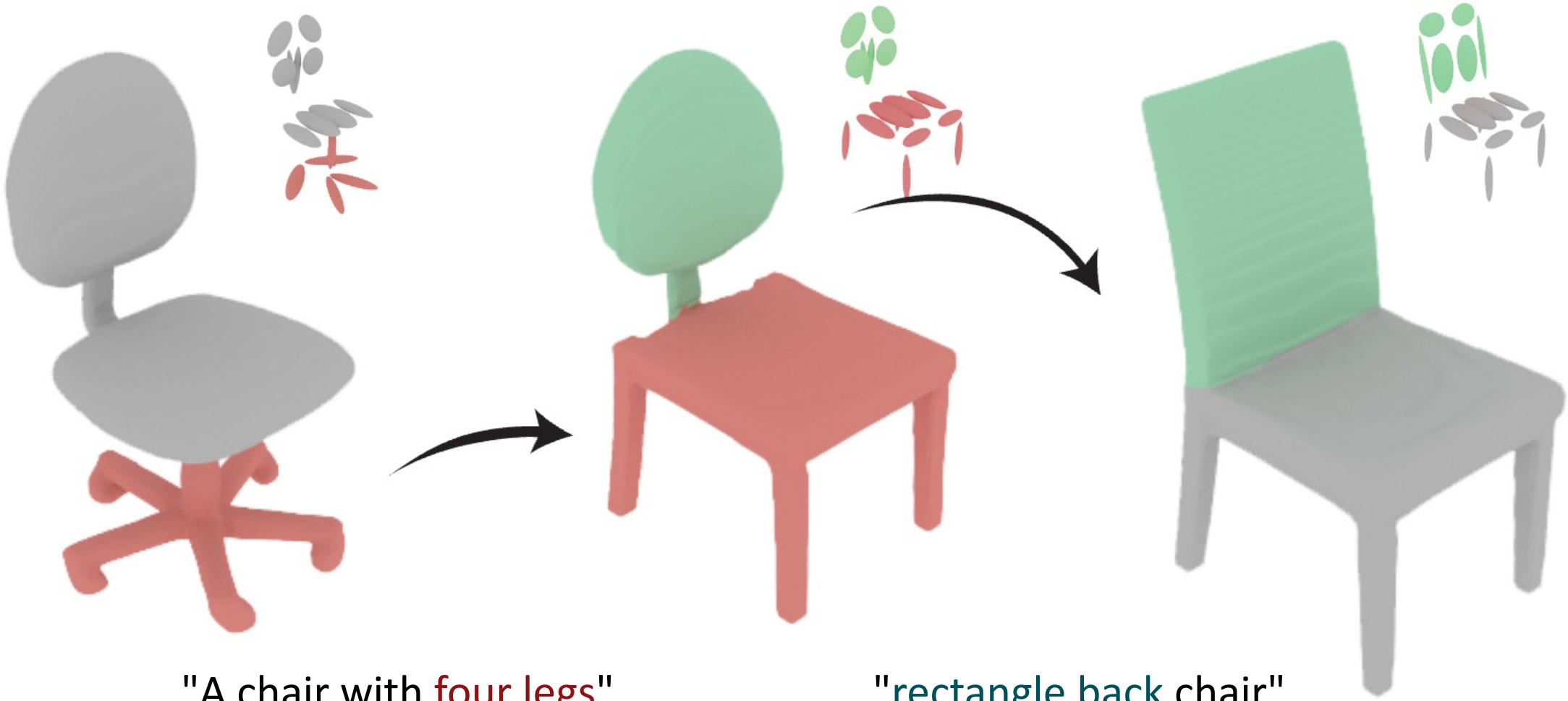
Text-to-3D Generation



”Chair has round arms and wheels."

”Its the one with gaps in the back."

Text-Guided Part Editing



"A chair with four legs"

"rectangle back chair"

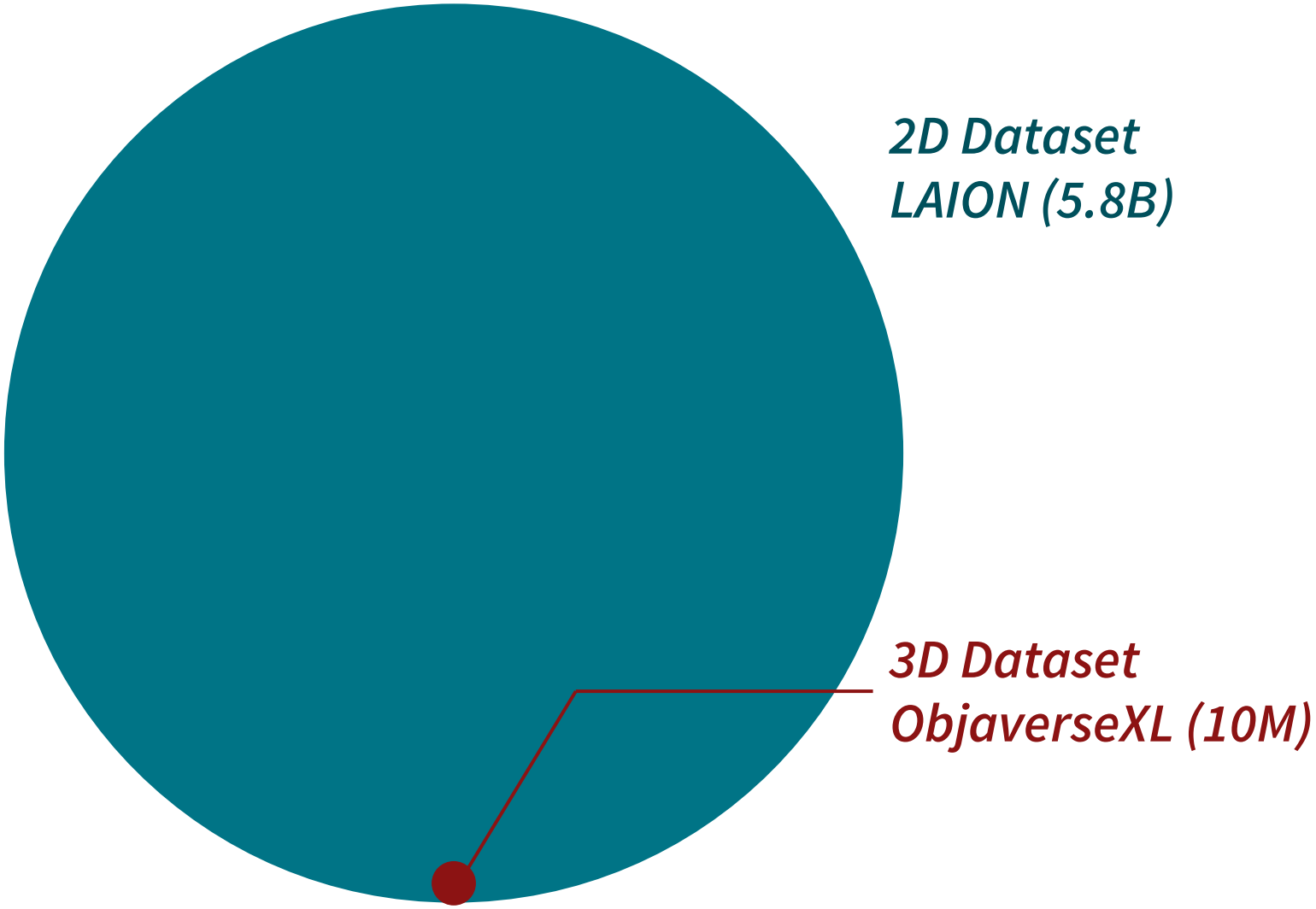
Limitation?

Koo et al., SALAD: Part-Level Latent Diffusion for 3D Shape Generation and Manipulation, ICCV 2023.

Challenge: Lack of Large-Scale 3D Dataset

Objaverse-XL
Allen Institute

Data Deficiency



Diversity of Imaginable 3D Shapes

"frog wearing a sweater"

"eggshell broken in two with an adorable chick standing next to it"

"ghost eating a hamburger"

"a pig wearing a backpack"

We have a small-scale 3D dataset

but images on an internet scale.

Images are projections of 3D from specific angles.

https://d2jv9003bew7ag.cloudfront.net/uploads/Tim-Noble-and-Sue-Webster-Wild-Mood-Swings-2009.jpg

3D Generation

How to generate a 3D object

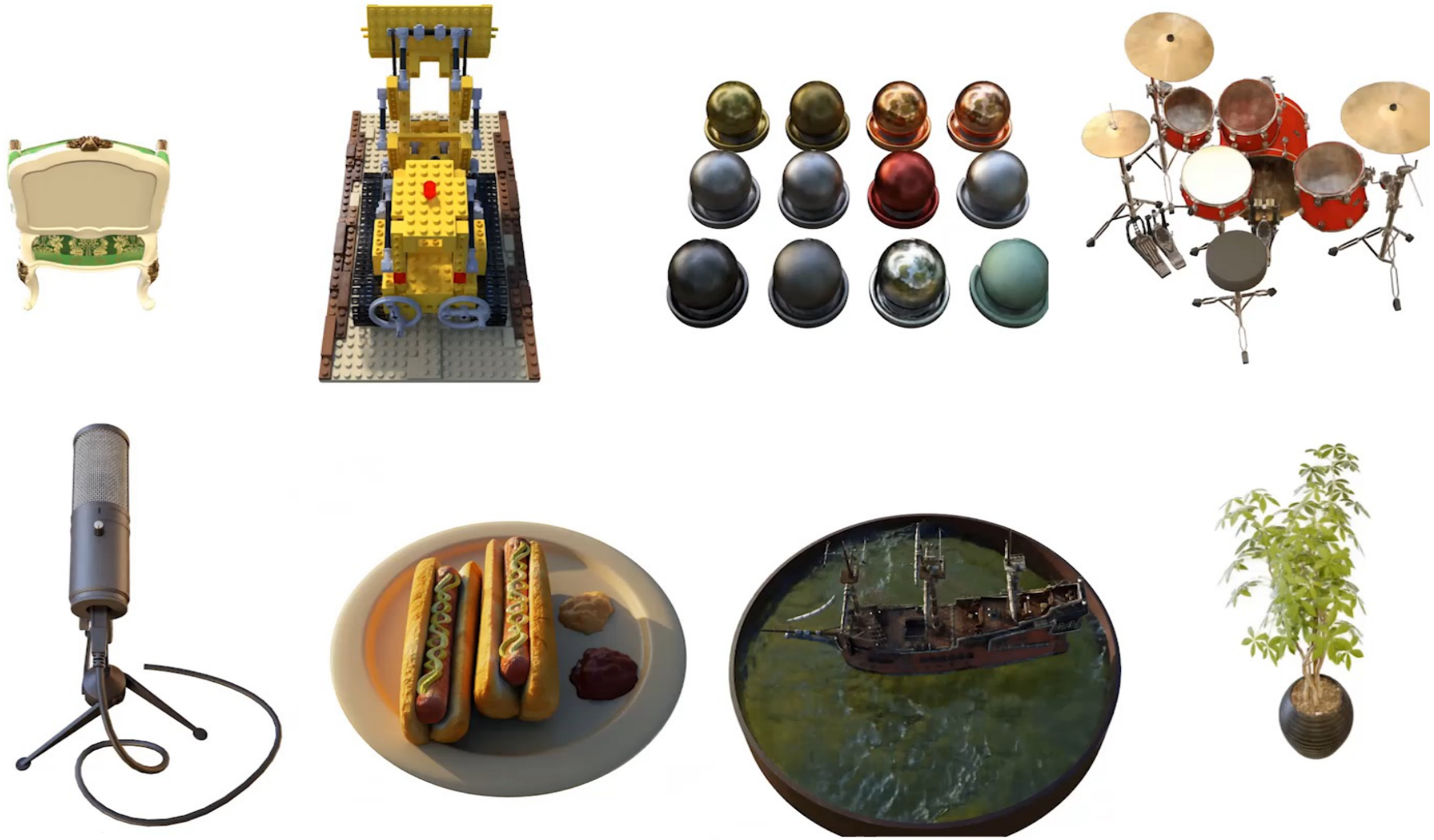
from a collection of 2D images?

3D Reconstruction

How to reconstruct a 3D object from a collection of 2D images of a specific object?

Mildenhall et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020.

NeRF



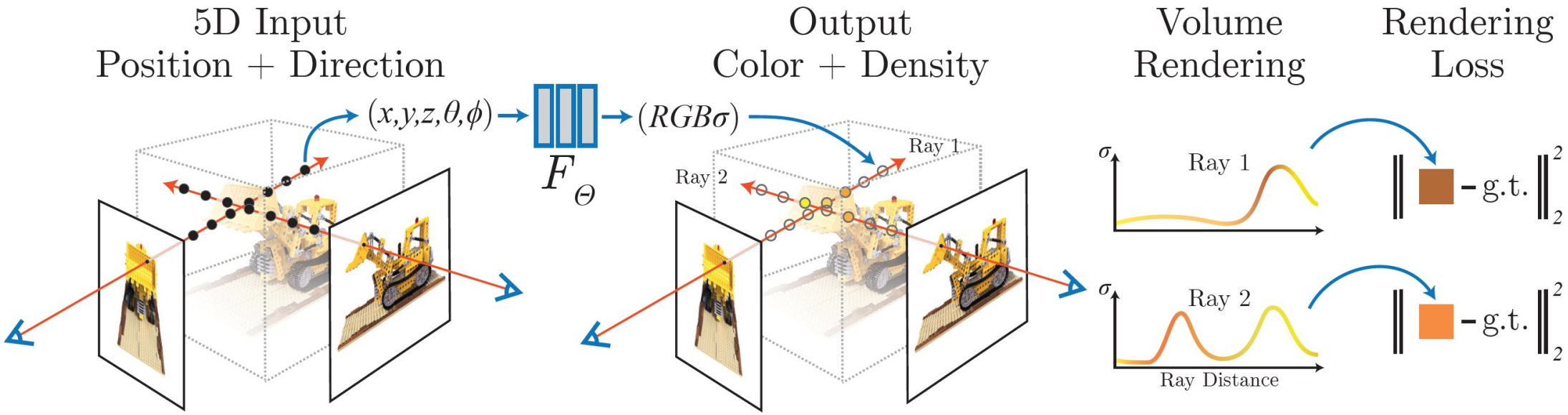
3D Reconstruction

- Input: A set of images with camera poses.
- Output: A representation of the 3D object.


Menu Views Capture
3D Gaussians
Camera Point view
Kerbl et al., 3D Gaussian Splatting for Real-Time Radiance Field Rendering, SIGGRAPH 2023.
Metrics
59.97 (16.67 ms)
Gaussian Splatting

NeRF Optimization

1. *Render* a NeRF representation into a specific view.
2. Compute the *difference* with the *given image*.
3. Update the NeRF using *gradient descent*.



Can we perform NeRF reconstruction

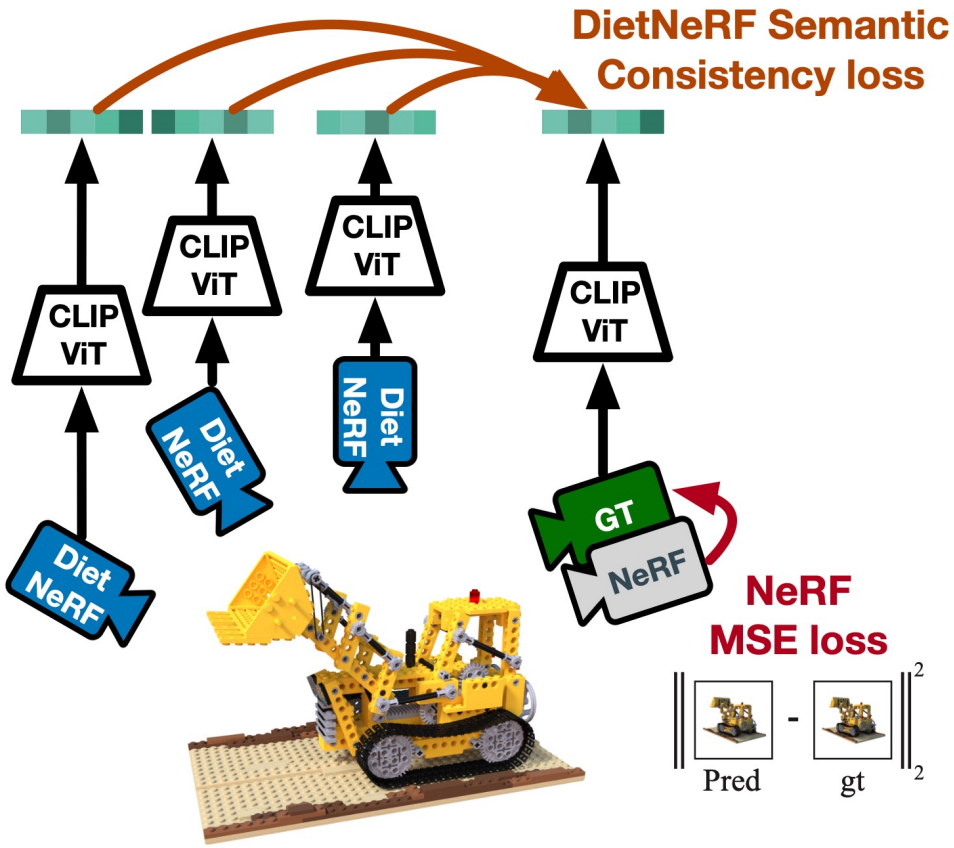
with a few images?

Then, we need additional information!

Few-Shot NeRF

***DietNeRF** [Jain et al., ICCV 2021]*

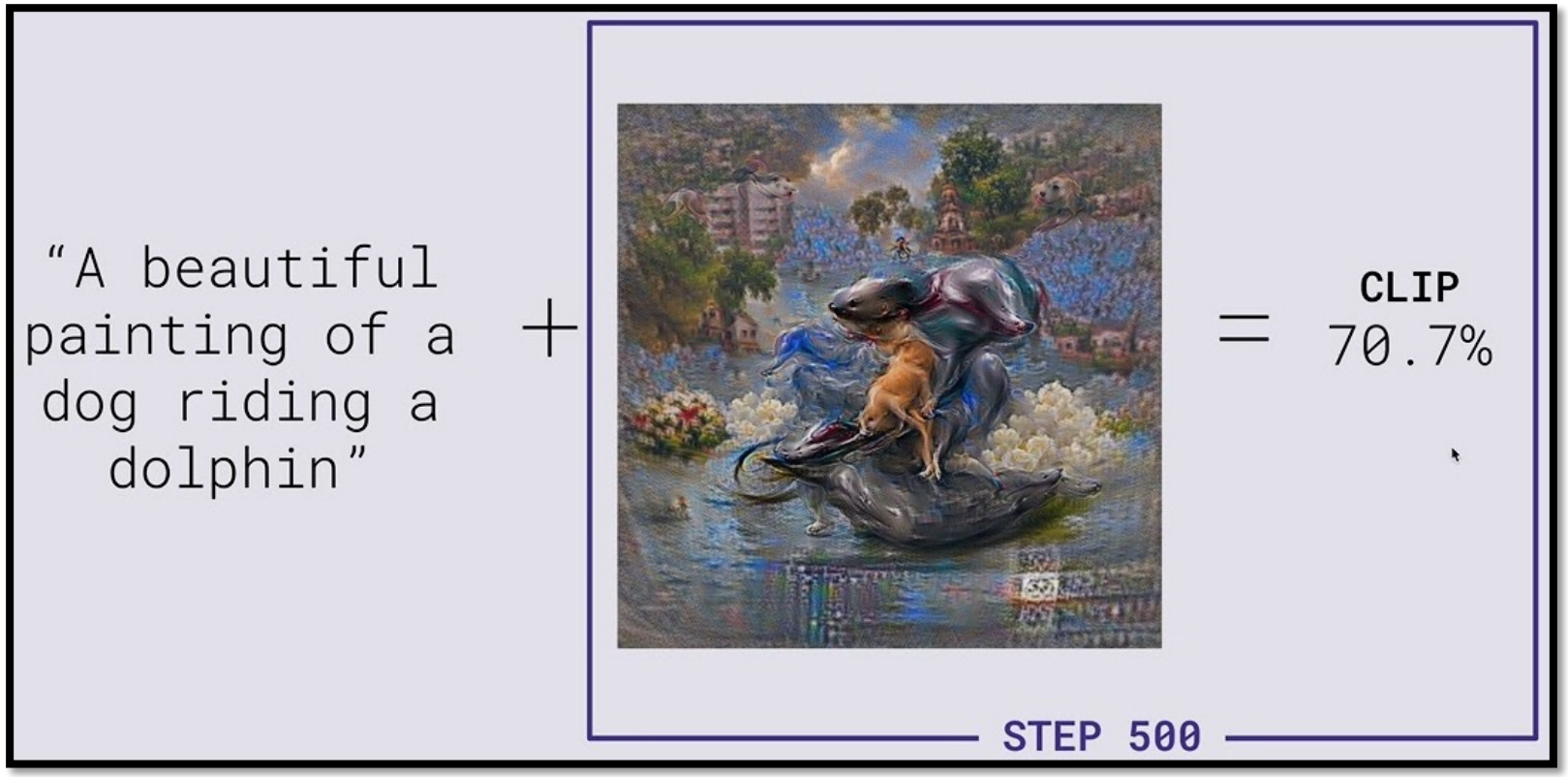
Use priors learned by CLIP** [Radford et al., 2021], **a text-image model.



"a bulldozer is a bulldozer from any perspective"

CLIP [Radford et al., 2021]

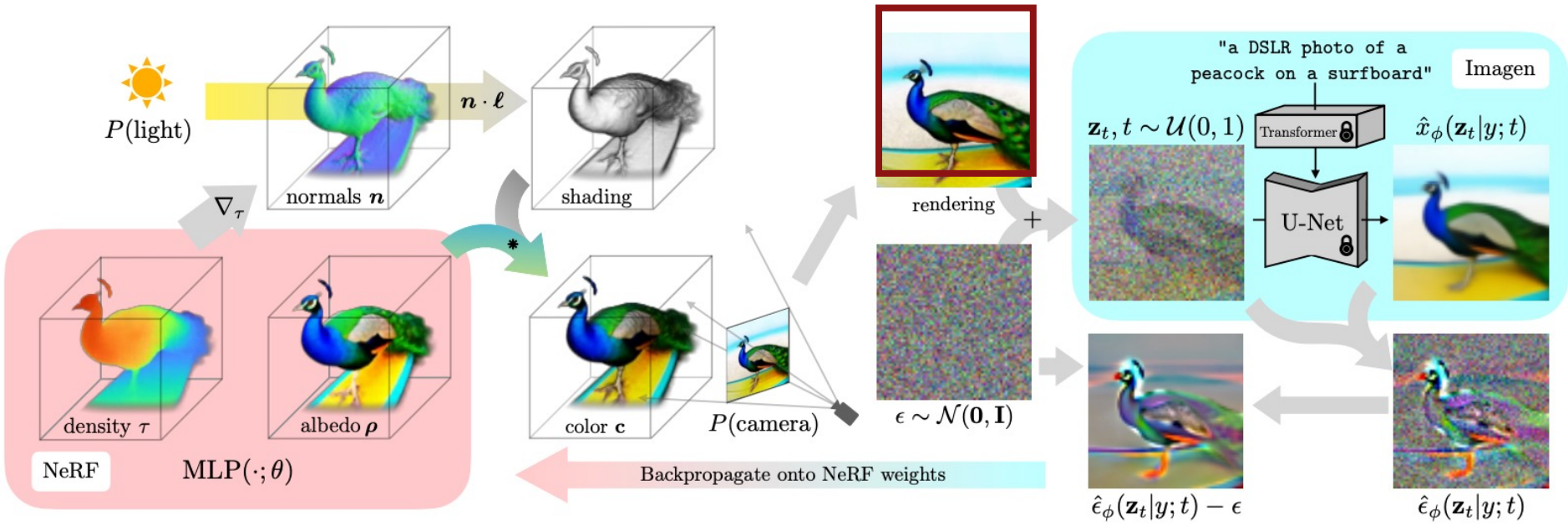
CLIP takes a text-image pair as input and assesses the *alignment* between the text and the image.



https://www.youtube.com/watch?app=desktop&v=-b7xKWeADHQ

Knowledge Distillation in 3D Generation

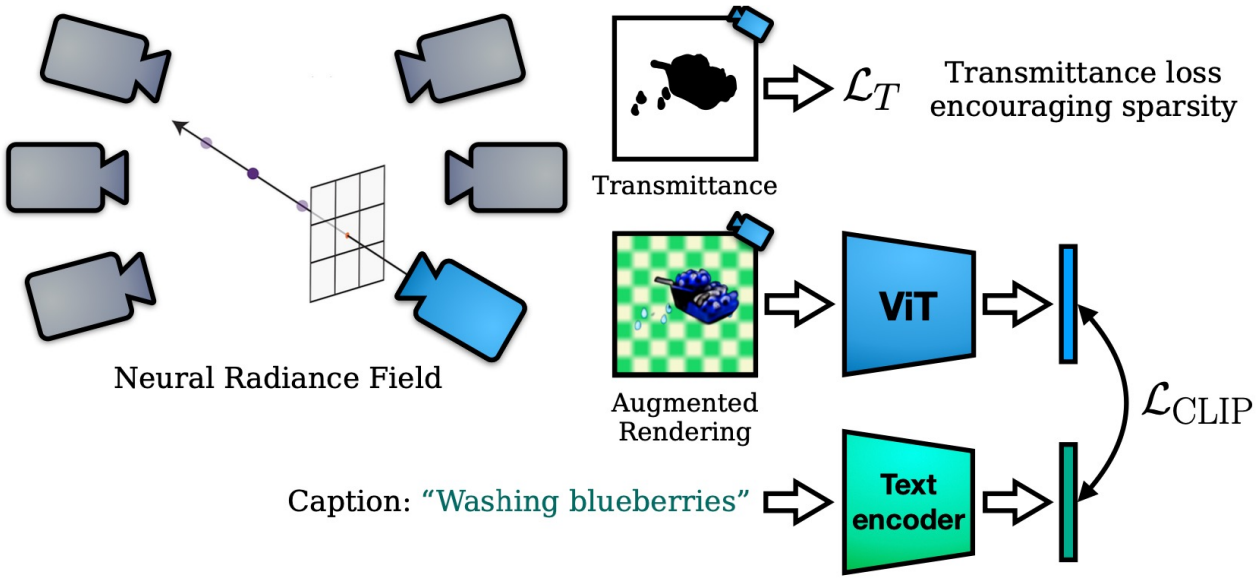
1. *Render* the NeRF representation into a specific view
2. Compute the *alignment* to the given text.
3. Update the NeRF using the *gradient descent*.

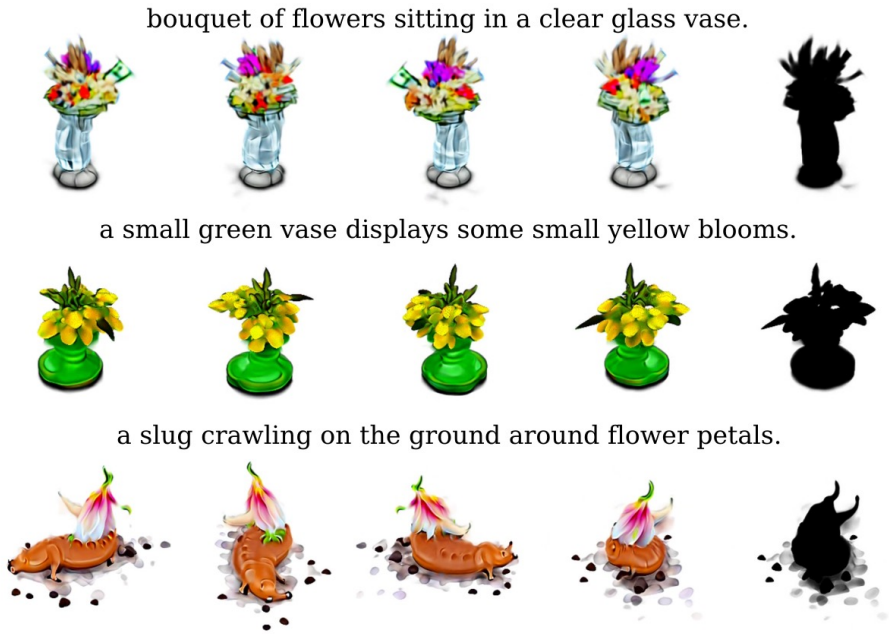


Extrem Case: Zero-Shot NeRF

***DreamFields** [Jain et al., CVPR 2022]*

Given a text prompt but no images, generate a 3D shape by maximizing similarity between a rendered image and the input prompt in the CLIP embedding space.





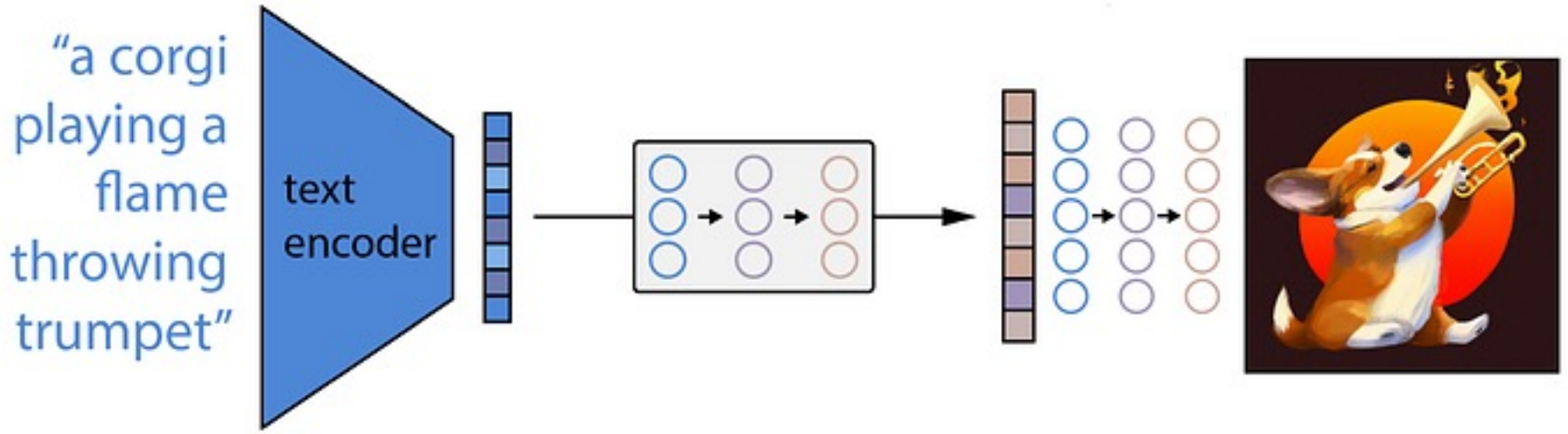
DreamFields [Jain et al., CVPR 2022]

The first 3D generative model using 2D priors!

an archair in the shape of a ____.
an archair imitating a ____.

a teapot in the shape of a ____.
a teapot imitating a ____.

Can we use an image diffusion model instead of CLIP?

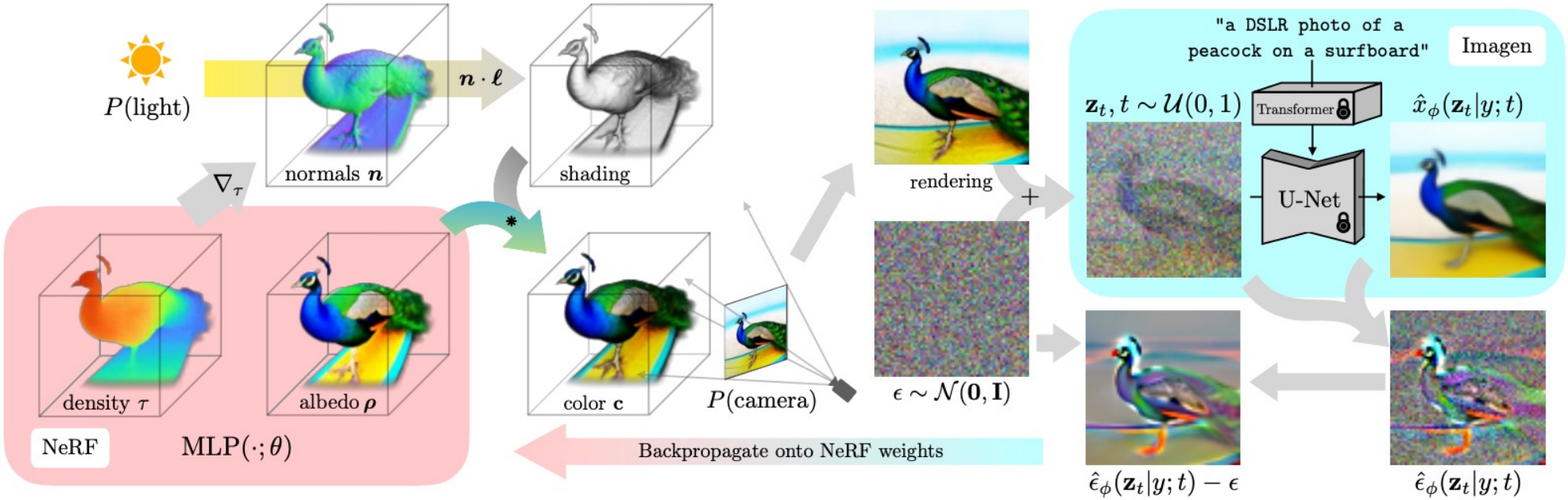


https://medium.com/latinxinai/text-to-image-with-stable-diffusion-4df16da2cfd5

Score Distillation Sampling (SDS)

DreamFusion [Poole et al., ICLR 2023]

Proposed the idea of Score Distillation Sampling (SDS), leveraging a pretrained diffusion model to measure the plausibility of rendered images.



Score Distillation Sampling (SDS)

- How can we utilize a pretrained diffusion model to measure the plausibility of rendered images?

- Review the loss function:

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

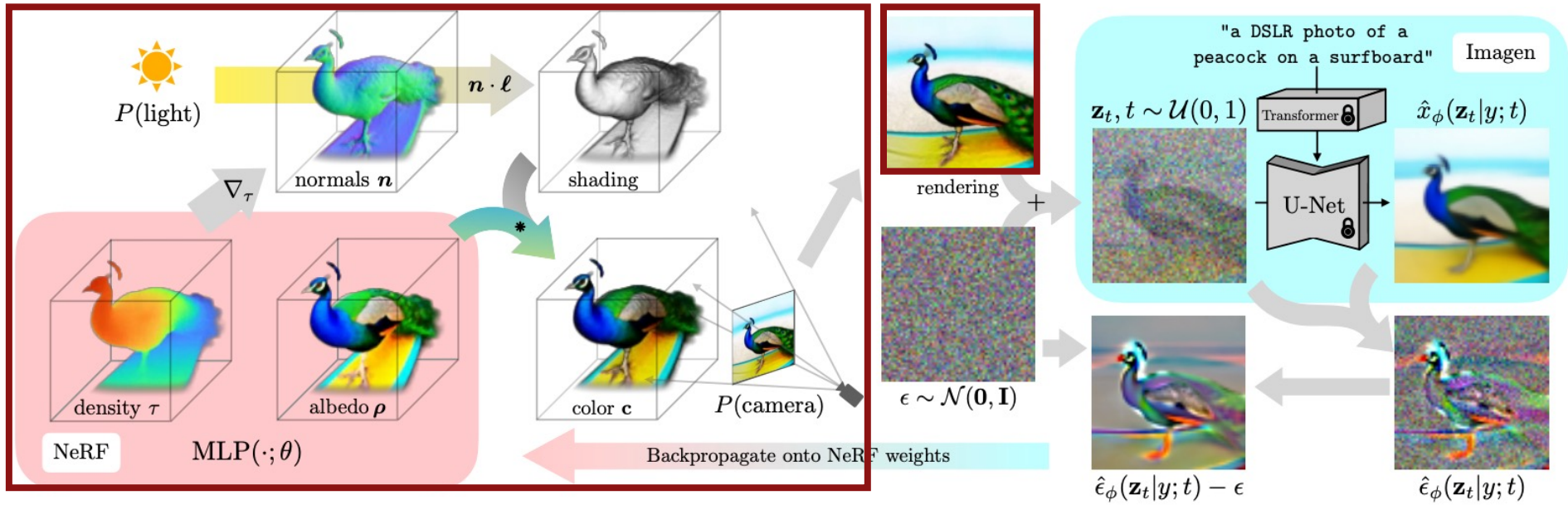
If the training of the diffusion model has converged, the loss for real data $\mathbf{x}_0$ will be close to zero.

Score Distillation Sampling (SDS)

1. *Render* the NeRF representation into a specific view.

   Let ϕ denote the NeRF parameter, and

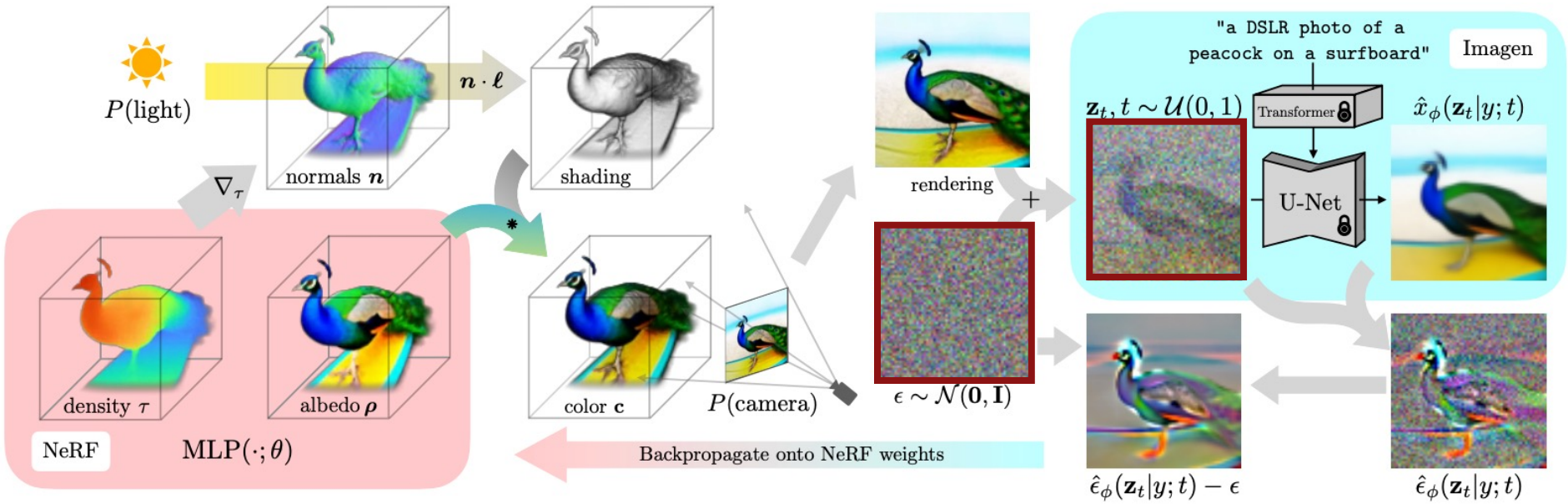
   $\mathbf{x}_0 = g(\phi)$ denote the rendered image.



Score Distillation Sampling (SDS)

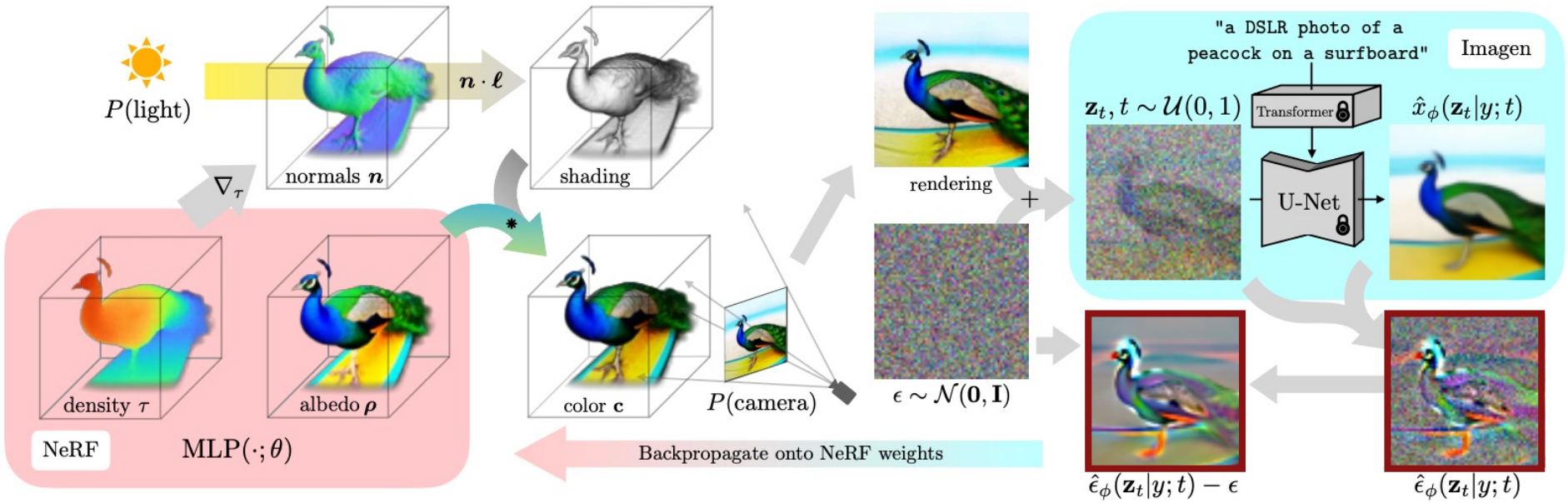
2. Add noise to the rendered image $\mathbf{x}_0$:

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



Score Distillation Sampling (SDS)

3. Perform gradient descent on $\mathcal{L}$ with respect to the NeRF parameters ϕ.



DreamFusion Results

"frog wearing a sweater"

"eggshell broken in two with an adorable chick standing next to it"

"ghost eating a hamburger"

"a pig wearing a backpack"

Why SDS Instead of Reverse Diffusion?

This is a scenario where the images are parameterized differently from how they were represented during the training of the diffusion model.

- *Training: Per-pixel colors.*
- *Inference: NeRF rendering.*

Jain et al., VectorFusion: Text-to-SVG by Abstracting Pixel-Based Diffusion Models, CVPR 2023.

Example: Vector Images / Sketches

The same idea but with a different parameterization of images.

Example: Mesh Editing

The same idea but with a different parameterization of images.

Source

ARAP
[Sorkine and Alexa, 2007]

APAP
[Yoo et al., 2024]

Stable-DreamFusion

Stable-Dreamfusion

A pytorch implementation of the text-to-3D model **Dreamfusion**, powered by the Stable Diffusion text-to-2D model.

ADVERTISEMENT: Please check out threestudio for recent improvements and better implementation in 3D content generation!

NEWS (2023.6.12):

- Support of Perp-Neg to alleviate multi-head problem in Text-to-3D.
- Support of Perp-Neg for both Stable Diffusion and DeepFloyd-IF.

Text-to-3D

a rabbit, animated movie character, high detail 3d model

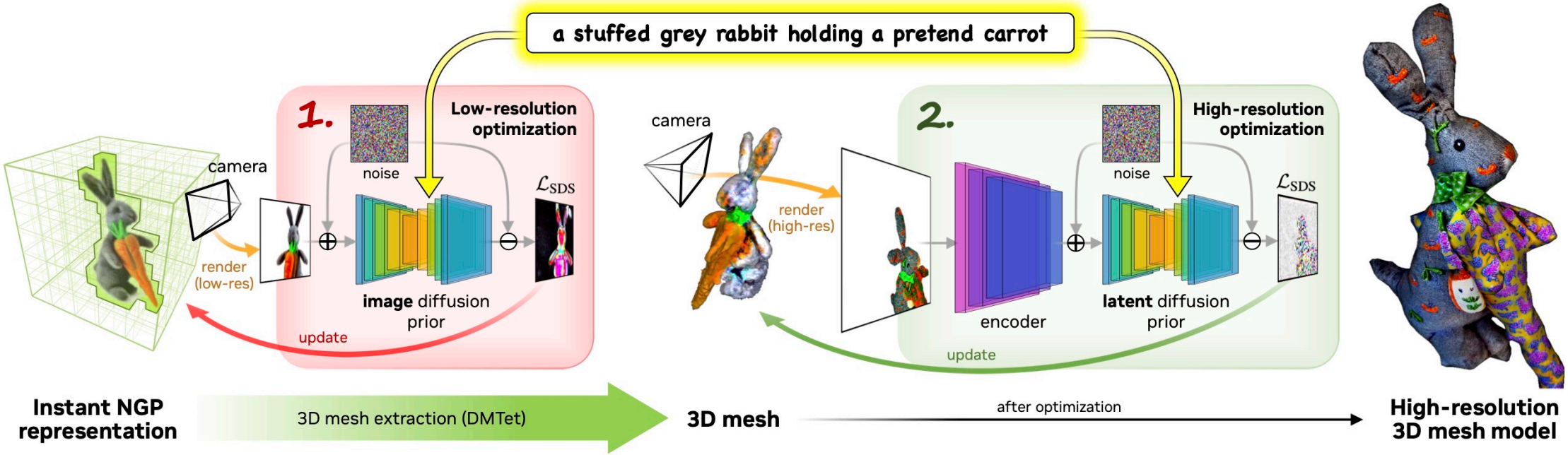
a DSLR photo of a delicious hamburger

a highly detailed stone bust of Theodoros Kolokotronis

a small saguaro cactus planted in a clay pot

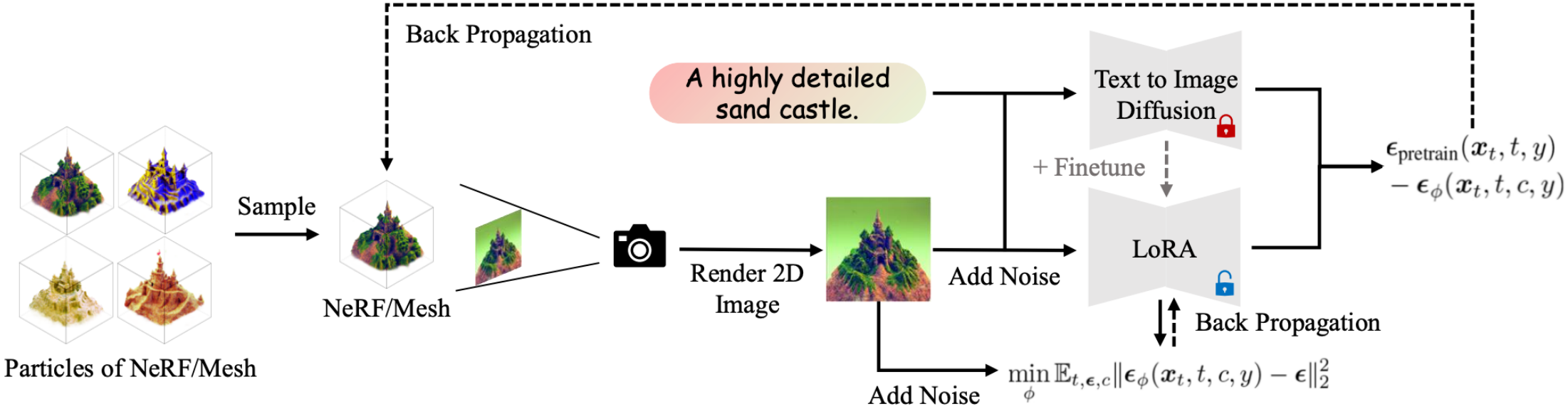
Magic3D [Lin et al., CVPR 2023]

- *Two-stage approach.*
- *Extract a coarse mesh in the first stage, and then texture the mesh in the second stage.*



ProlificDreamer [Wang et al., arXiv 2023]

- *Minimize the SDS loss for the multiple samples of the NeRF parameters ϕ.*
- *Finetune the diffusion model with the Low Rank Adaptation (LoRA) technique.*



Limitation of SDS

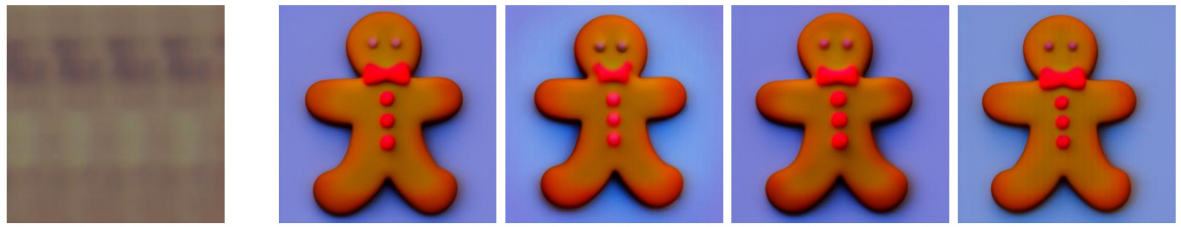
It does not converge well without a high CFG weight (e.g., $w = 400$) and thus suffers from model collapse.

"a delicious hamburger"

$w = 400$

$w = 7.5$

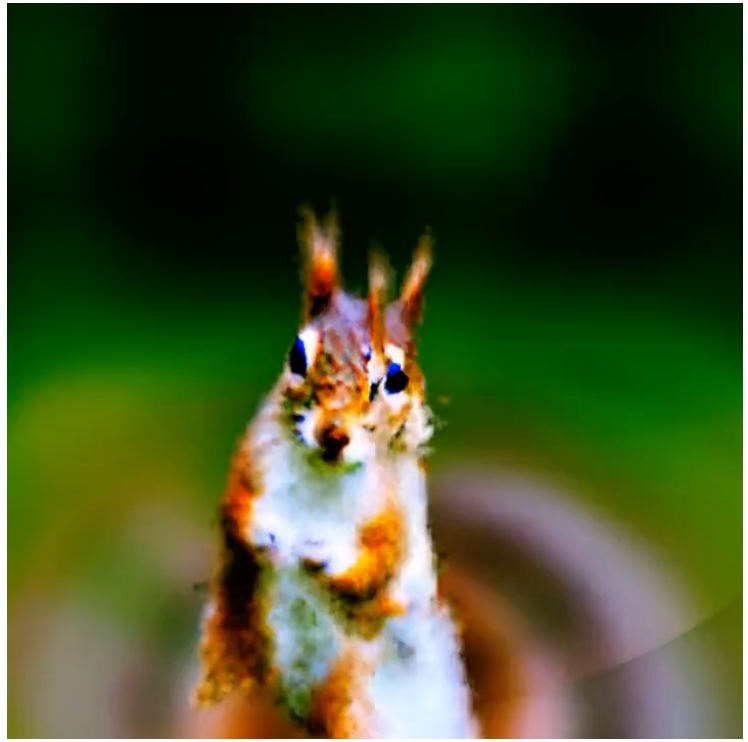
Credit: Jaihoon Kim

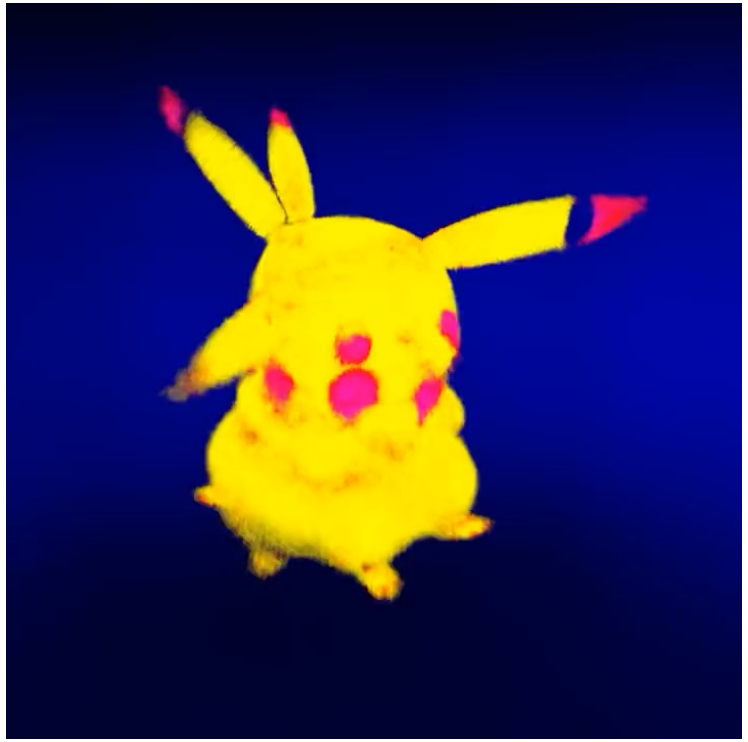


"gingerbread man"

Huang et al., DreamTime: An Improved Optimization Strategy for Text-to-3D Content Creation, arXiv 2023.

Limitation of 3D Generation from 2D Priors





Twitter @_akhaliq

Limitation of 3D Generation from 2D Priors

Supervision for geometry is still needed!

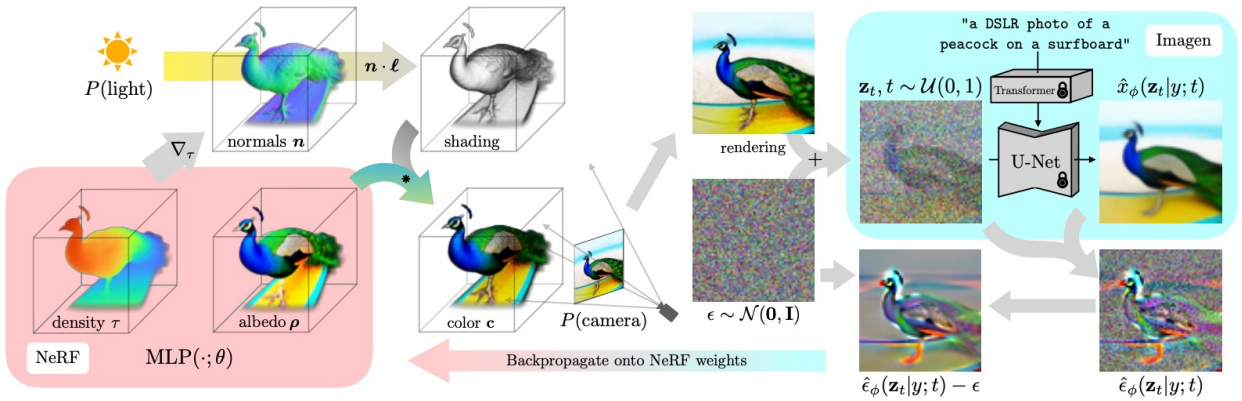
ProlificDreamer, Wang et al., 2023.

StableDreamFusion

What's Next for 3D Generative Models?

1. Combining 3D Supervision

While pretrained image generative models will continue to be valuable for 3D generation, the key to producing realistic and solid 3D shapes would lie in utilizing small-scale 3D priors.



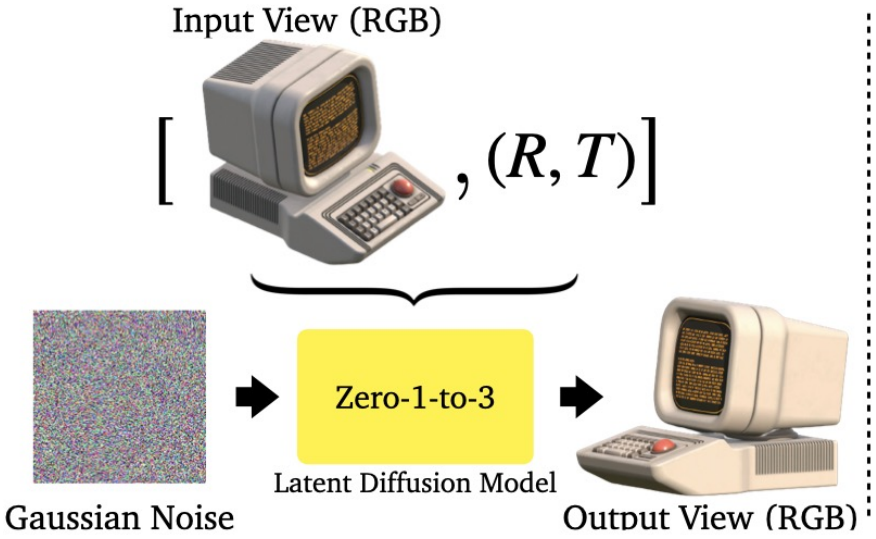
DreamFusion
Google

Objaverse
Allen Institute

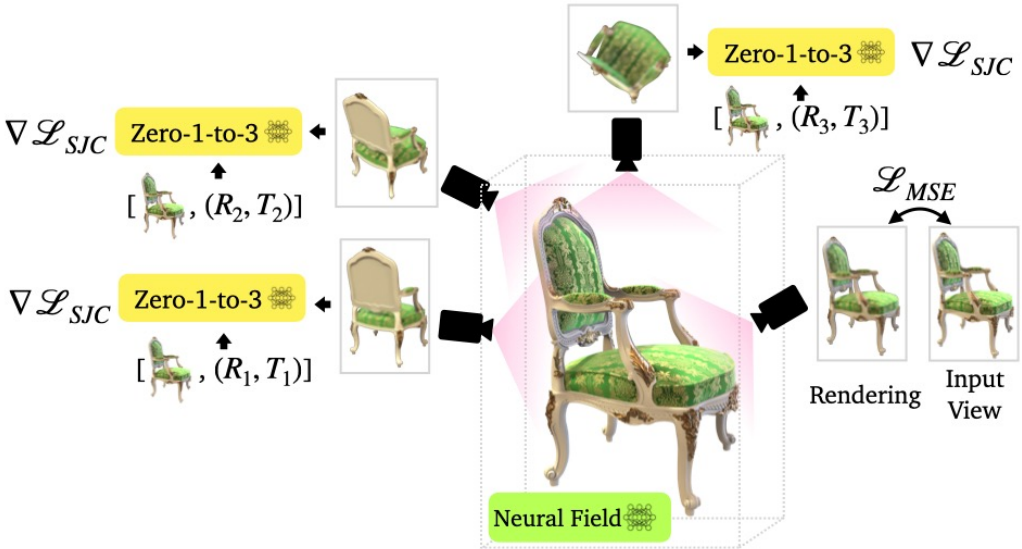
Zero-1-to-3 [Liu et al., 2023]

Novel view generation

An image diffusion model generating a novel view image conditioned by another view image and camera pose.



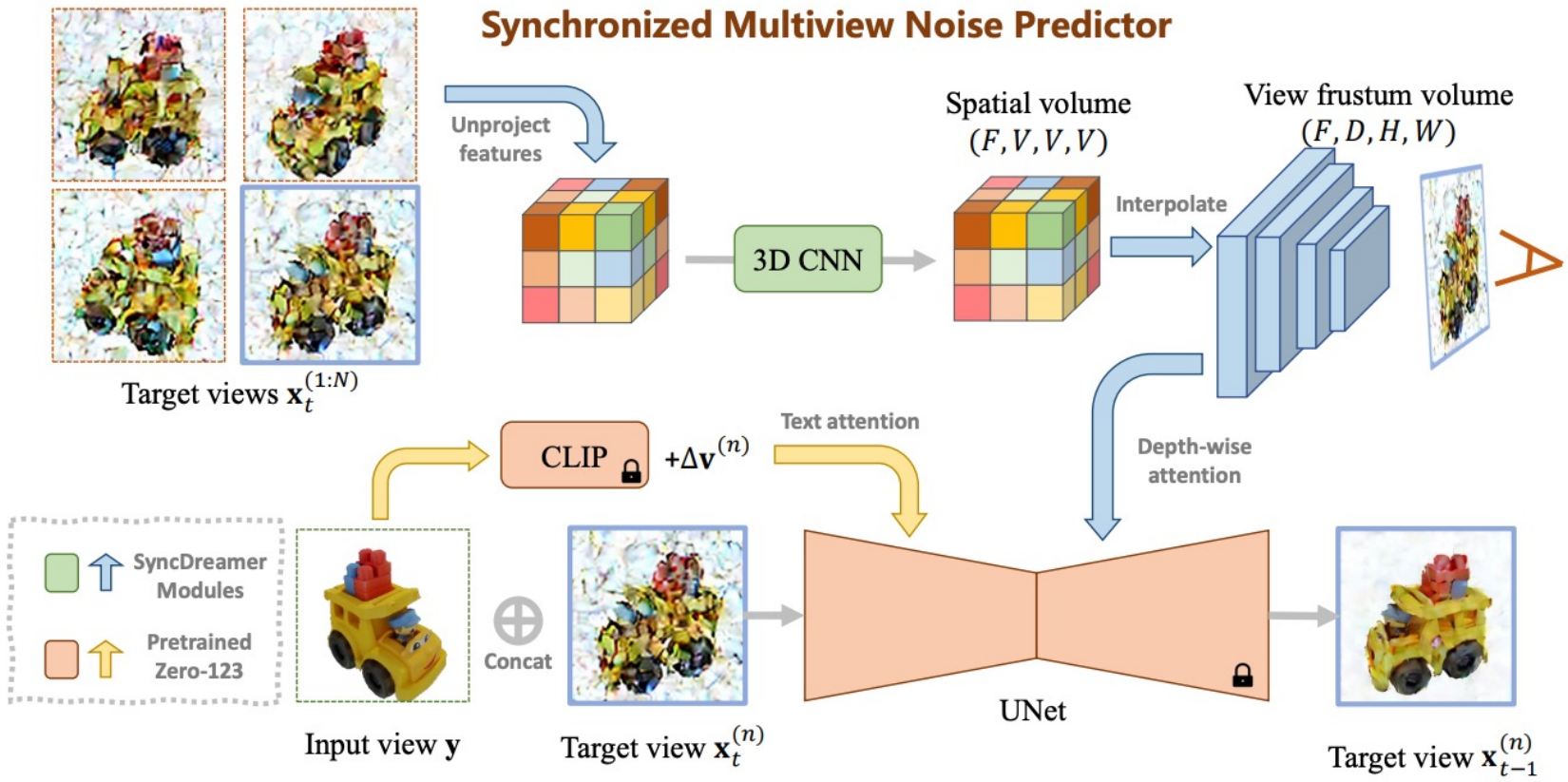
Novel View Synthesis



3D Reconstruction

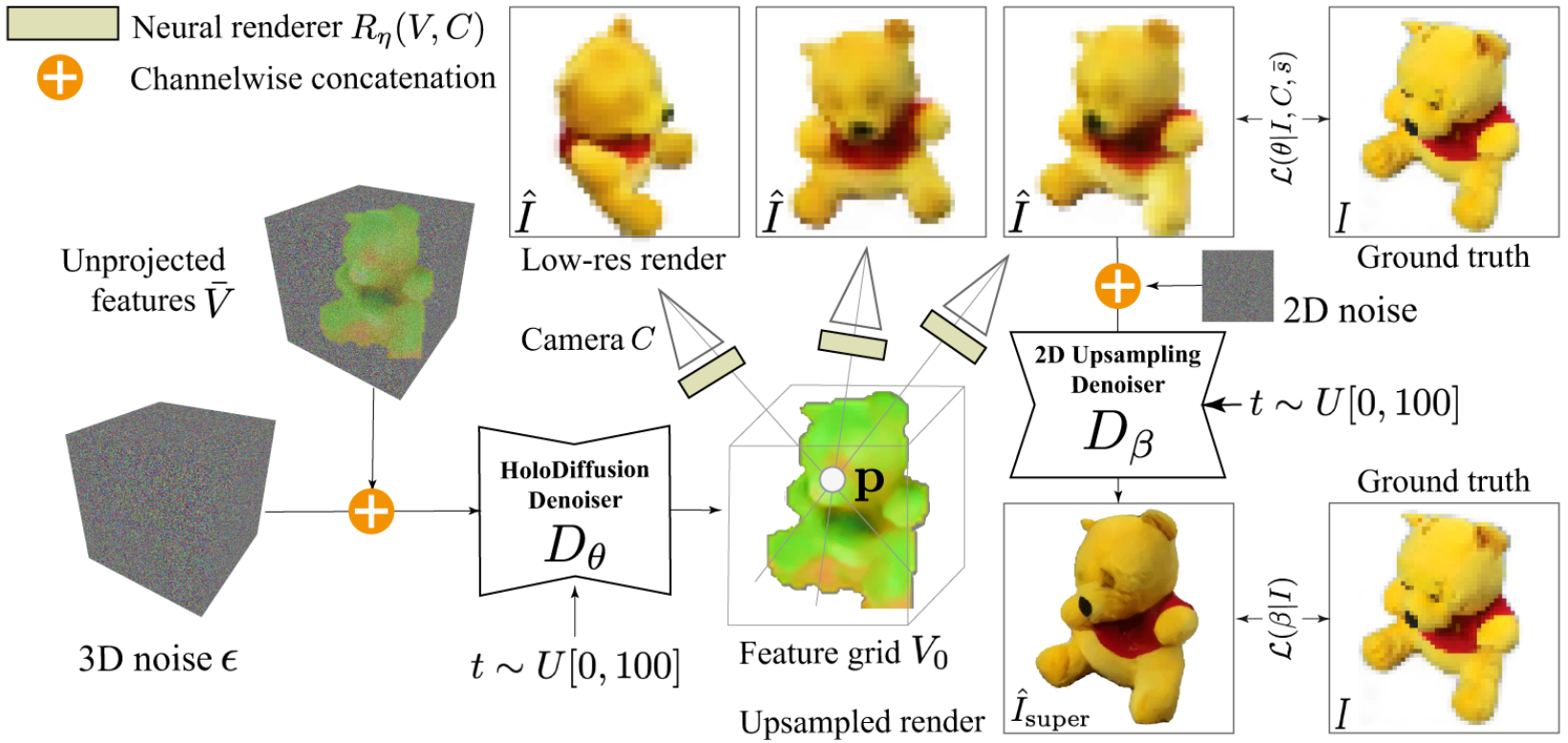
SyncDreamer [Liu et al., 2023]

Utilize Zero 1-to-3 to learn the joint probability distribution of multi-view images.



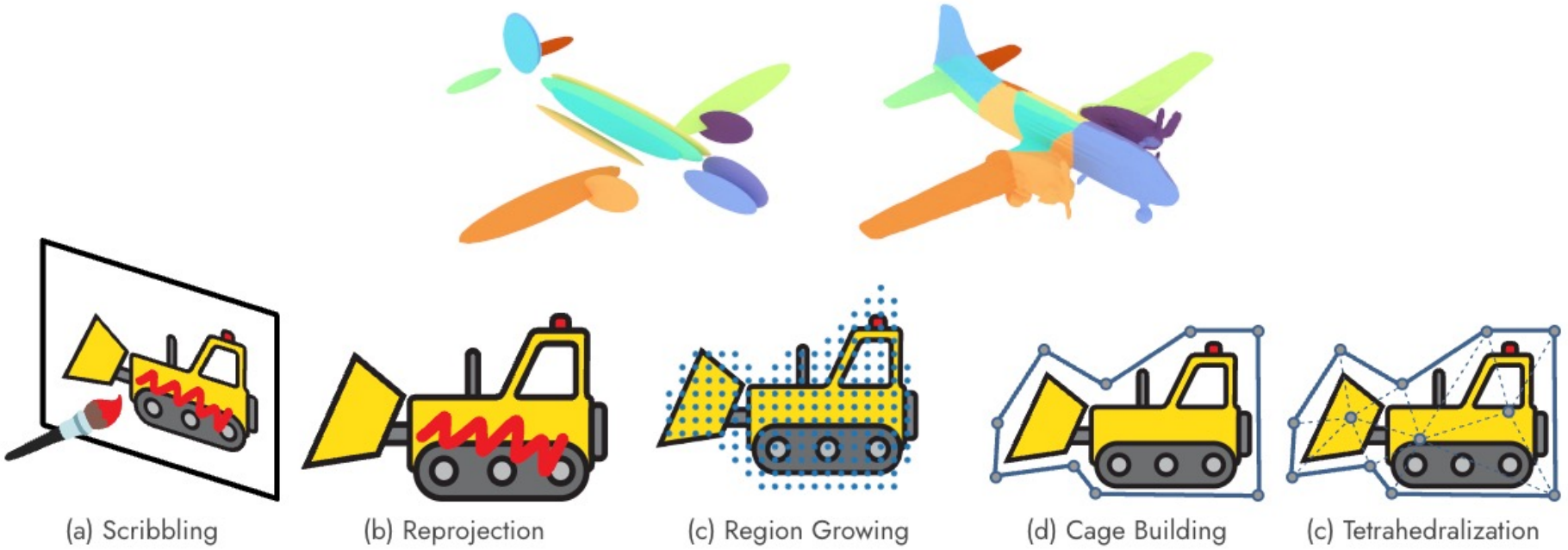
HoloFusion [Karnewar et al., 2023]

- *Train a 3D diffusion model using multi-view images only.*
- *Can be extended to integrate 2D priors.*



2. Generation → Editing

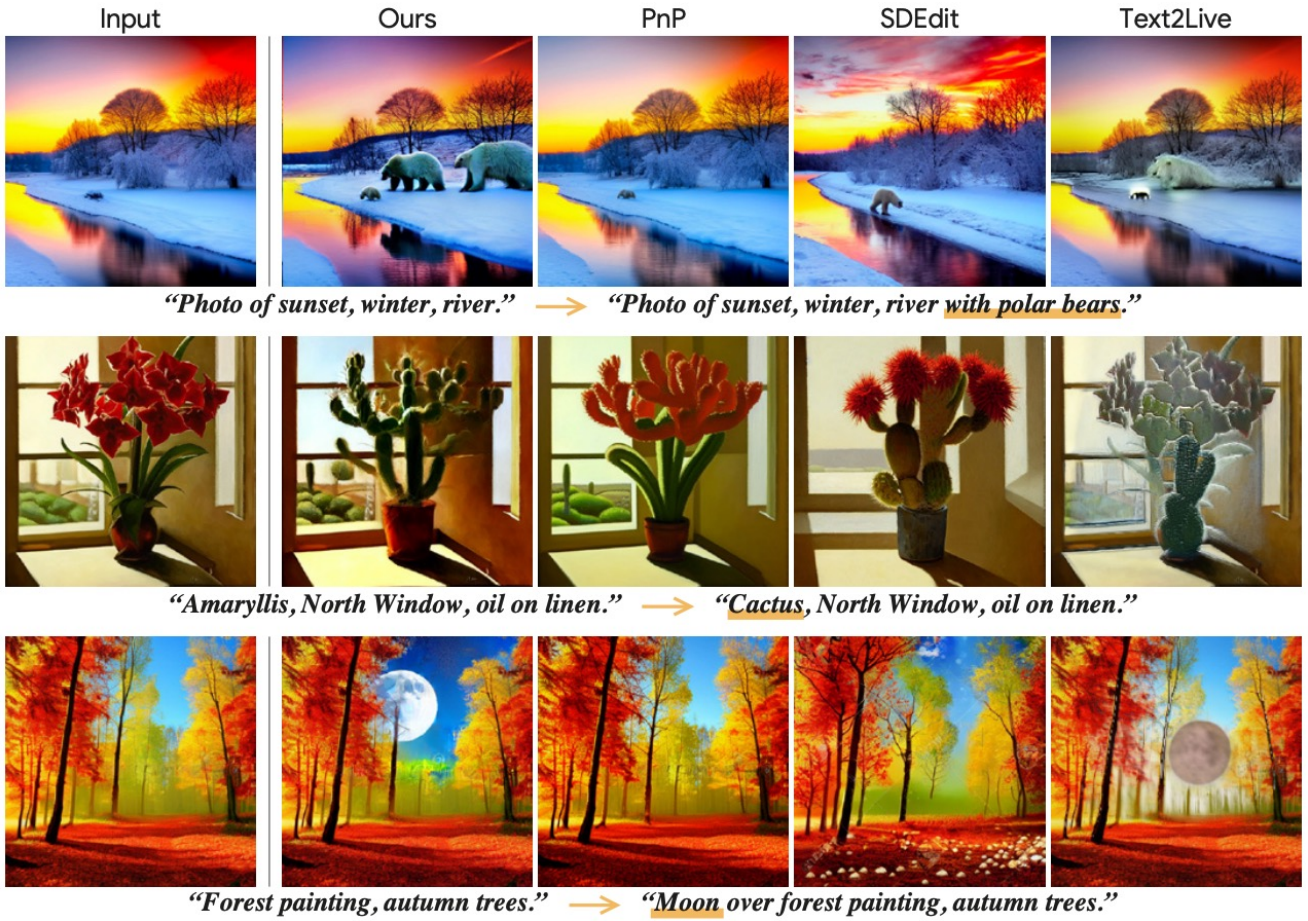
The focus of 3D generative models will shift towards creating versatile models capable of not only generating but editing and manipulating 3D shapes.



NeRFshop, Jambon et al., I3D 2023.

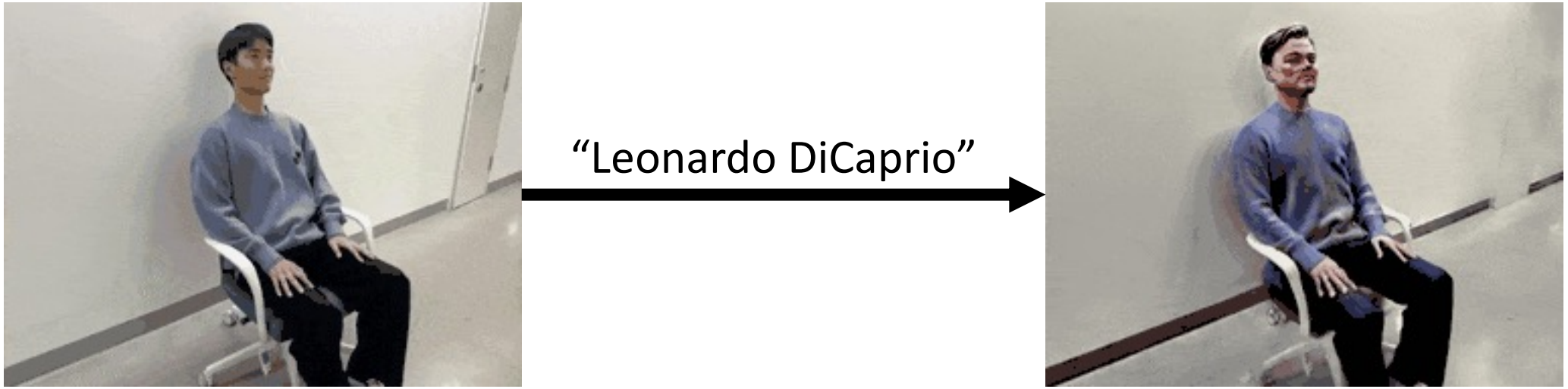
Delta Denoising Score [Hertz et al., 2023]

A new loss function for zero-shot image editing.



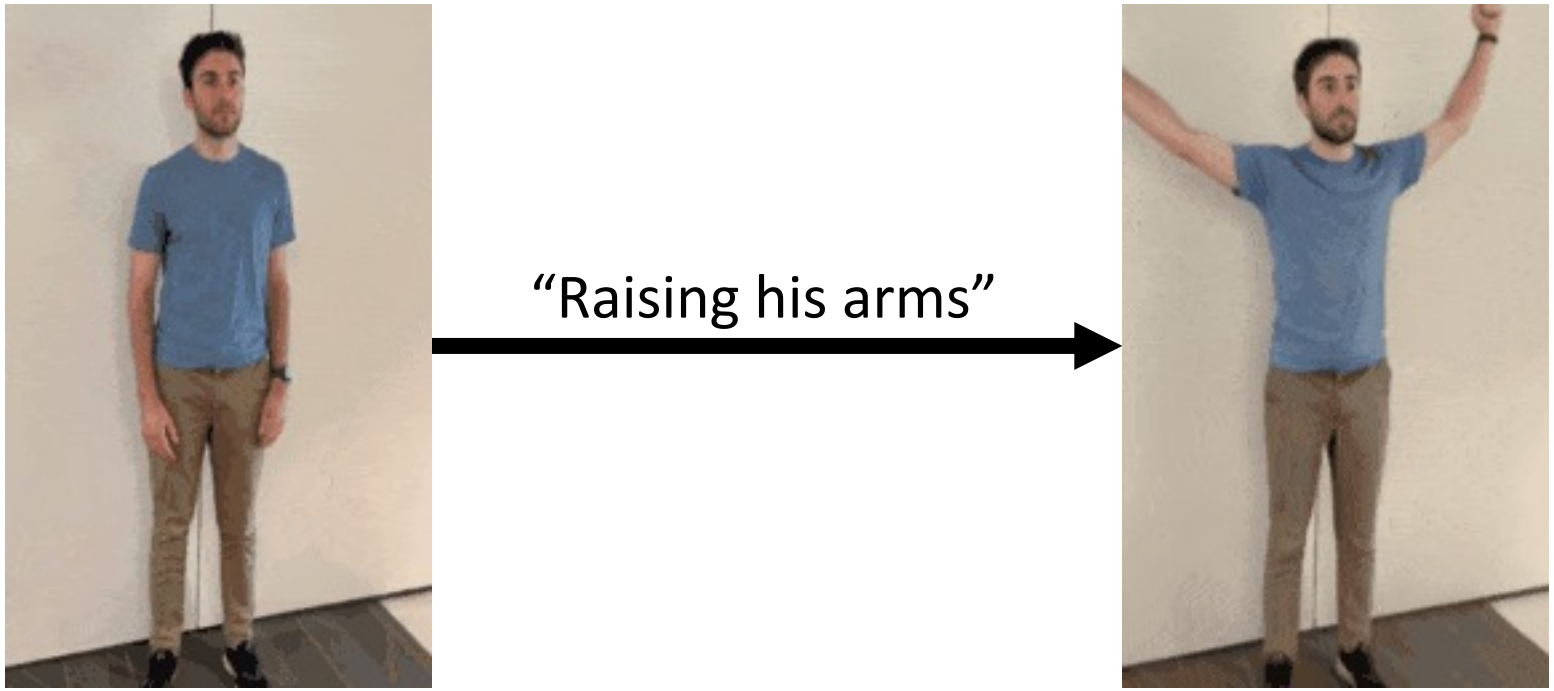
Posterior Distillation Sampling [Koo et al., 2024]

A new loss function for zero-shot NeRF editing.



Posterior Distillation Sampling [Koo et al., 2024]

A new loss function for zero-shot NeRF editing.



Kim *et al., SyncTweedies: A General Generative Framework Based on Synchronized Diffusions, arXiv 2024.*

3. Texture Generation

How can we generate photorealistic texture given a 3D mesh or Gaussian splats?