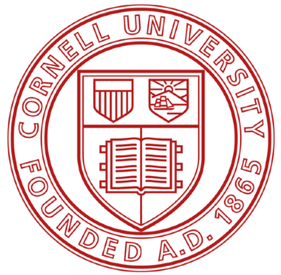
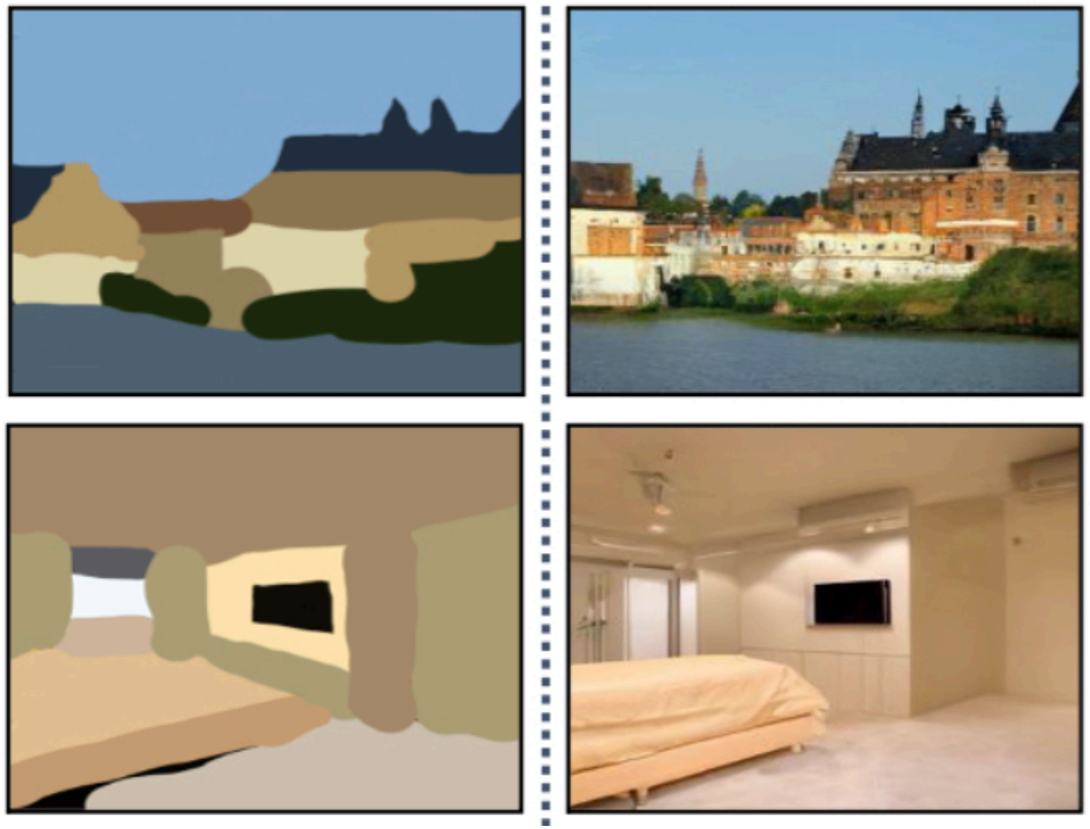
Lecture 18: Image manipulation with diffusion models

CS 5788: Introduction to Generative Models



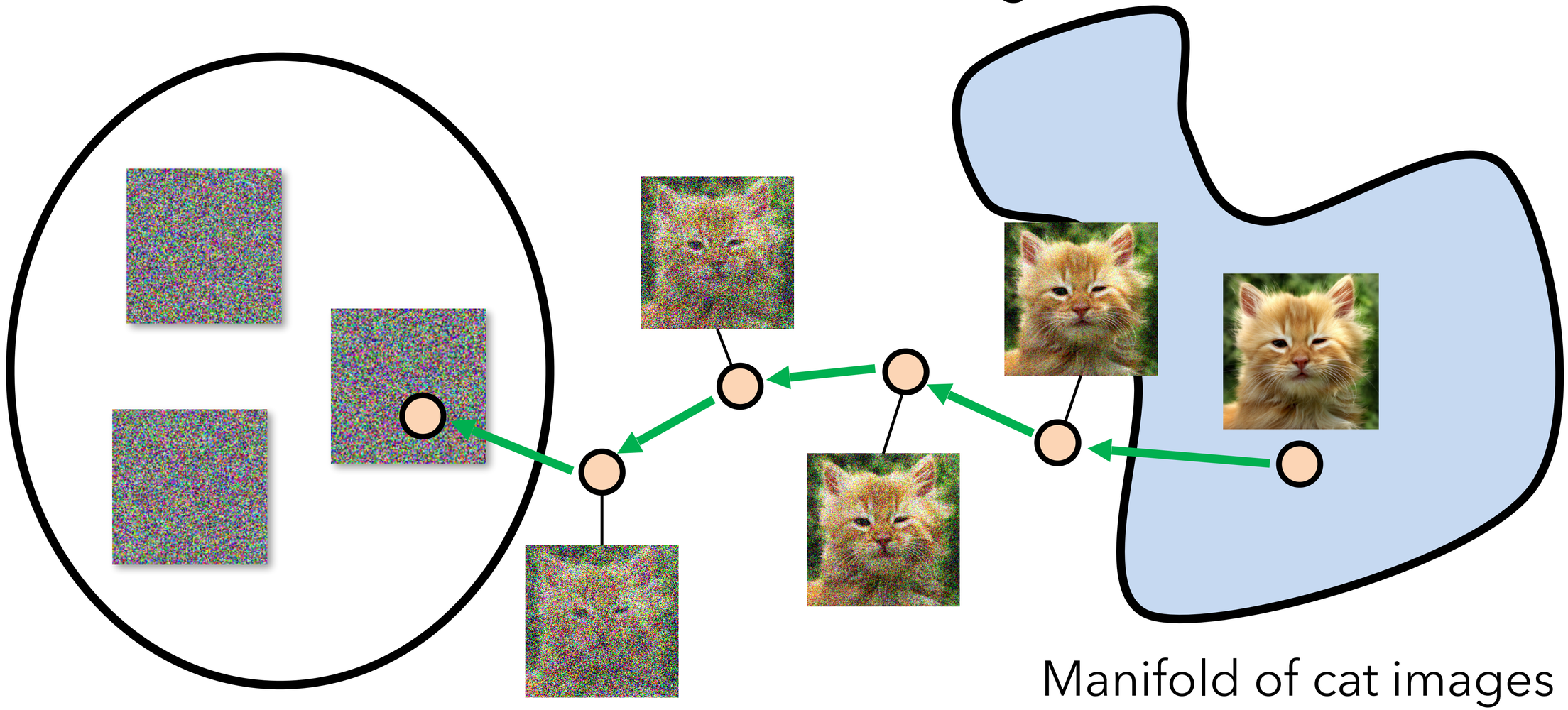
Case study: generating and manipulating images

stroke to image

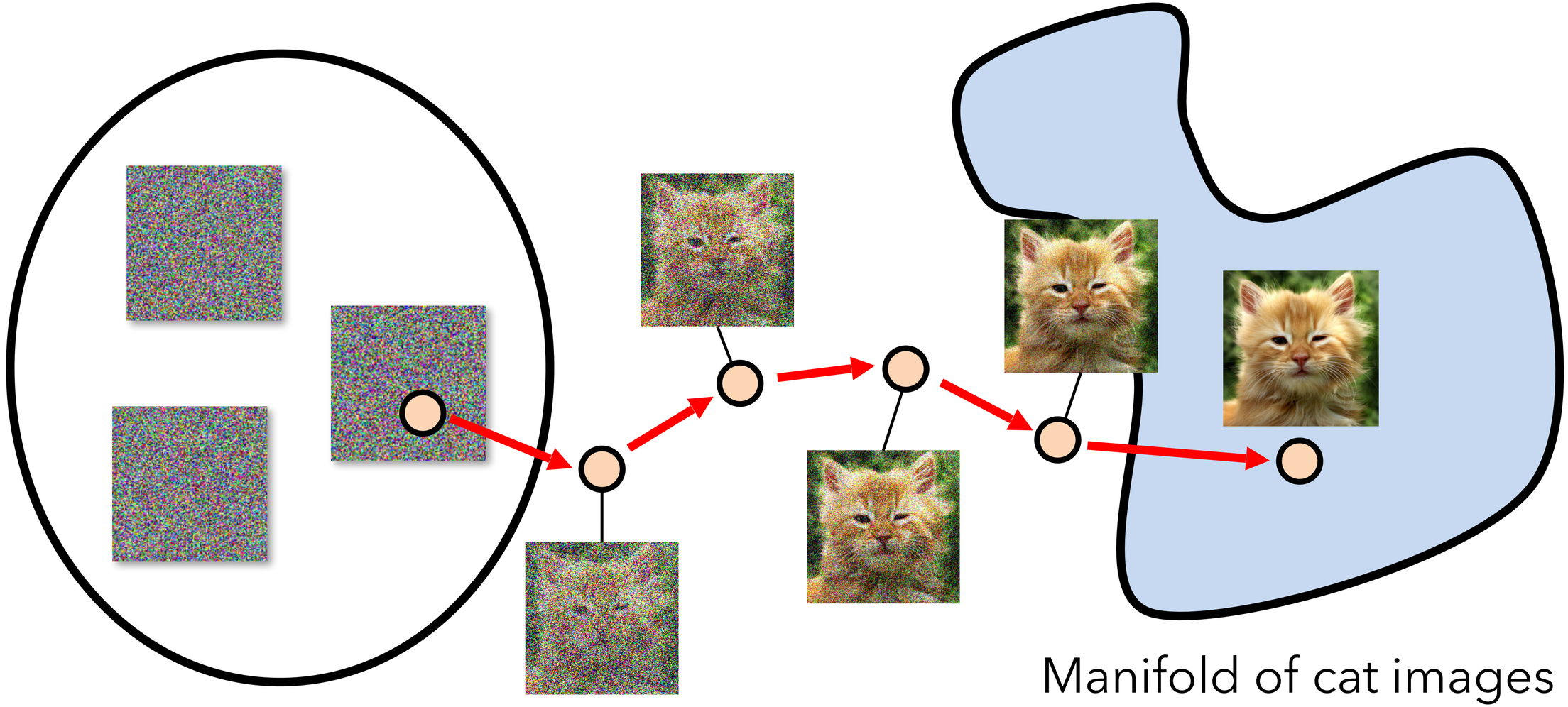


“swap sunflowers with roses”

Recall: diffusion models (and flow matching)

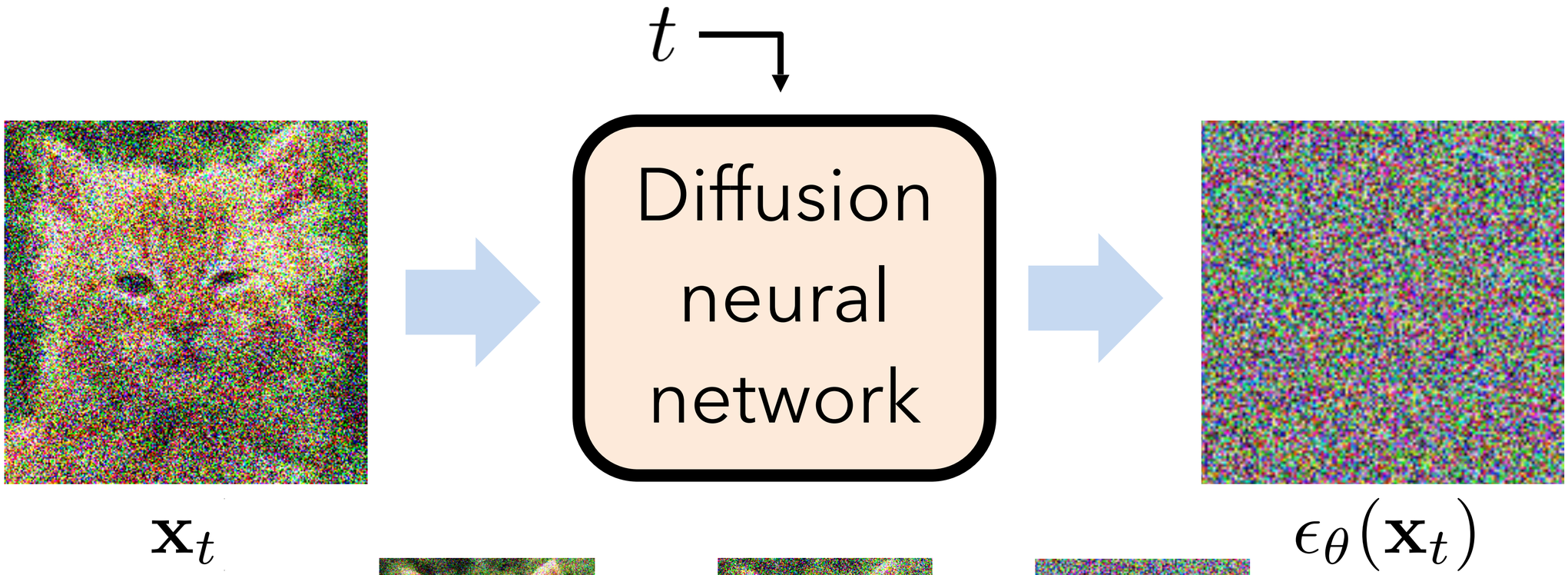


Slide source: N. Snavely & S. Seitz



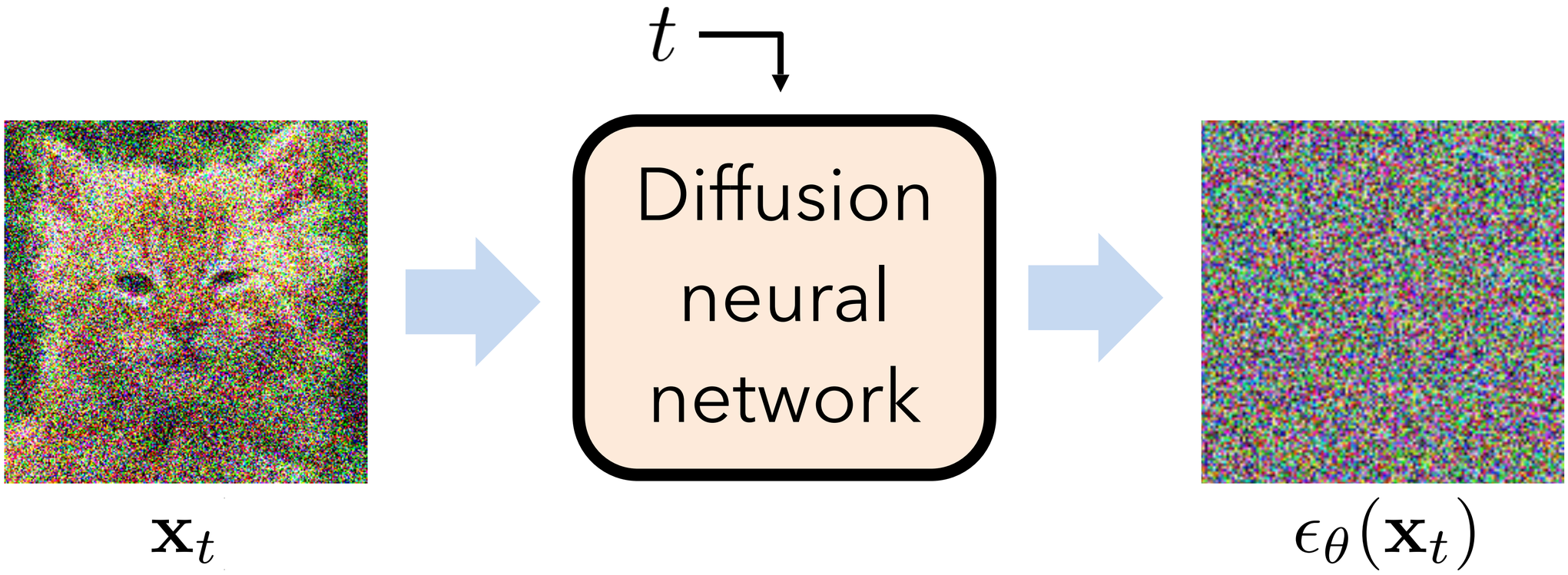
Slide source: N. Snavely & S. Seitz

Recall: noise estimation network



Adapted from N. Snavely

Recall: noise estimation network



Score function interpretation:

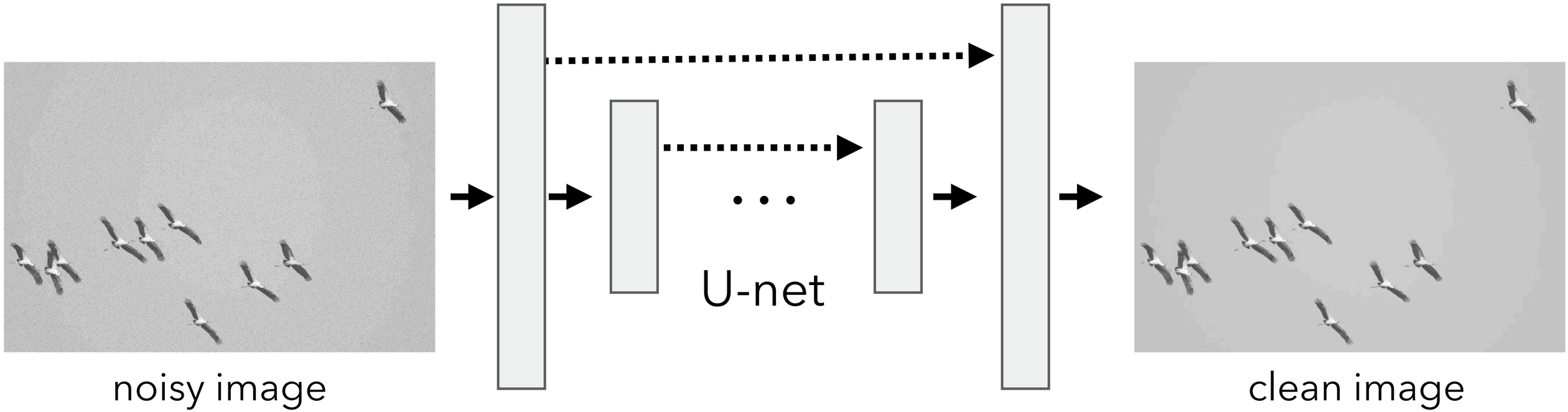
$$\epsilon_\theta(\mathbf{x}_t) \approx -\sigma_t \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$$

Adapted from N. Snavely

Today

- How do you design this denoiser?
- How do you condition on other signals?
- Can we use these models to *manipulate* images?

U-net as a denoiser (in PS3)

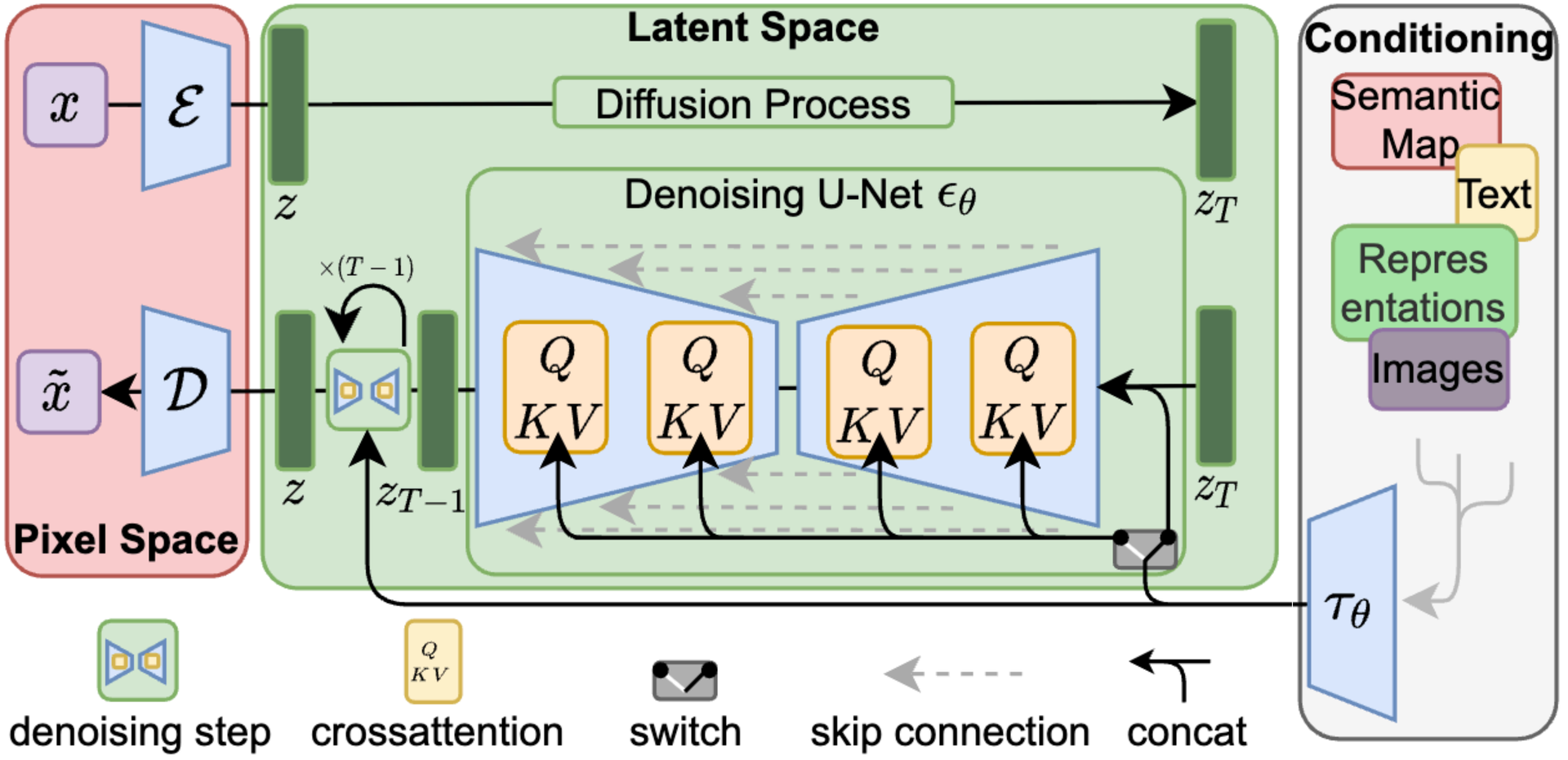


noisy image

clean image

Pros: Simple, relatively easy to train.
Cons: Not scalable. Heavy coarse-to-fine “inductive bias”. Awkward to condition on data that doesn’t live on a grid.

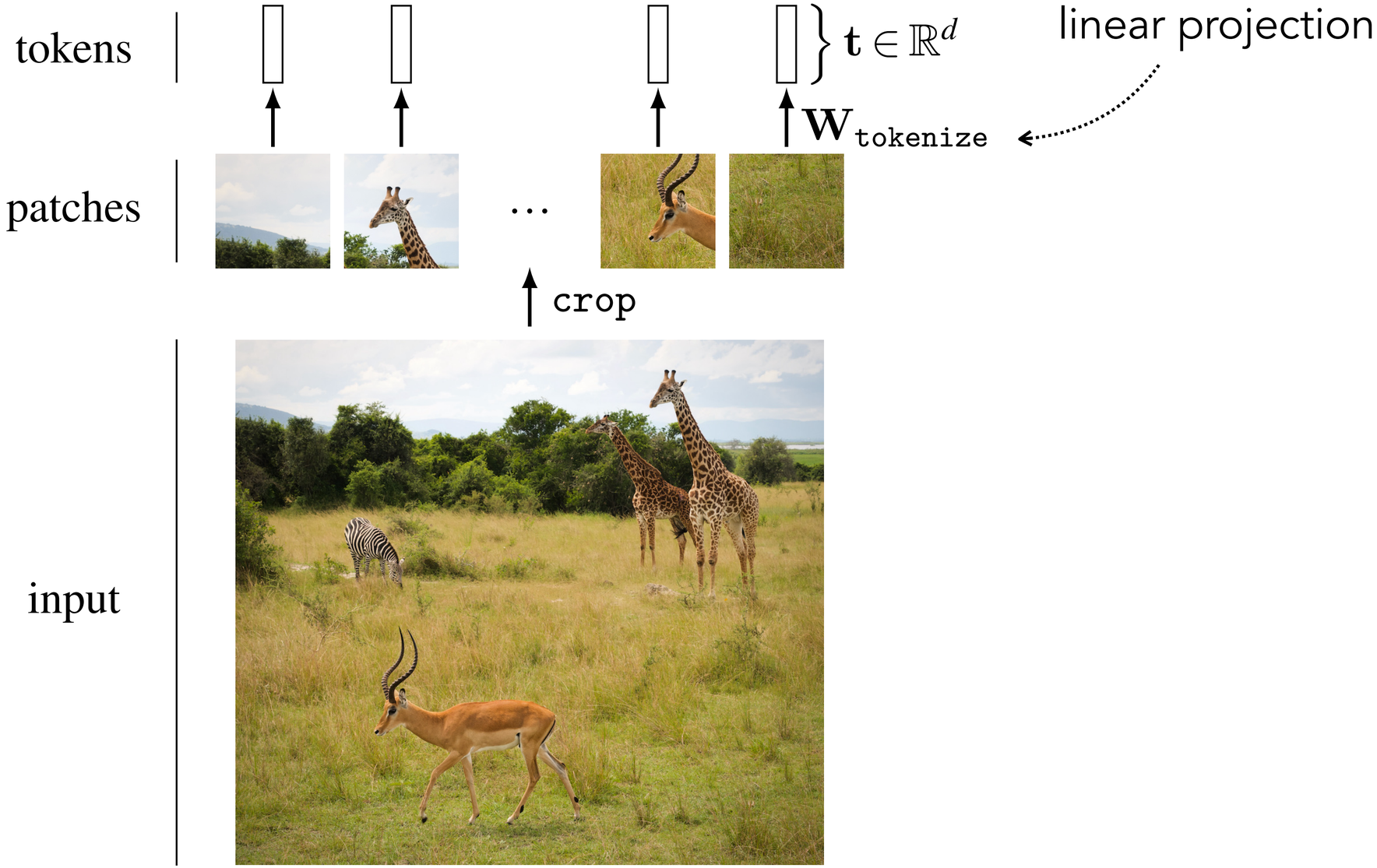
Example: architecture used in Stable Diffusion



Source: [Rombach et al., "Latent diffusion", 2021]

Are there more flexible, scalable architectures?

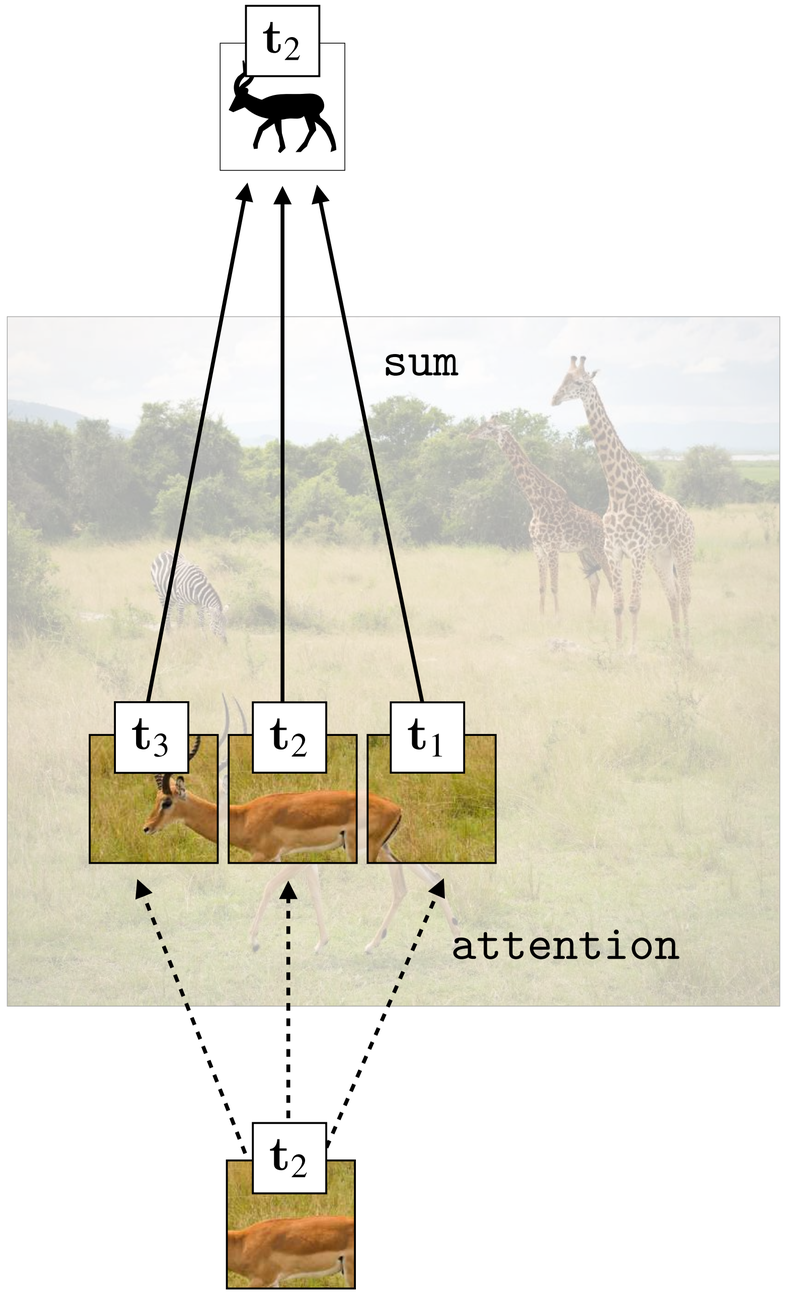
Vision transformer (ViT)



[Dosovitskiy et al., 2020]

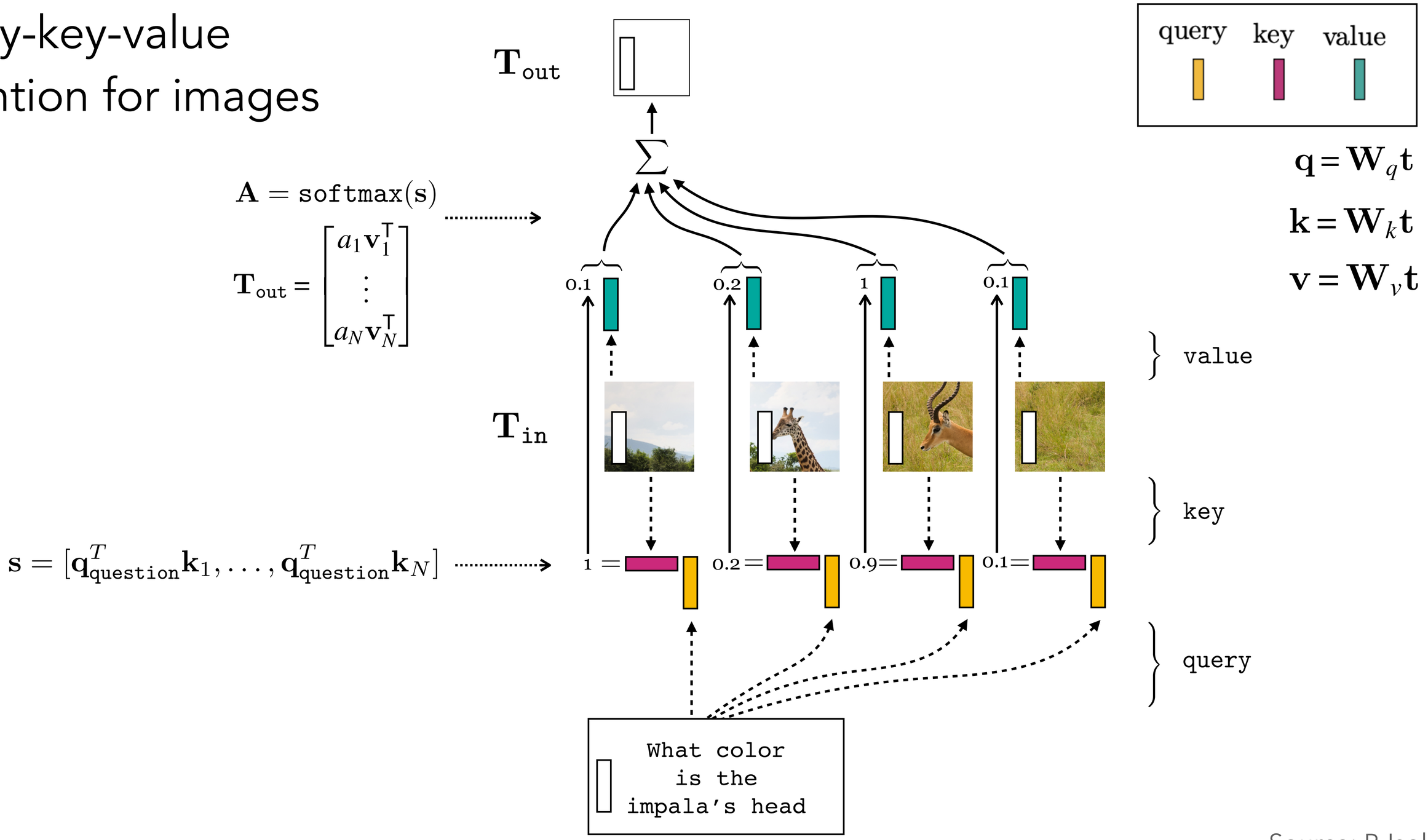
Slide source: P. Isola

Self-attention



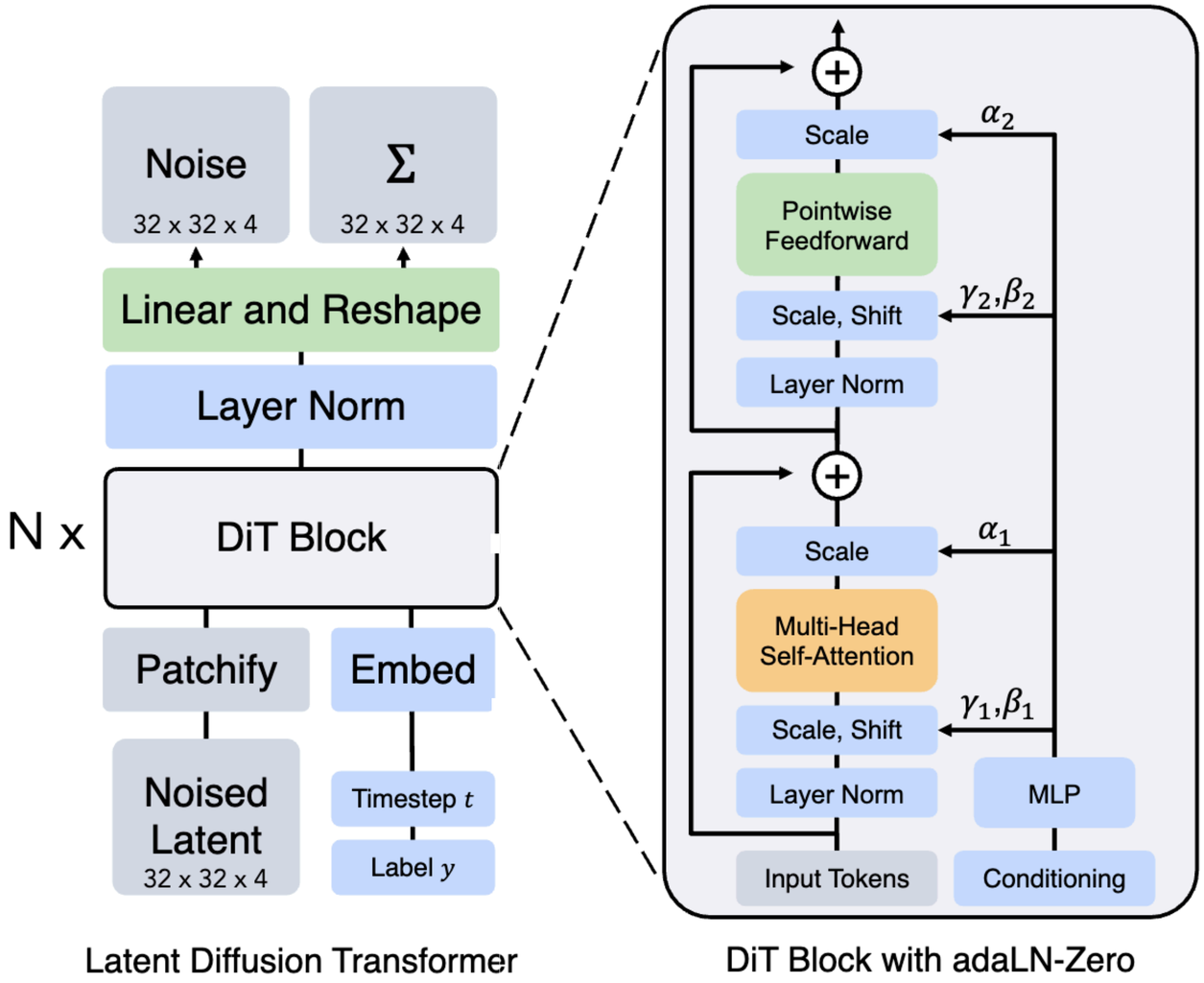
Source: P. Isola

query-key-value attention for images



Source: P. Isola

Diffusion transformer (ViT for diffusion)



Samples on ImageNet in 2023

Use a vision transformer estimates noise from latents.

[Peebles and Xie, 2023]
see also [Jabri et al., 2022]

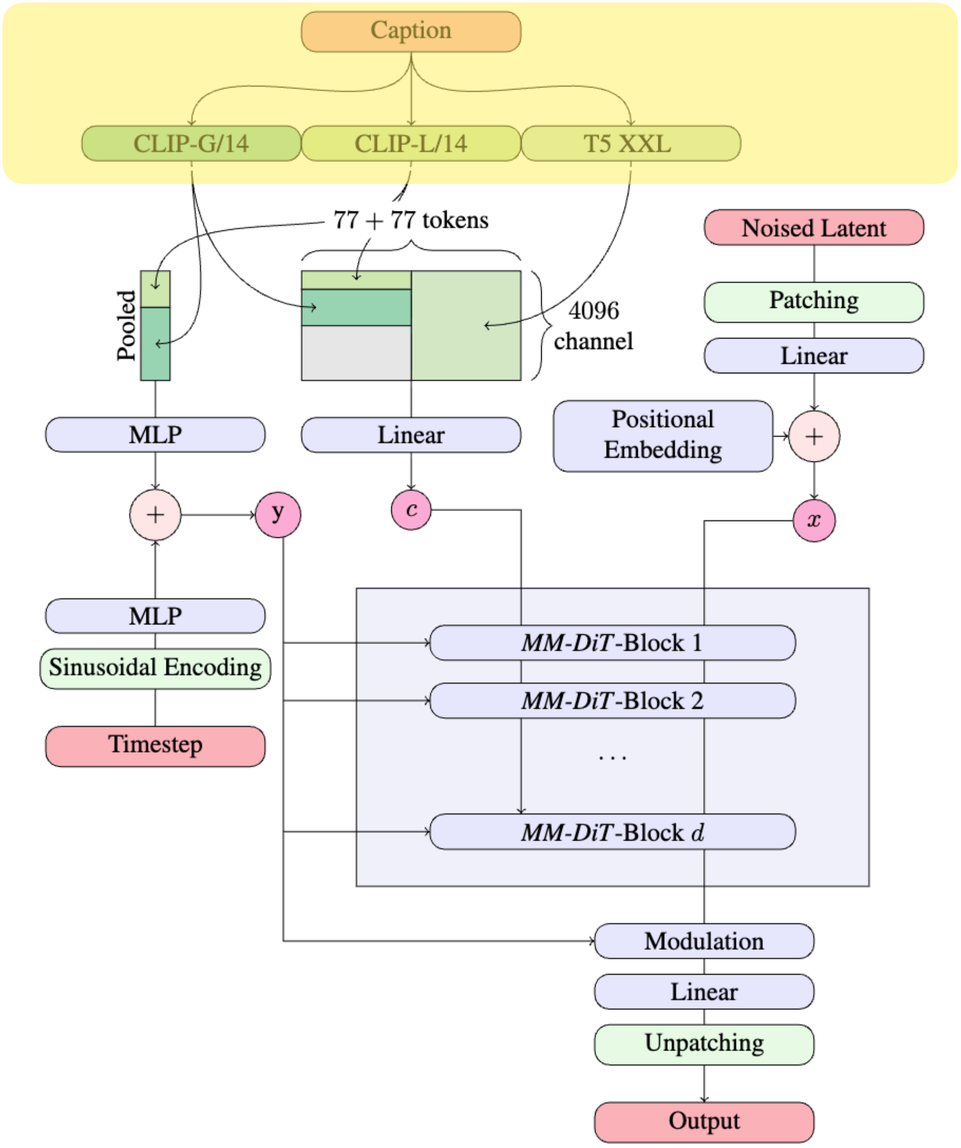
How do you condition on language?

“Sprouts in the shape of text 'Imagen' coming out of a fairytale book.”

“A dragon fruit wearing karate belt in the snow.”

Source: L. Shen, [C. Saharia et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. NeurIPS 2022]

Conditioning on language

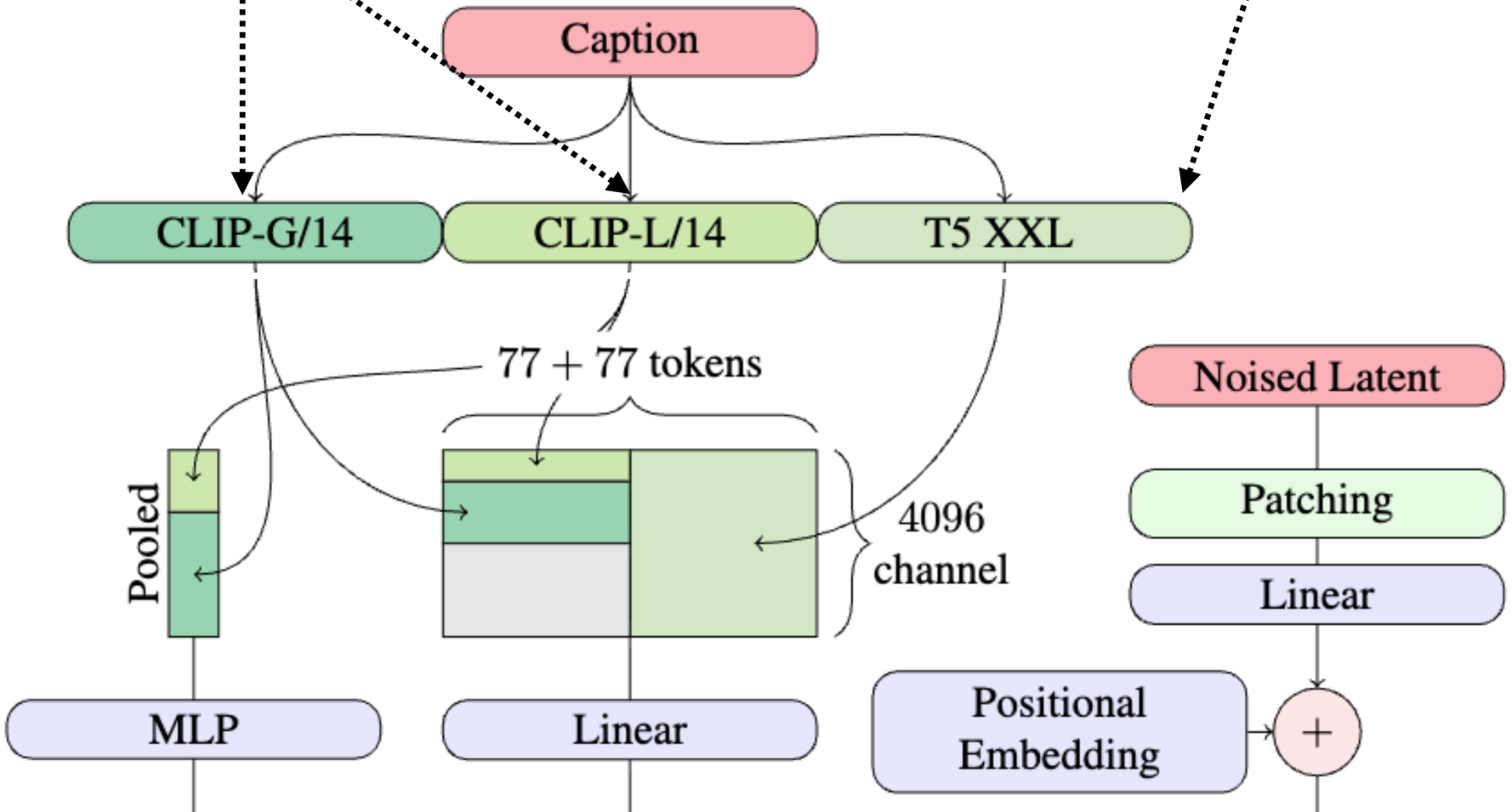


Case study: Stable Diffusion 3 [Esser et al., 2024]

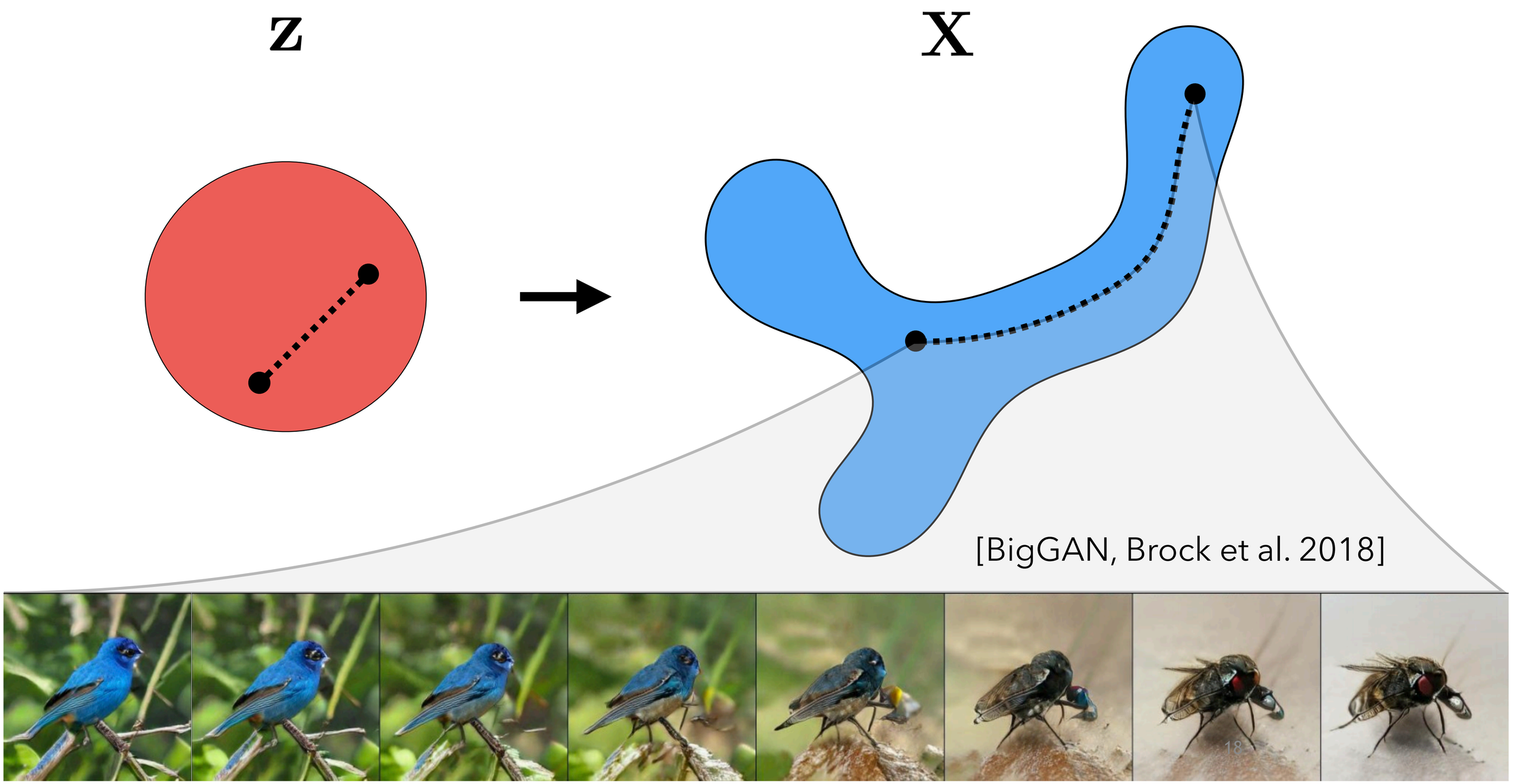
Conditioning on language

CLIP features, i.e., joint embedding between text and images.

Features from an LLM.

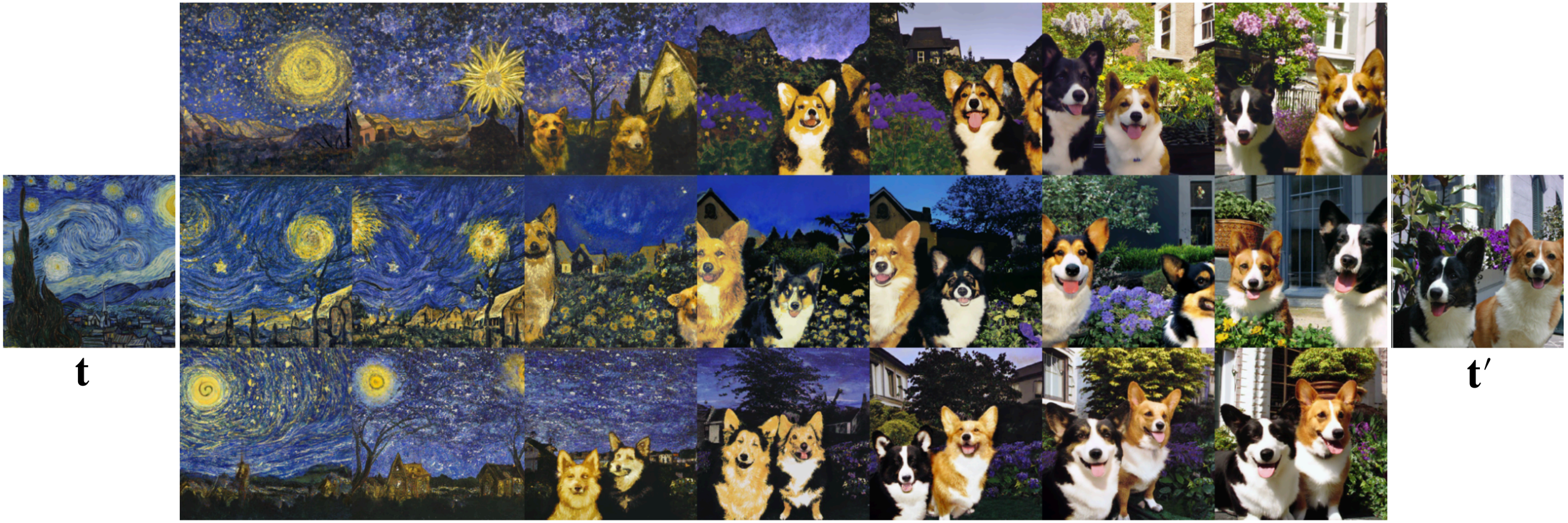


Recall: latent space interpolation (previously with GANs)



Interpolating images via text embeddings

Interpolate between text features (CLIP), generating an image at each point.

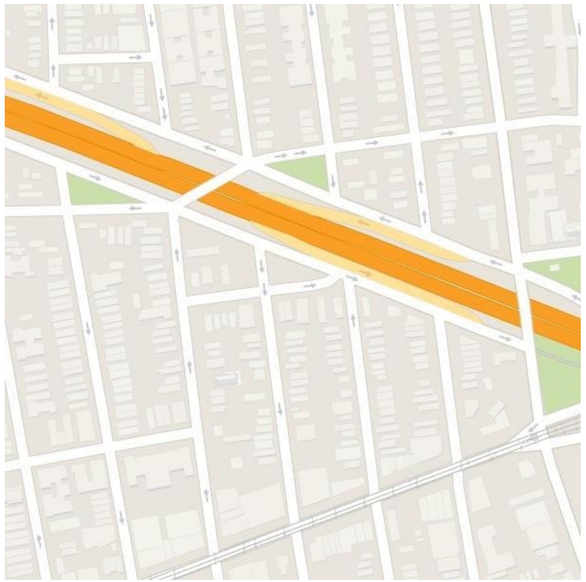


$$\text{text\_embedding2image}\left((1-w)\mathbf{t} + w\mathbf{t}' \text{ for } w \in [0,1]\right)$$

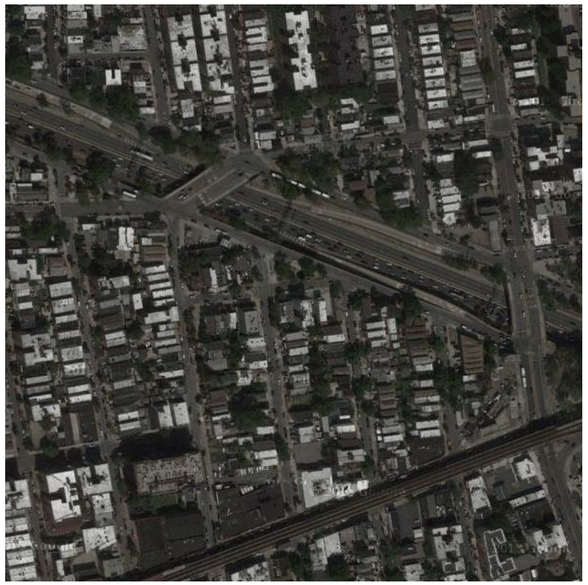
Source: [Ramesh et al., “DALL-E 2”]

Manipulating images

Recall: image-to-image translation (previously with GANs)



Google Map



Satellite photo

For GANs, this required a special architecture and loss.
Do we need this for diffusion models as well?

Adapted from P. Isola

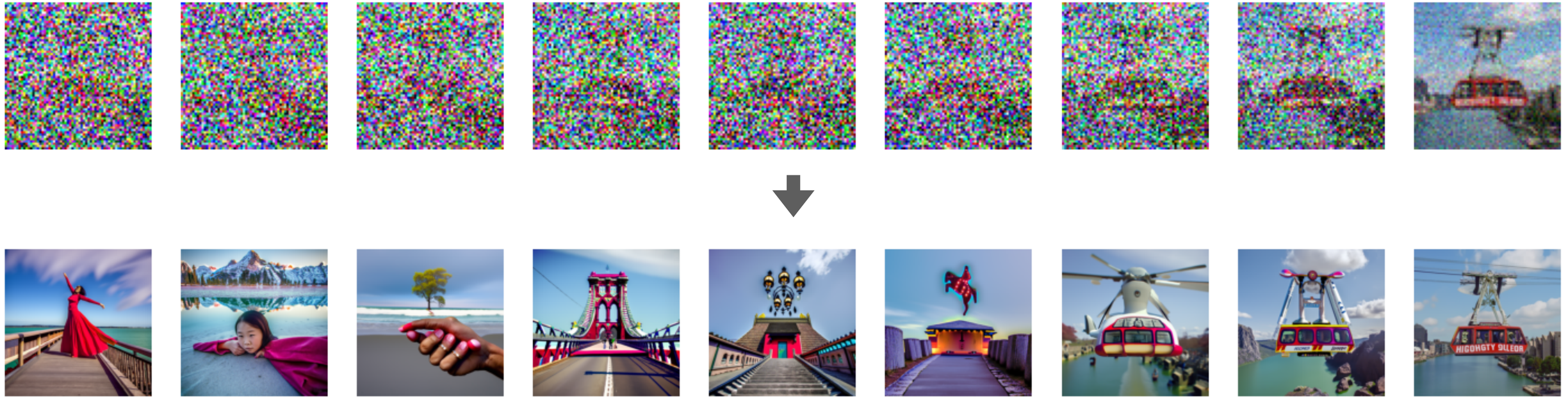
What’s special about diffusion models?

- They iteratively generate images.
- They can be guided by tweaking the score function.
- The iterative process has a schedule: i.e., generate noisy images first, then clean ones.

Denoising from intermediate noise levels

Add noise to this image, then run the reverse diffusion process.

Denoising from intermediate noise levels

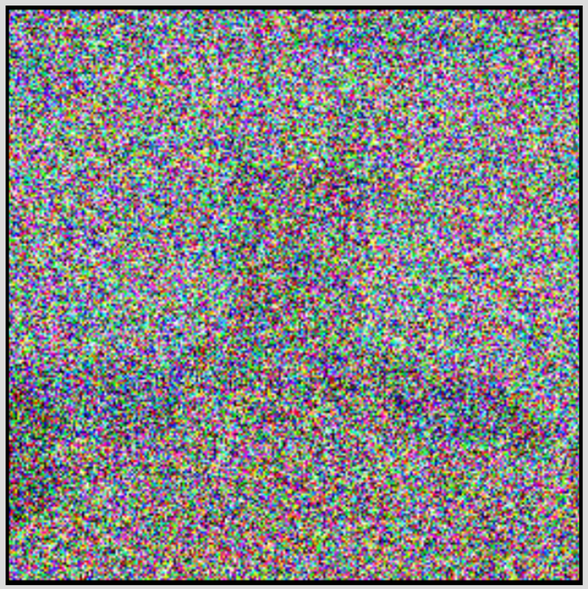


$t = 0$

Sketch to photo

Add noise to a sketch

$\mathbf{x}_0$



$\mathbf{x}_t$

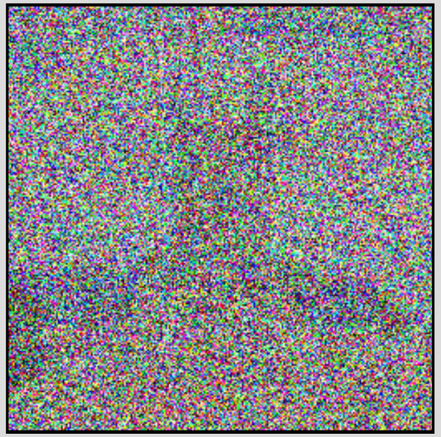
$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \text{ for } \epsilon \sim N(0,1)$$

[Meng et al. "SDEdit", 2022]

Sketch to photo

Add noise to a sketch

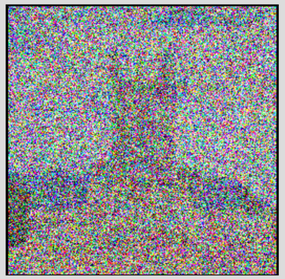
$\mathbf{x}_0$



$\mathbf{x}_t$

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \text{ for } \epsilon \sim N(0,1)$$

Denoise from $t \rightarrow 0$

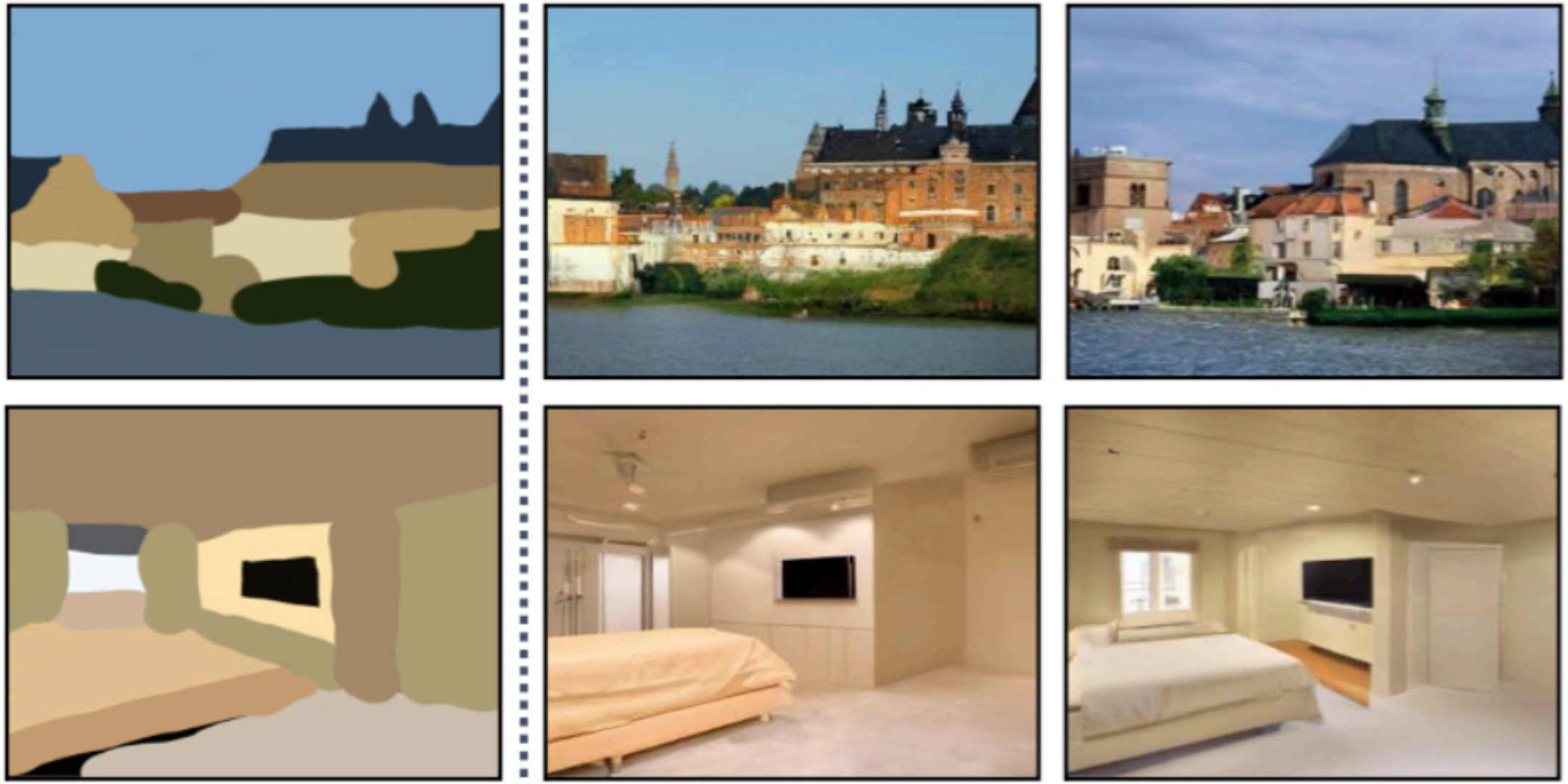


Denoise using diffusion model trained on *real* images.

SDEdit: Stochastic Differential Editing

[Meng et al. "SDEdit", 2022]

Sketch to photo

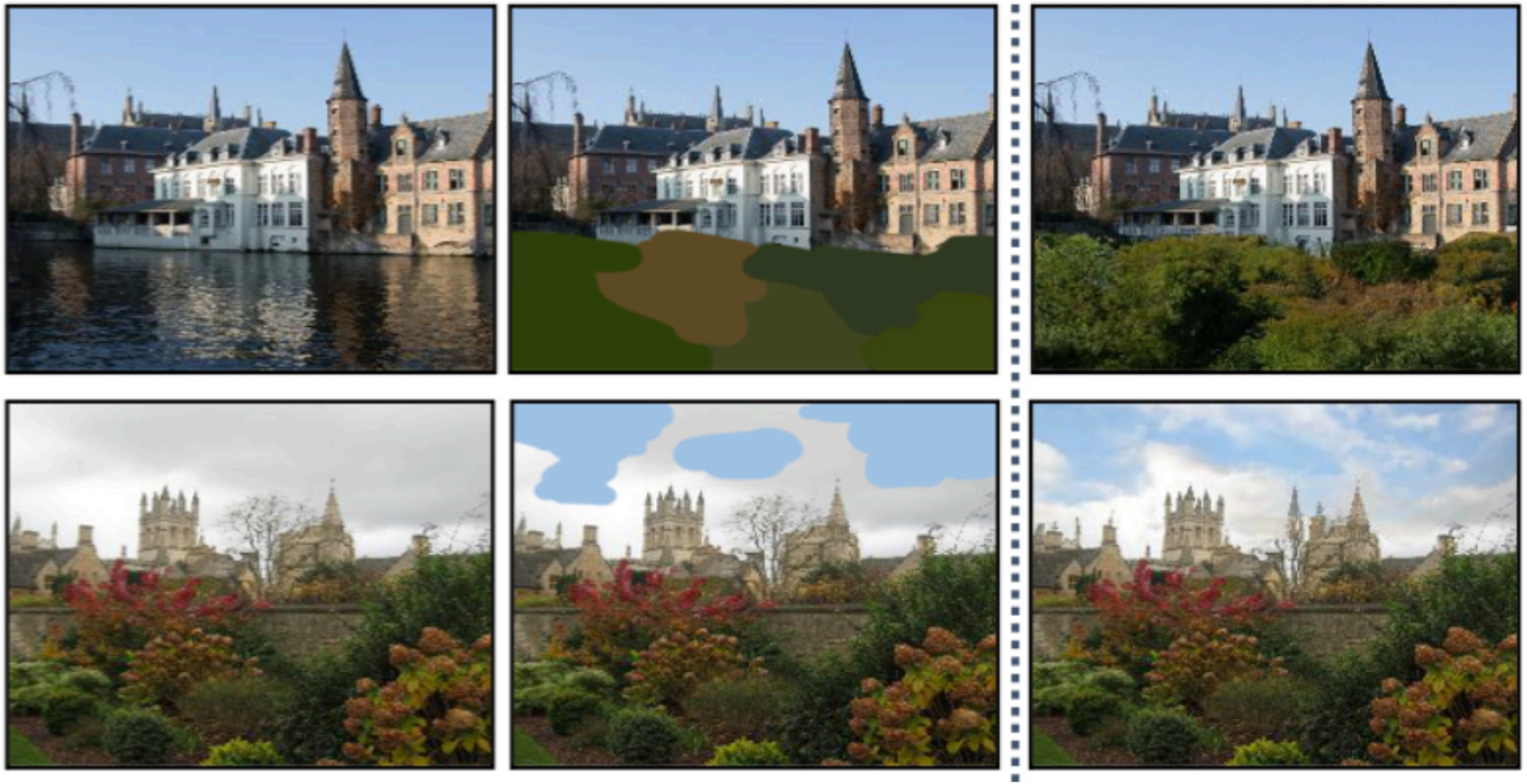


Input

Output

Source: [Meng et al. “SDEdit”, 2022]

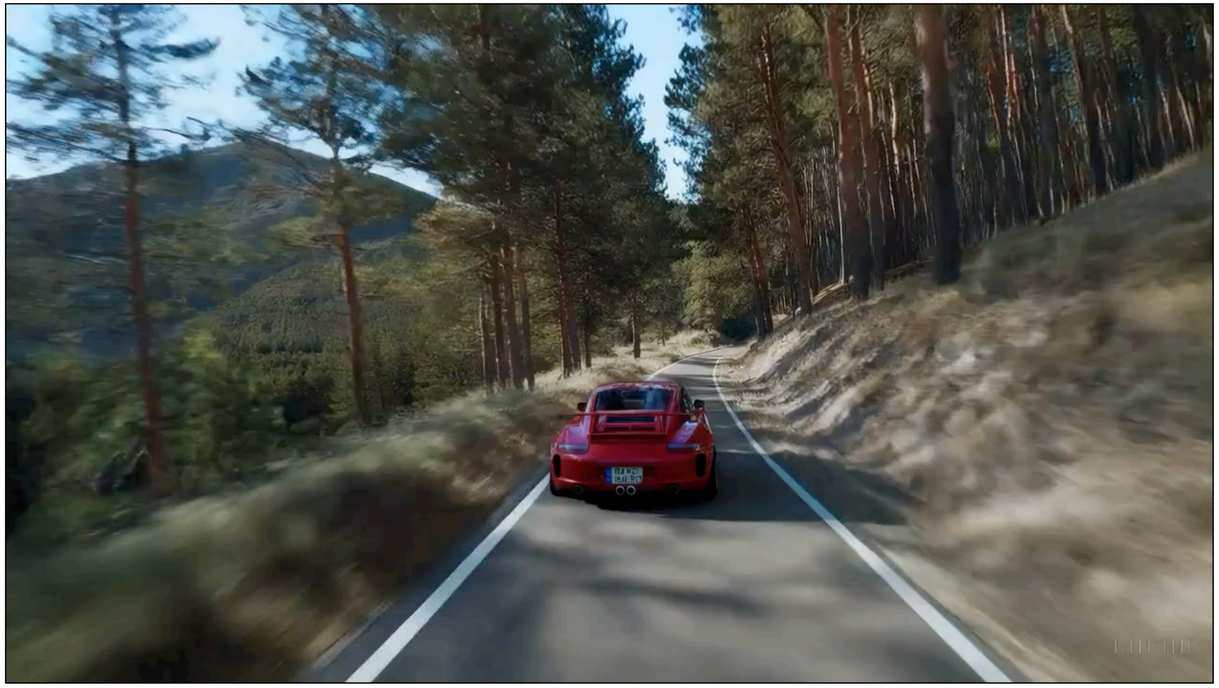
Stroke-based image editing



Source image | Image + stroke | Regenerated image

Source: [Meng et al. “SDEdit”, 2022]

Video-to-video translation with SDEdit and Sora



original generated video

rewrite the video in a pixel art style

Video-to-video translation with SDEdit and Sora

original generated video

change the video to a medieval theme

Creating paired images

Generating two images with similar prompts:

Photo of a cat riding a bicycle

Photo of a cat riding a car

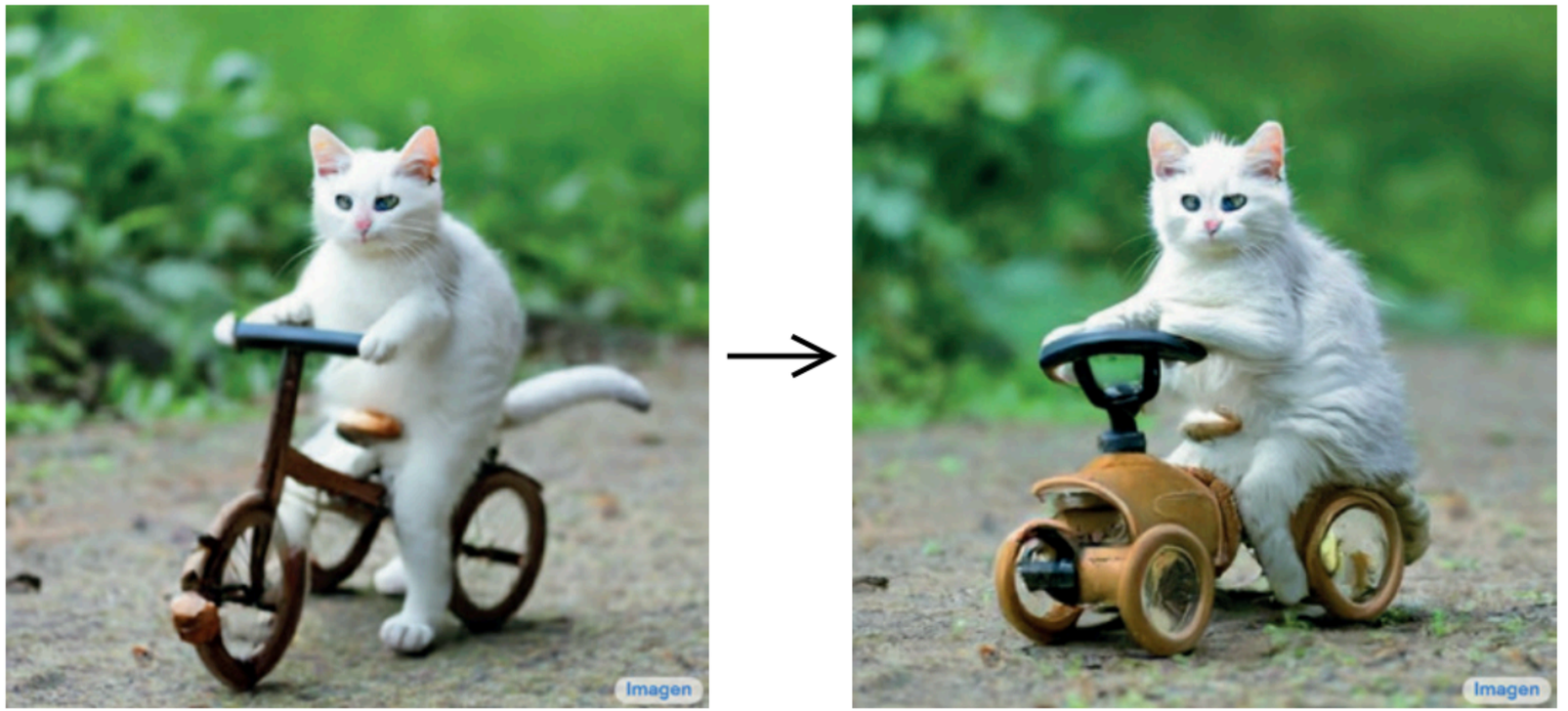
The images are quite different.

Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

Adapted from A. Hołyński

Creating paired images

What we'd like instead:

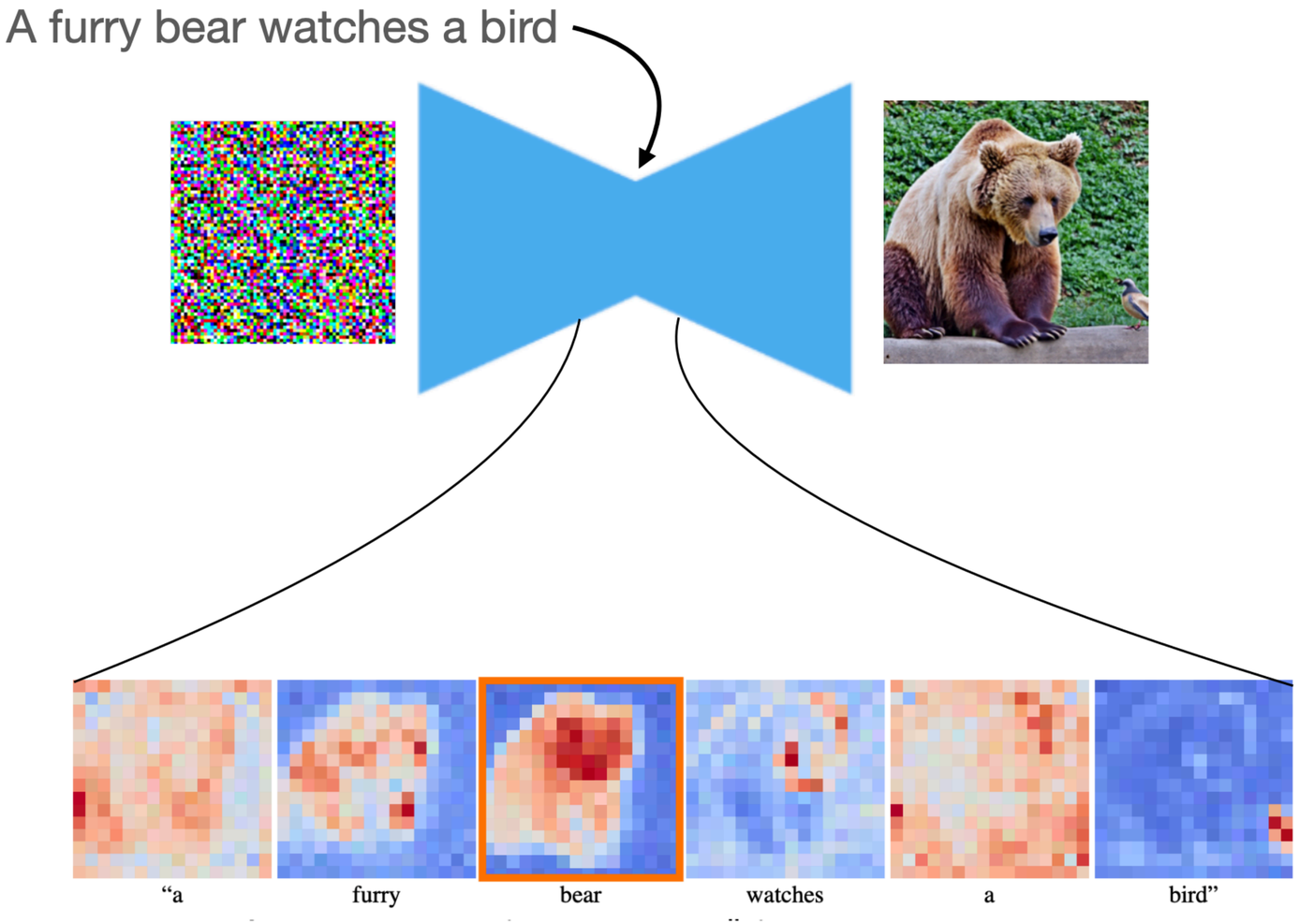


"Photo of a cat riding on a ~~bicycle~~ car."

Hertz et al. "Prompt-to-Prompt Image Editing with Cross-Attention Control"

Adapted from A. Hołyński

What's happening inside the network?

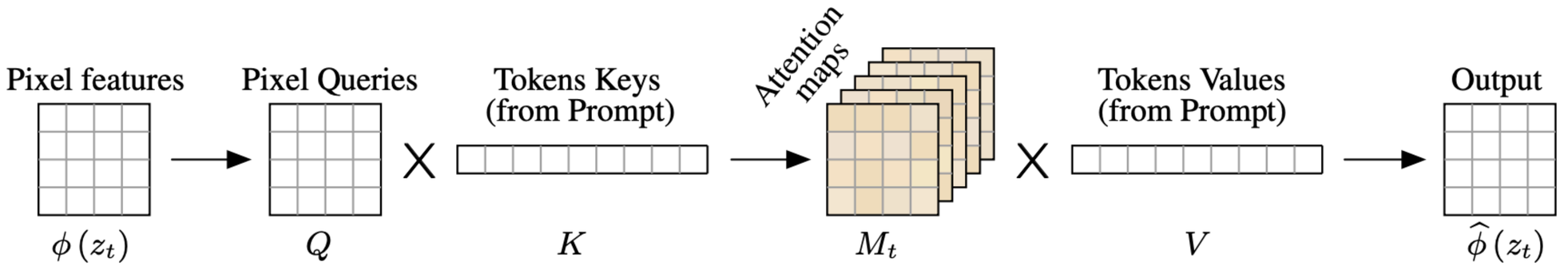


Hertz et al. "Prompt-to-Prompt Image Editing with Cross-Attention Control"

Slide source: A. Hołyński

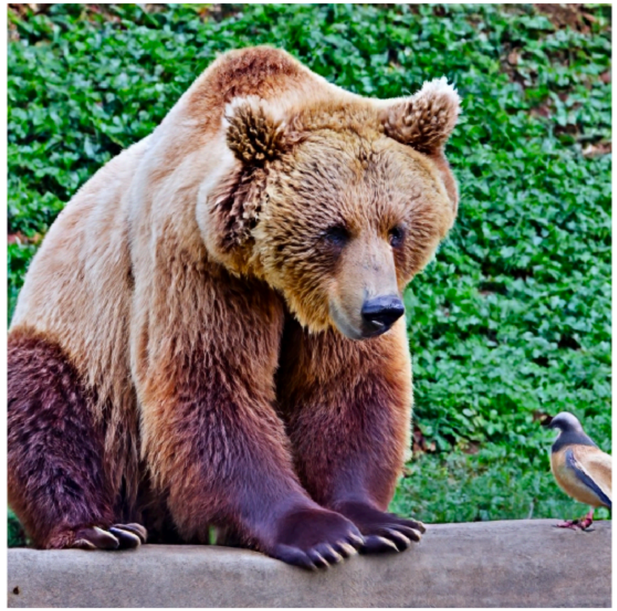
Attention visualization

Empirical observation: cross-attention between text and image often conveys style, content, and structure.

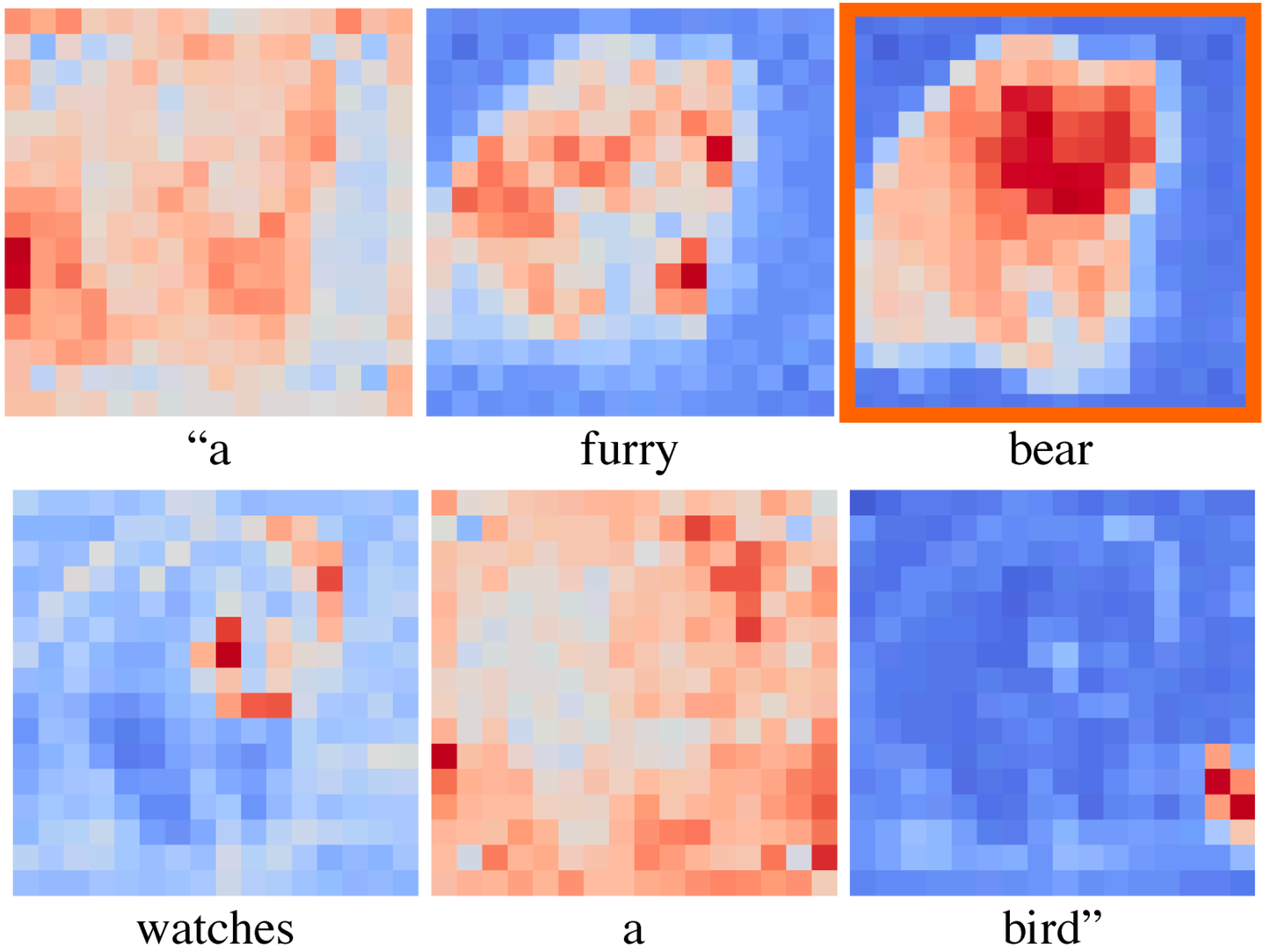


Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

Attention visualization

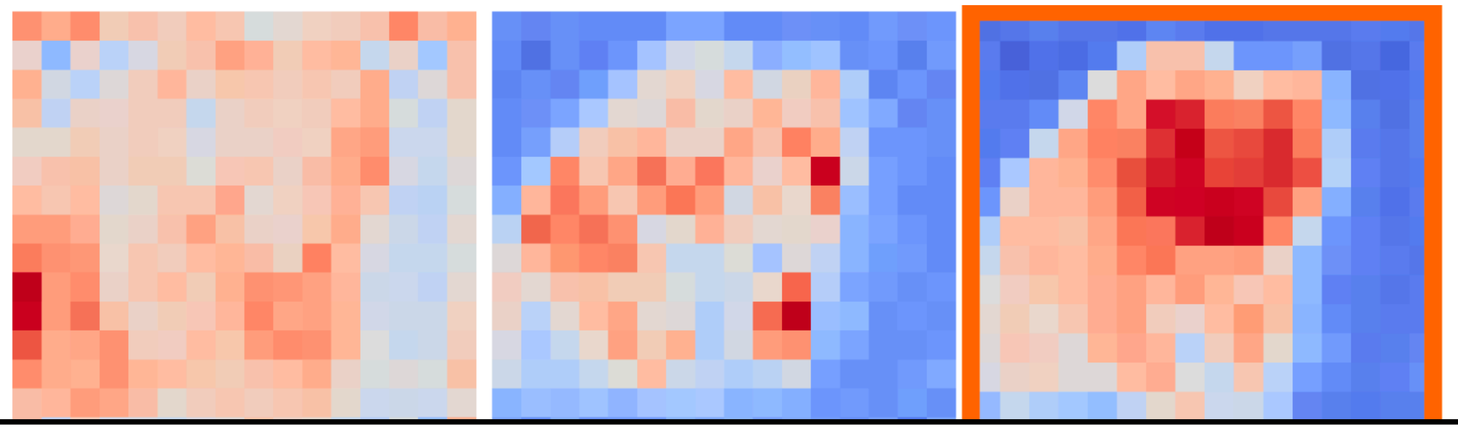


“a furry bear watches a bird”



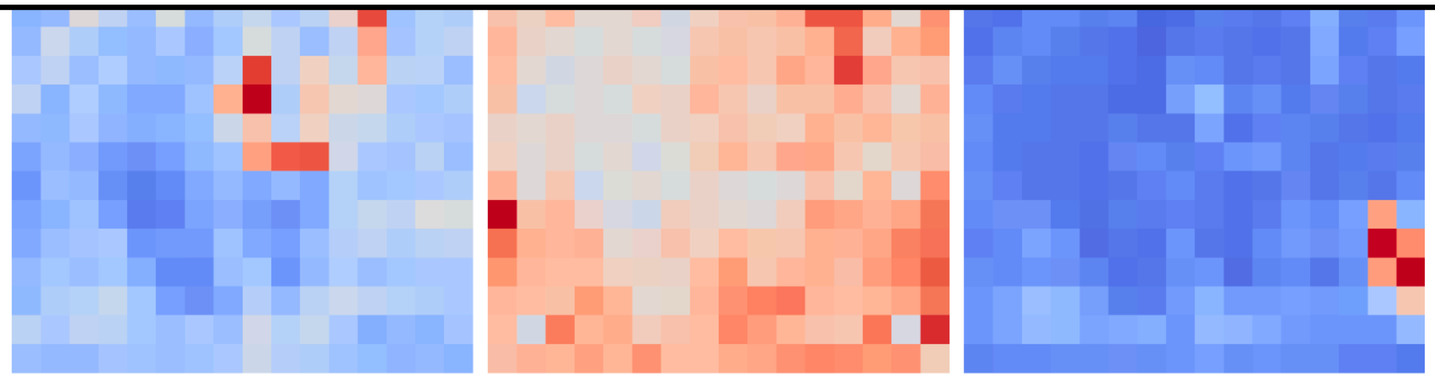
Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

Attention visualization



What if we manipulate the attention maps?

"a furry bear watches a bird"



Hertz et al. "Prompt-to-Prompt Image Editing with Cross-Attention Control"

Change text prompt, resample using same random seed.

“lemon cake.”

“cheese cake.”

“apple cake.”

“pumpkin cake.”

“lego cake.”

“beet cake.”

“pepperoni cake.”

Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

What if we also freeze the attention maps?

“lemon cake.”

“cheese cake.”

“apple cake.”

“pumpkin cake.”

“lego cake.”

“beet cake.”

“pepperoni cake.”

Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

Freeze the attention map for “apples” or “basket” during generation.

“A basket full of apples.”

Source image

apples → cookies

apples → oranges

apples → chocolates

apples → kittens

basket → bowl

basket → box

basket → nest

Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

“A photo of a butterfly on a flower.”

This is a neat trick. Can we train a *model* that captures these abilities?

Hertz et al. “Prompt-to-Prompt Image Editing with Cross-Attention Control”

Prompt-to-Prompt

"Photo of a cat riding on a bicycle."

source image

cat → dog

cat → chicken

cat → squirrel

cat → elephant

Hertz et al. "Prompt-to-Prompt Image Editing with Cross-Attention Control"

Bootstrapping to instruction-based image editing

“Swap sunflowers with roses”

“Add fireworks to the sky”

“Replace the fruits with cake”

“What would it look like if it were snowing?”

“Turn it into a still from a western”

“Make his jacket out of leather”

Generating training data for instruction-based editing

Source: [Brooks et al., "InstructPix2Pix", 2023]

Inpainting (hole filling)

Input

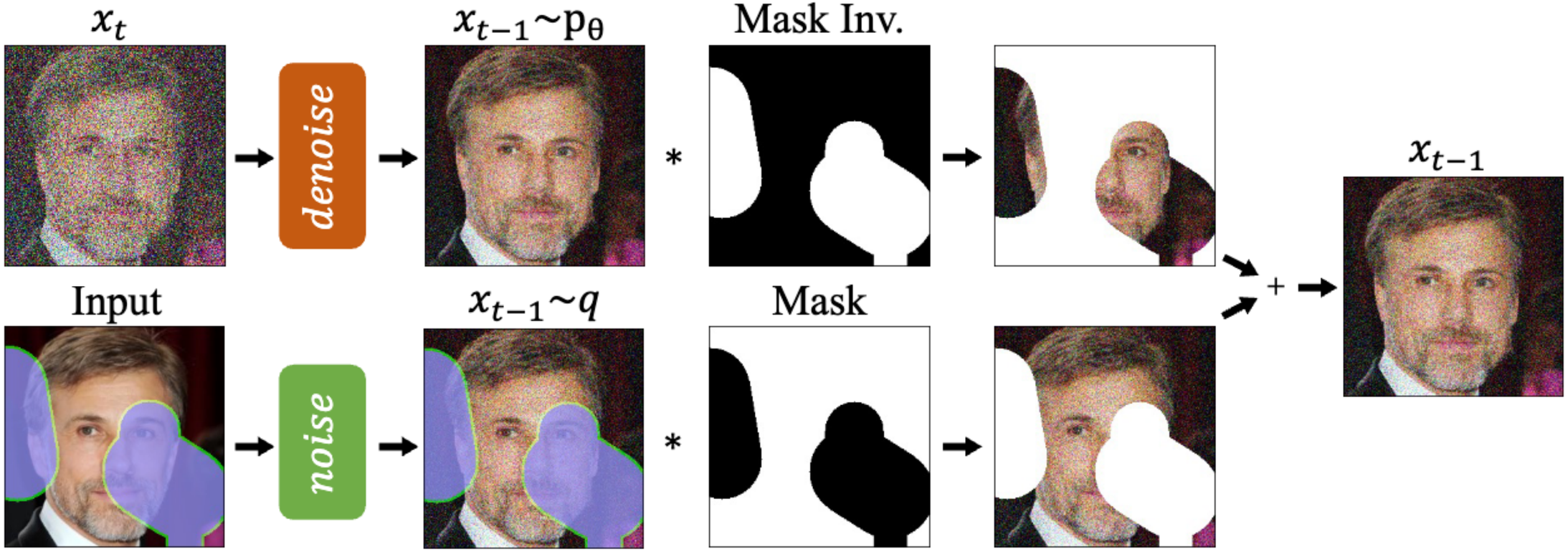
Samples

Do we need a new model for this?

Source: [Lugmayr et al., RePaint]

Zero-shot inpainting

Constrain the pixels each denoising iteration.



Source: [Lugmayr et al., RePaint]

Zero-shot inpainting

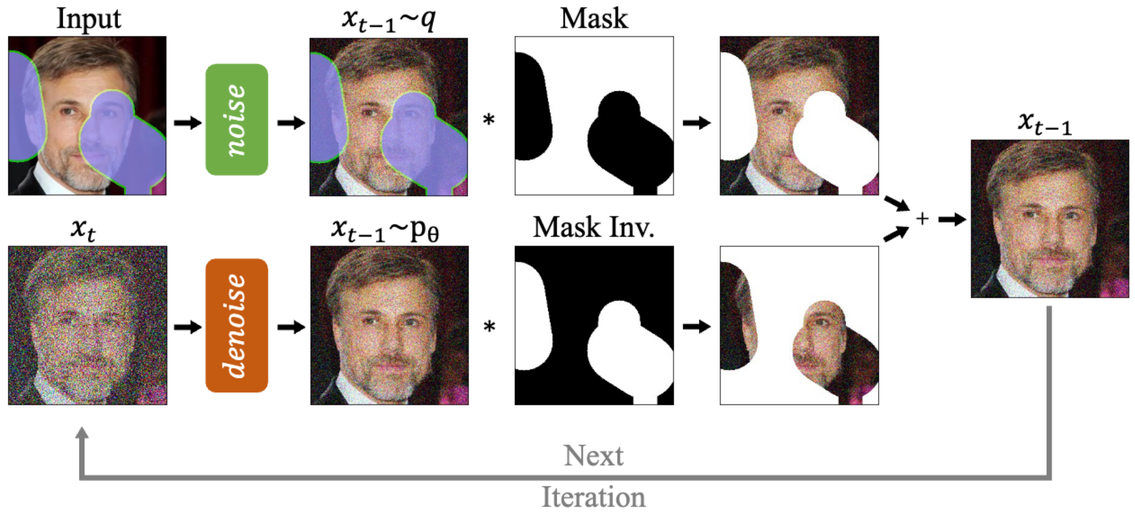
Pseudocode:

$$x_{t-1}^{\text{unknown}} \sim \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

$$x_{t-1}^{\text{known}} \sim \mathcal{N}(\sqrt{\bar{\alpha}_t} x_0, (1 - \bar{\alpha}_t)\mathbf{I})$$

$$x_{t-1} = m \odot x_{t-1}^{\text{known}} + (1 - m) \odot x_{t-1}^{\text{unknown}}$$

Constrain pixels each iteration.



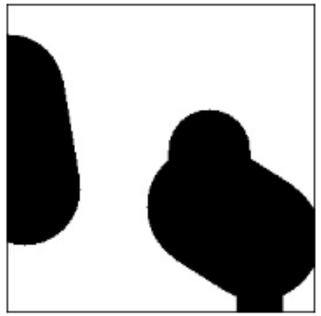
Source: [Lugmayr et al., RePaint]

Inverse problems

- Inpainting is an example of a *linear inverse problem*.
- We want to recover an unknown image $\mathbf{x} \in \mathbb{R}^n$ $\mathbf{x} \sim p(\mathbf{x})$, given measurements $\mathbf{y} \in \mathbf{R}^m$ obtained by corrupting $\mathbf{x}$ by a linear transformation $A \in \mathbb{R}^{m \times n}$:

$$\mathbf{y} = A\mathbf{x}$$

masked image

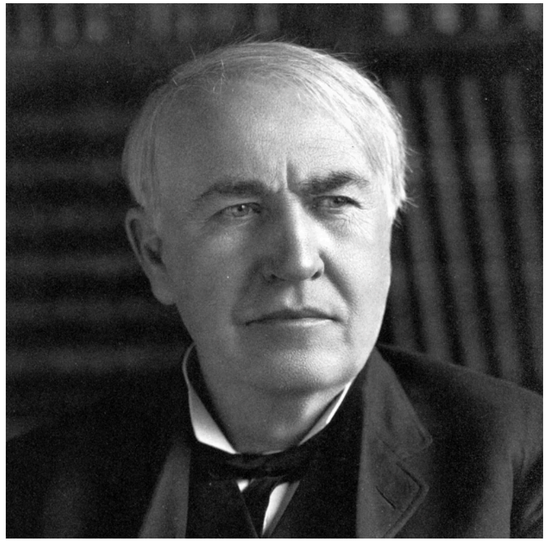


mask (as a matrix)

unknown image
(with holes filled)

- We solved this by tweaking the reverse diffusion process.
- This approach works for inverse problems, too!

Recovering the high frequencies of an image



reference image $\mathbf{x}$

=

low pass $\mathbf{y}$

+

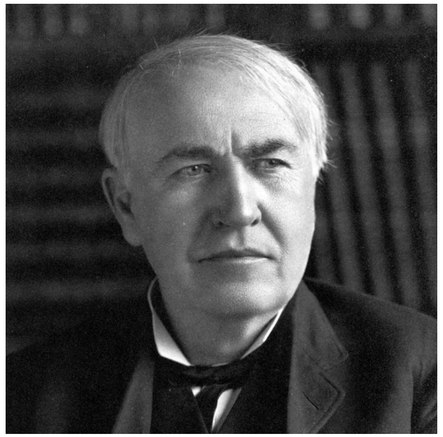
high pass

Break an image down into “low pass” (by blurring) and “high pass” (everything else).
Since blurring is a linear transformation, we can write this as a linear inverse problem:

$$\mathbf{y} = A_{\text{blur}}\mathbf{x}$$

[Choi et al., “ILVR”, 2021]

Recovering the high frequencies of an image

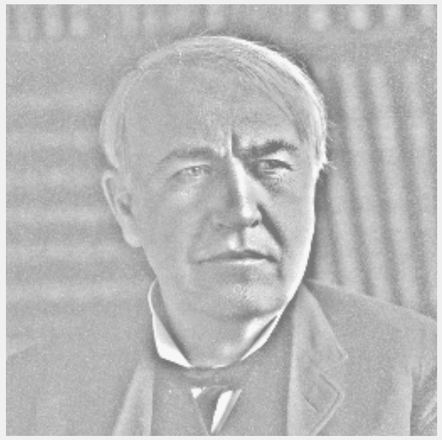


reference image $\mathbf{x}$

=

low pass $\mathbf{y}$

+



high pass

Each iteration of the reverse diffusion process:

- Generate $\mathbf{x}_{t-1}$ from $\mathbf{x}_t$ as usual (by denoising).
- Force the low frequencies of $\mathbf{x}_{t-1}$ to match those of the reference image

$$\mathbf{x}'_{t-1} = \left(\mathbf{x}_{t-1} - A_{\text{blur}}\mathbf{x}_{t-1}\right) + \text{add\_noise}(\mathbf{y})$$

[Choi et al., “ILVR”, 2021]

Recover the high frequencies, freezing various numbers of low frequencies.

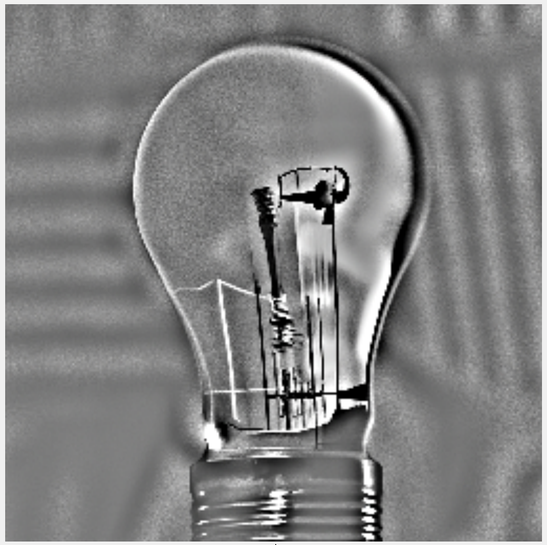
Figure source: [Choi et al., “ILVR”, 2021]

What if we use a very different prompt?

=

low pass

+



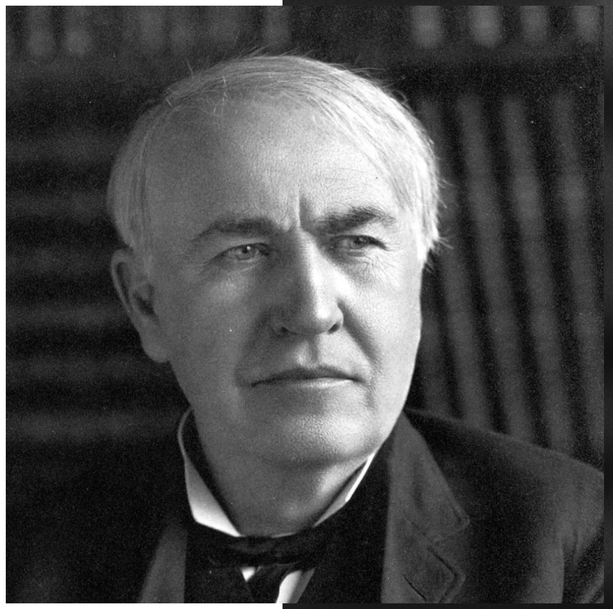
diffusion model

"a photo of a lightbulb"

Daniel Geng

Aaron Park

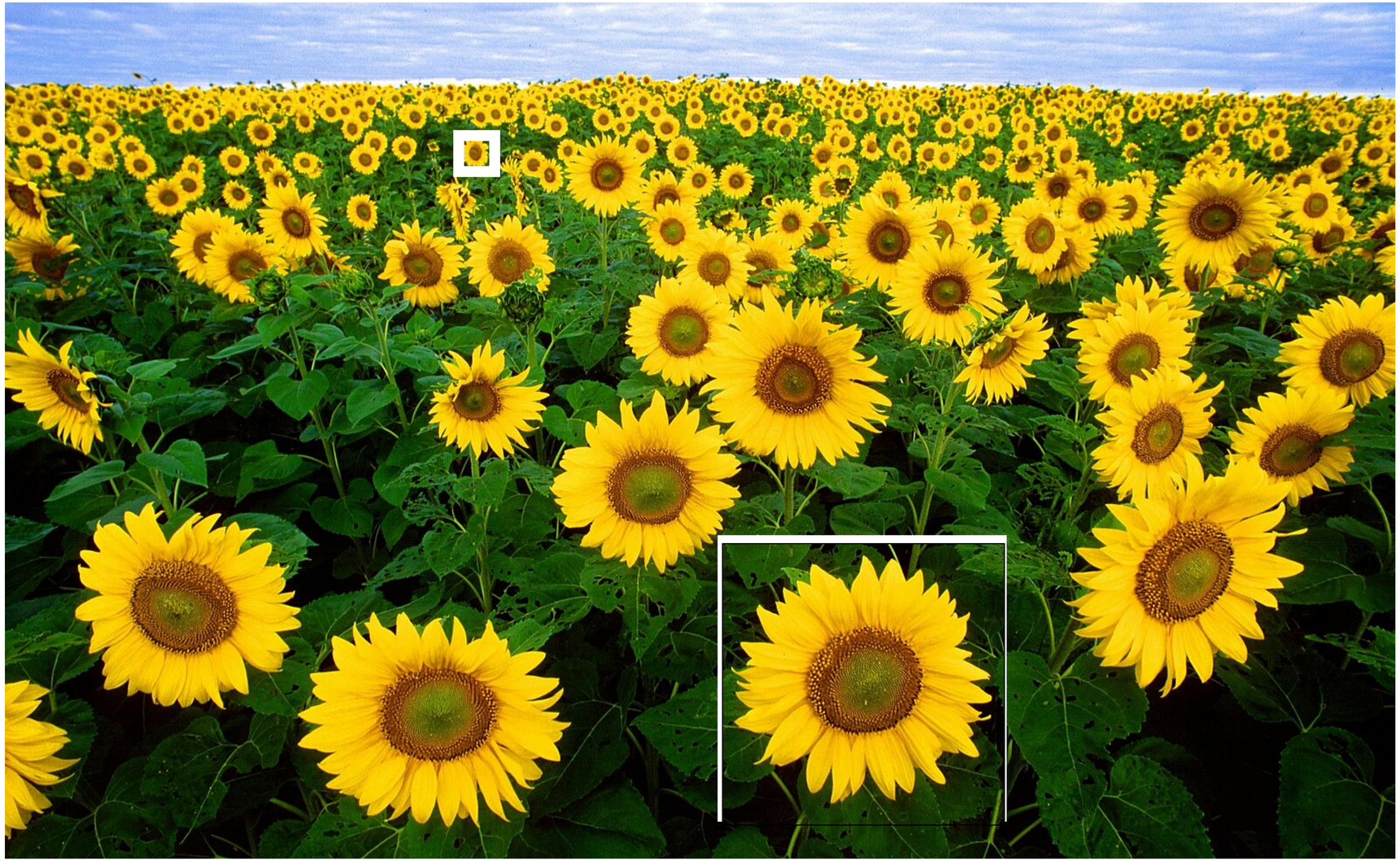
[Geng et al., "Factorized Diffusion", 2024]



These are examples of **hybrid images:** images that change their appearance when seen from a distance [Oliva et al., 2006].

Charles Allan Gilbert
All Is Vanity. 1892.

Why does this work?

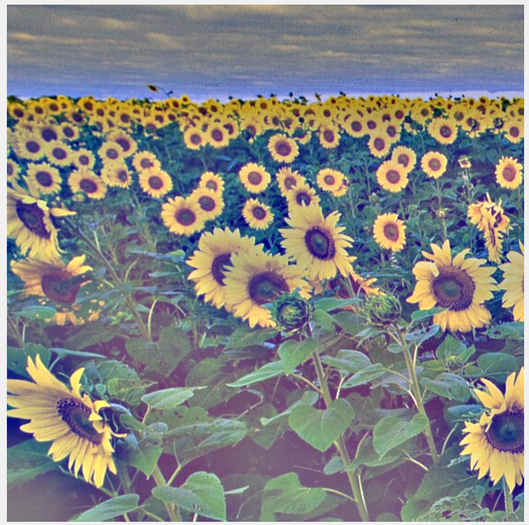


Frequency and distance

linear transformations

Decomposition into linear factors

=



near

+

in between

+

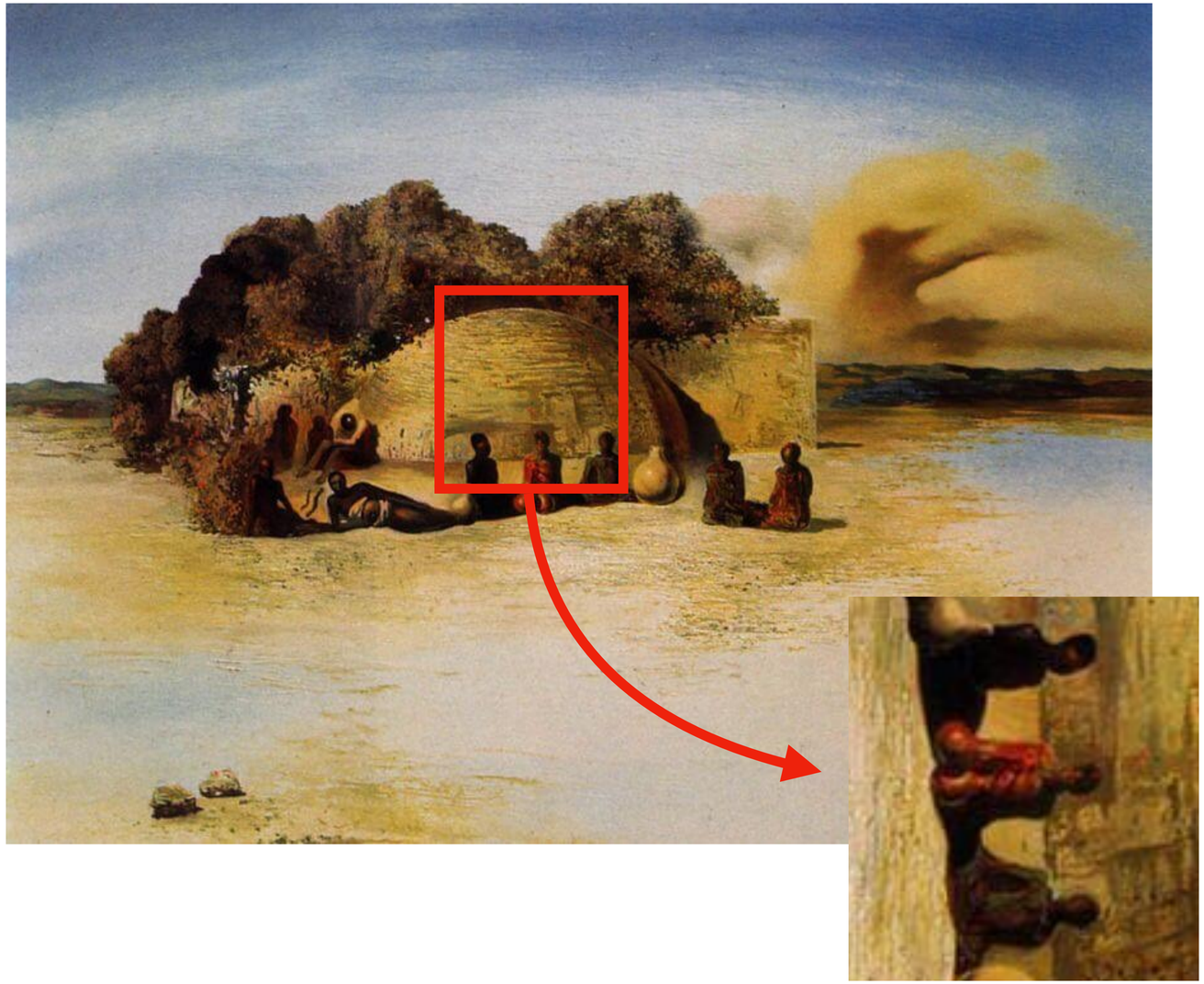
far

[Burt & Adelson, “Laplacian Pyramid”, 1983, Oliva et al., “Hybrid Images”, 2006]

Thanks to Walter Scheirer and Luba Elliot for suggesting this!

What about other types of illusions?

Salvador Dalí,
Paranoiac Face. 1937.

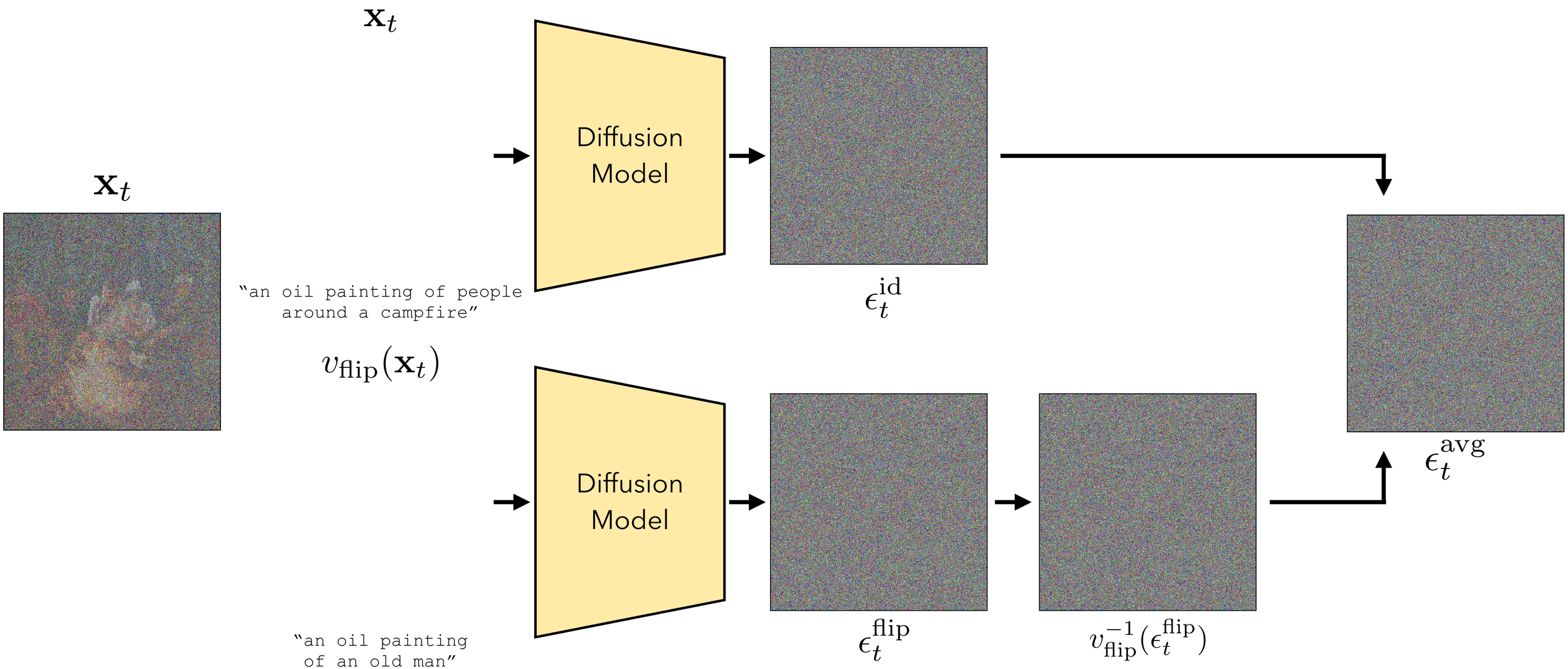


Daniel Geng

Aaron Park

[Visual Anagrams: Generating Multi-View Optical Illusions with Diffusion Models, CVPR 2024]

Generating an illusion



Generating an illusion

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

$\mathbf{x}_0$

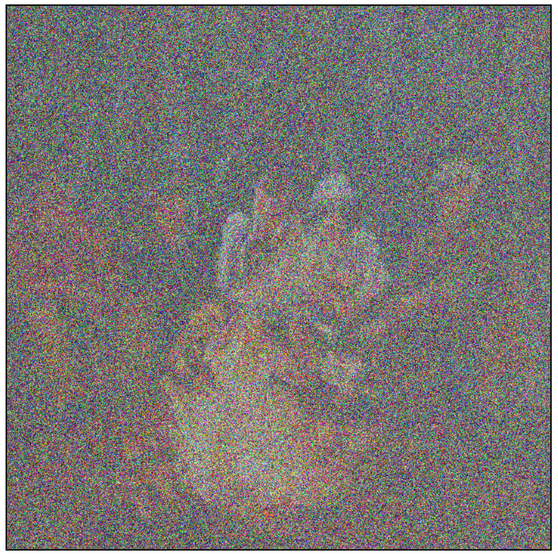
an oil painting of people around a campfire

180°

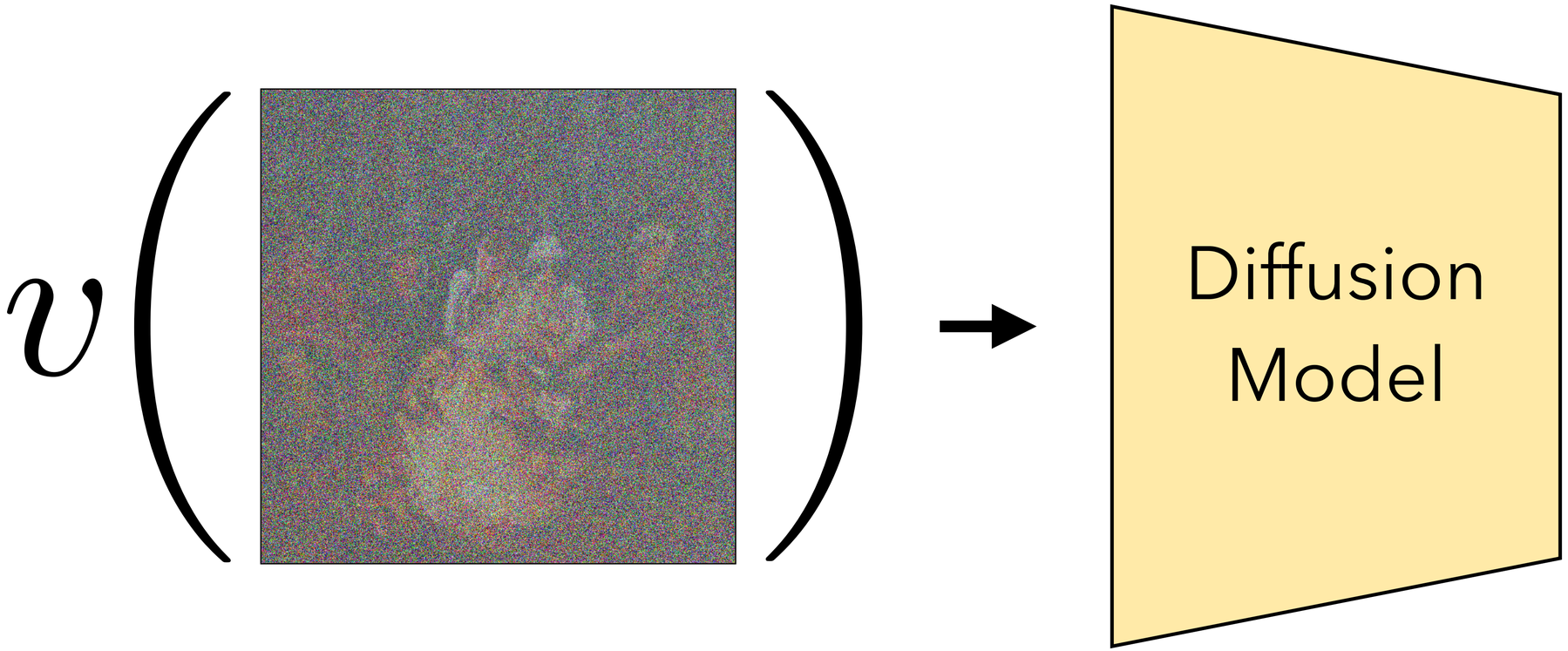
a pop art of
albert einstein

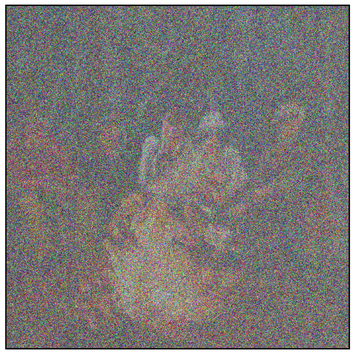
135°

a pop art of
albert einstein



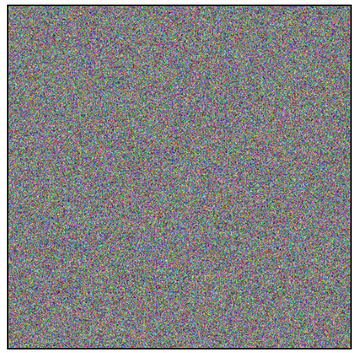
Diffusion Model


$v($
$)$
Diffusion
Model



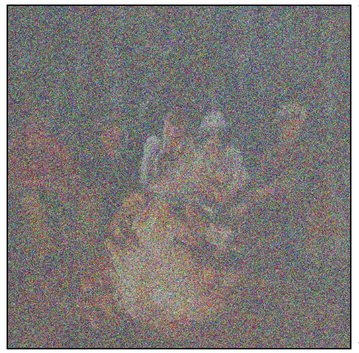
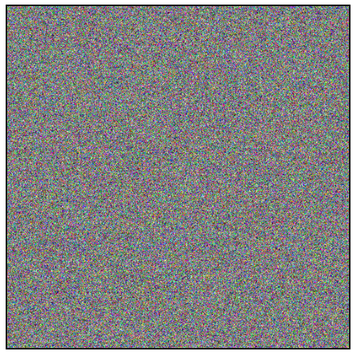
$$= w_t^{\text{signal}}$$

$$+ w_t^{\text{noise}}$$

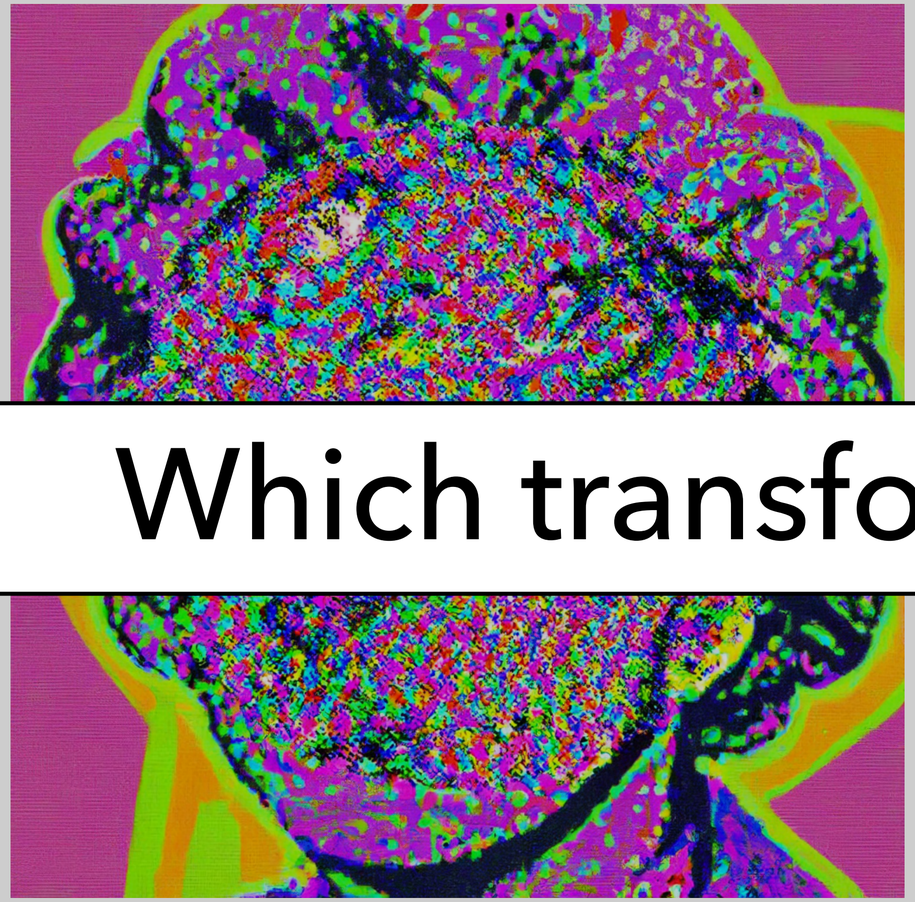


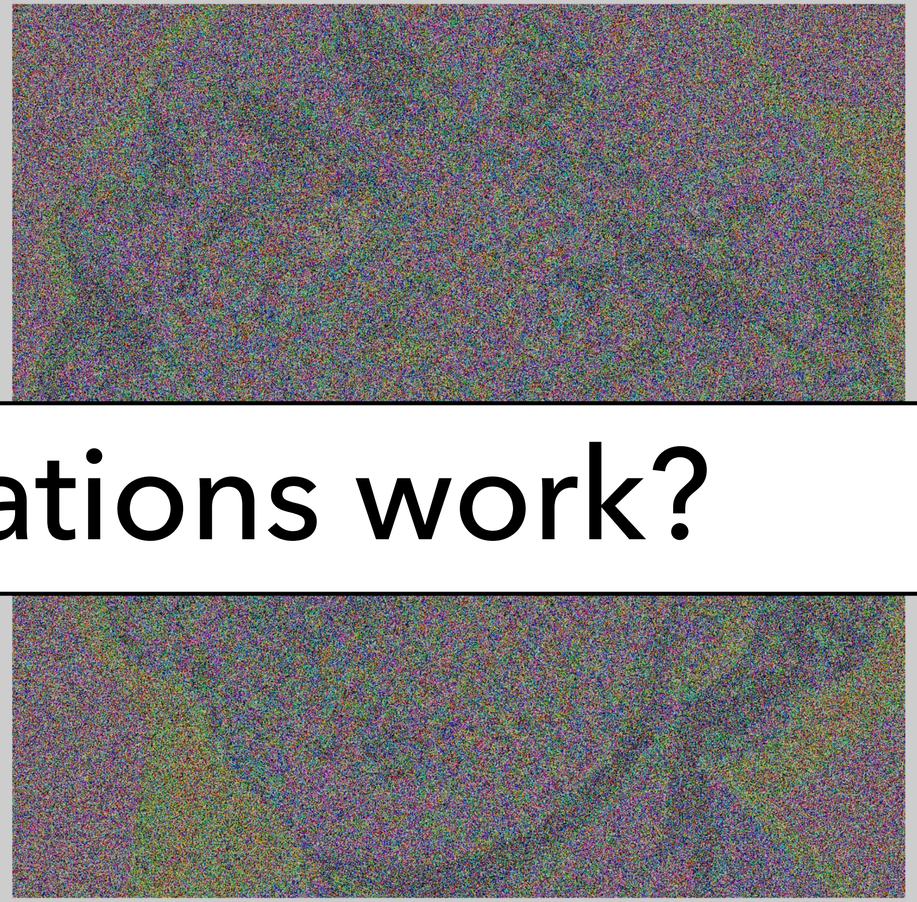
$$\sim \mathcal{N}(0, \mathbf{I})$$

1. Noisy image is weighted sum of signal and noise
2. Noise must be i.i.d. standard Gaussian

$$v\left(\text{  }\right) = v\left(w_t^{\text{signal}} \text{  } + w_t^{\text{noise}} \underbrace{\text{  }}_{\sim \mathcal{N}(0, \mathbf{I})}\right)$$

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian



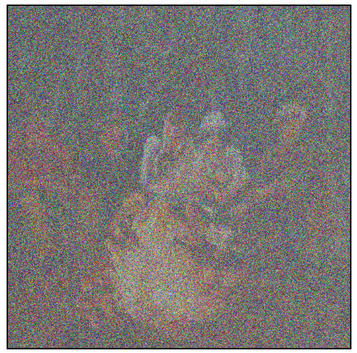


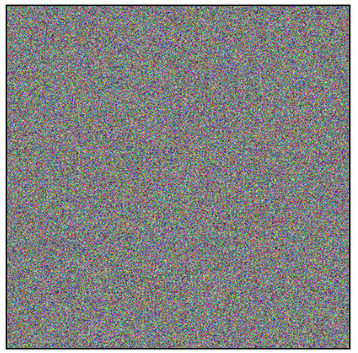
Which transformations work?

a pop art of
albert einstein

$\mathbf{x}_t$

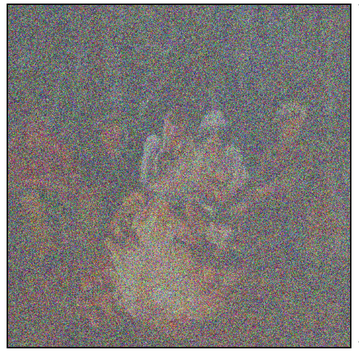
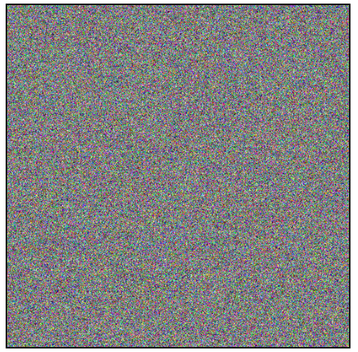
$$v\Big(\;\Big) = v\Big(w_t^{\text{signal}}\;\; + w_t^{\text{noise}}\;\;\Big)$$





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

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian

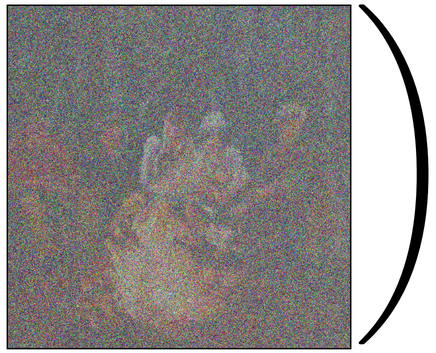
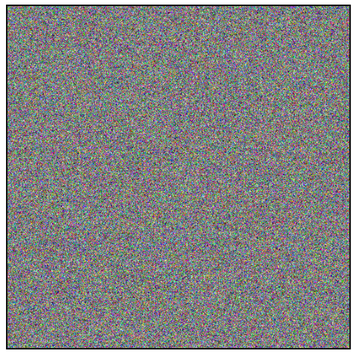
$$v\Big(\text{  }\Big) = \mathbf{A}\Big(w_t^{\text{signal}}\ \text{  }\ + w_t^{\text{noise}}\ \text{  }\Big)$$

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

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian

Transformed Signal

Transformed Noise

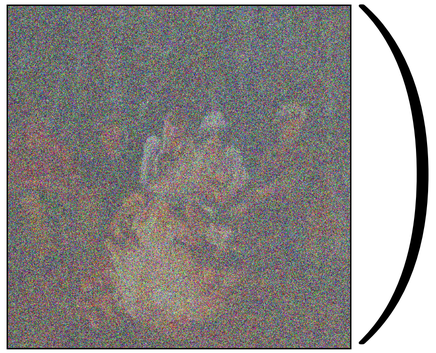
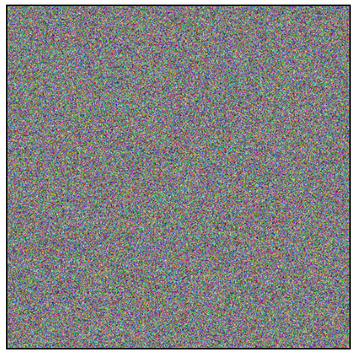
$$v\left(\text{  }\right) = w_t^{\text{signal}}\,\mathbf{A}\,\text{  } + w_t^{\text{noise}}\,\mathbf{A}\,\text{  }$$

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

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian

Transformed Signal

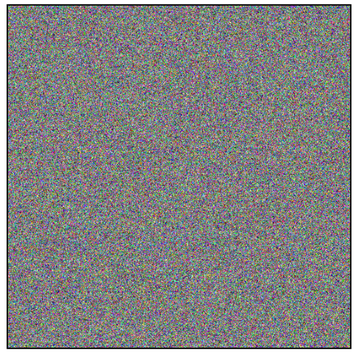
Transformed Noise

$$v\left(\text{  }\right) = w_t^{\text{signal}}\,\mathbf{A}\,\text{  } + w_t^{\text{noise}}\,\mathbf{A}\,\text{  }$$

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

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian

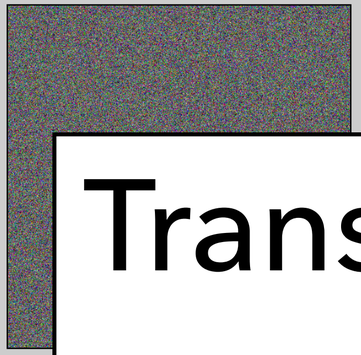
Transformed Noise

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

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

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian

Transformed
Noise



A

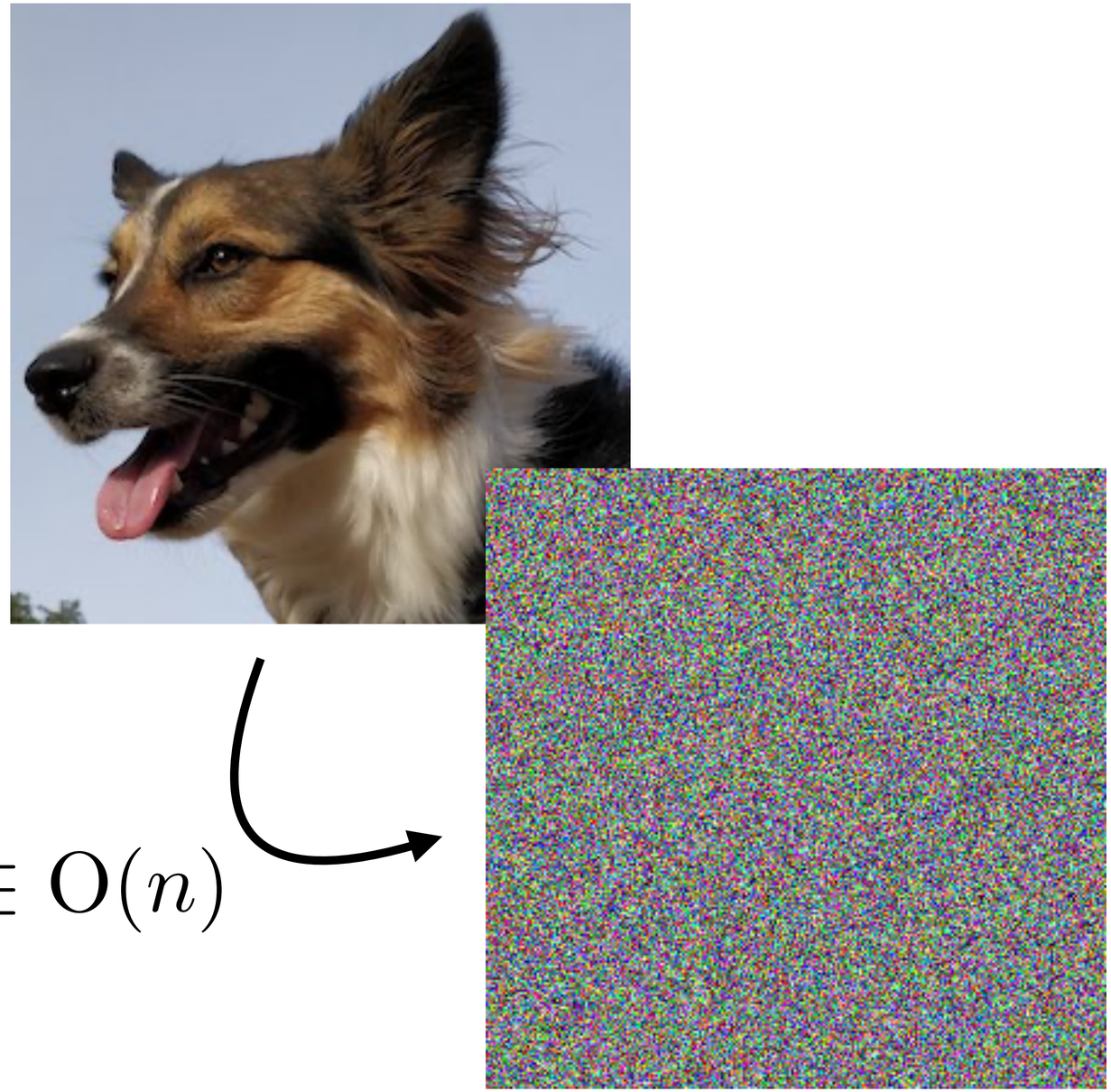
nal

Transformations must be orthogonal

1. Noisy image is weighted sum of *signal* and *noise*
2. Noise must be i.i.d. standard Gaussian

Most orthogonal transformations are uninterpretable…

… but any permutation is orthogonal!



Visual Anagrams

An image that changes appearance
under a permutation of its pixels

(an ambigram!)

180° Rotations

the word "happy",
cursive writing

See Also: Boheng Zhao et al. “AmbiGen,” arXiv 2023

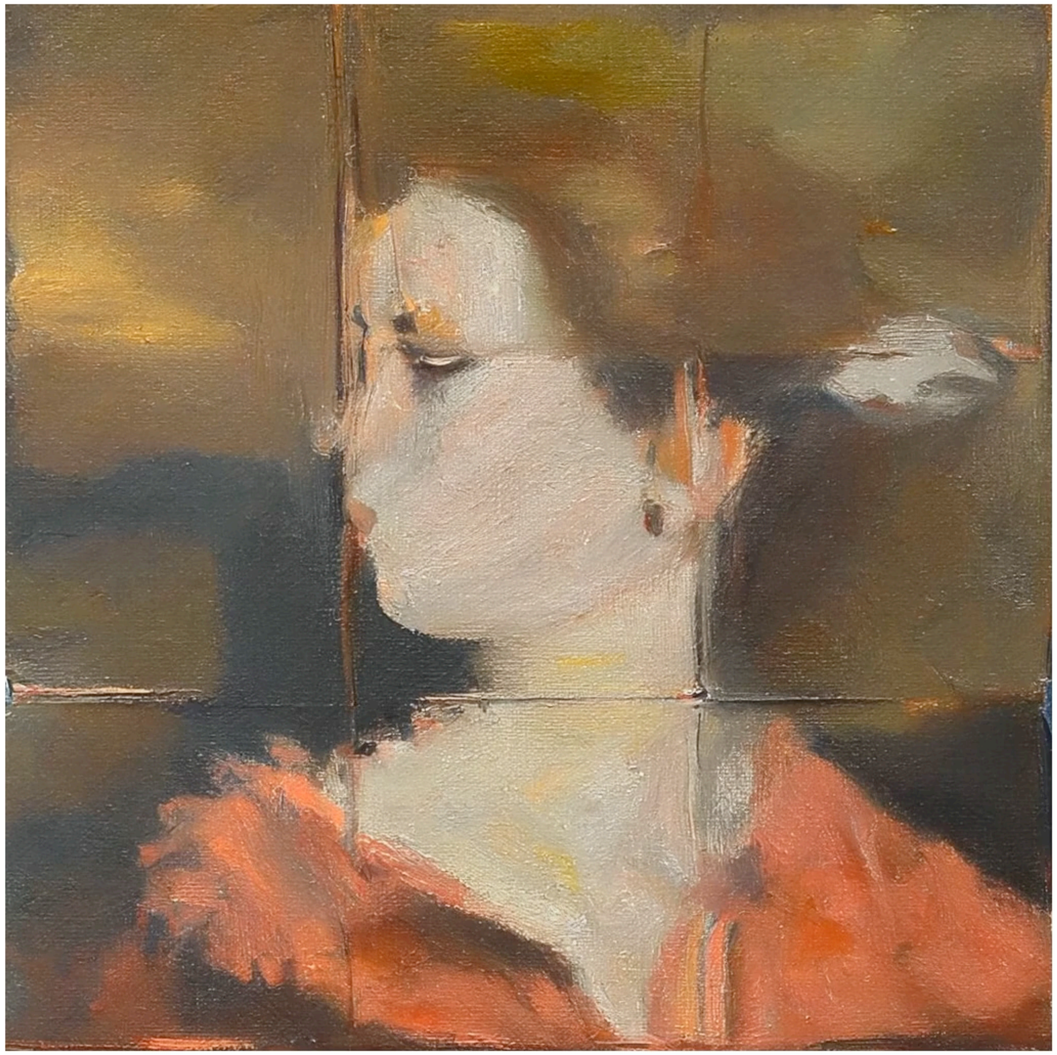
90° Rotations

a lithograph of a table

“Inner Circle”

an oil painting
of an old man

“Square Hinge”



an oil painting
of a young lady

Thank you to Ryan Burgert for this idea! Inspired by an illusion from Steve Mould.

“Polymorphic” Jigsaw Puzzles

a watercolor of a kitten

“Polymorphic” Jigsaw Puzzles

a watercolor of a rabbit

Real puzzles!

Patch Permutations

a pencil sketch of a kangaroo

Hans Holbein, *The Ambassadors*. 1533

Skews

Skews

an oil painting
of a skull

an oil painting of
a tudor portrait

Inversions $v(\mathbf{x}) = -\mathbf{x}$

Inversions

a lithograph of a landscape

Why does this work?

Compositionality

By learning the score…

$$\epsilon_\theta(\mathbf{x}_t) \propto -\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$$

…you get many other scores for free

$$\nabla_{\mathbf{x}_t} \log p_{\text{flipped},t}(\mathbf{x}_t)$$

$$\nabla_{\mathbf{x}_t} \log p_{\text{rotated},t}(\mathbf{x}_t)$$

$$\nabla_{\mathbf{x}_t} \log p_{\text{permuted},t}(\mathbf{x}_t)$$

$$\vdots$$

Compositionality

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_{\text{flipped},t}(\mathbf{x}_t)$$

$$= \nabla_{\mathbf{x}_t} \log \left(p_t(\mathbf{x}_t) p_{\text{flipped},t}(\mathbf{x}_t) \right)$$

$$\Longrightarrow$$

$$\mathbf{x} \sim p^{\text{prod}}(\mathbf{x}) \propto p(\mathbf{x}) p_{\text{flipped}}(\mathbf{x})$$

Very similar to composition in energy-based models, where you can add the energy functions together.

Nan Liu, Shuang Li, Yilun Du, et al. “Compositional Visual Generation.” ECCV 2022
Yilun Du, et al. “Reduce, Reuse, Recycle.” ICML 2023

Other applications of compositional generation

(a) Composing Language Descriptions (Composed Stable Diffusion)

"A photo of cherry blossom trees" AND "Sun dog" AND "Green grass"

"A church" AND "Lightning in the background" AND "A beautiful pink sky"

"A stone castle surrounded by lakes and trees," AND "Black and white"

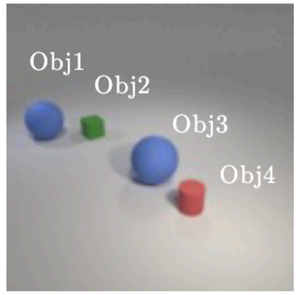
"A stone castle surrounded by lakes and trees," AND (NOT "Black and white")

"A mystical tree " AND "A dark magical pond" AND "Dark"

"A mystical tree " AND "A dark magical pond" AND (NOT "Dark")

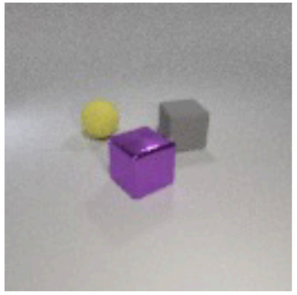
(c) Composing Objects

Obj1 (0.1, 0.5) AND Obj2 (0.5, 0.3) AND Obj3 (0.5, 0.65) AND Obj4 (0.7, 0.5)

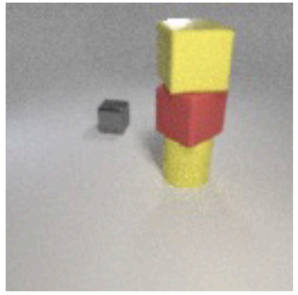


Obj1 (0.1, 0.65) AND Obj2 (0.3, 0.55) AND Obj3 (0.5, 0.45) AND Obj4 (0.7, 0.3)

(d) Composing Object Relations



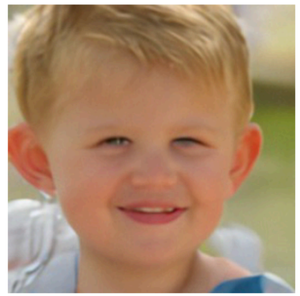
"A large purple metal cube to the left of a large gray rubber cube" AND "A large purple metal cube to the right of a large yellow rubber sphere"



"A large yellow rubber cylinder to the right of a small gray metal cube" AND "A large yellow rubber cylinder below a large red rubber cube"

(e) Composing Facial Attributes

(NOT Female) AND Smiling AND (NOT Glasses)

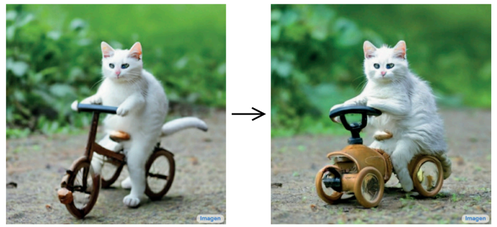


Male AND Blonde hair AND (NOT glasses)

Source: Nan Liu, Shuang Li, Yilun Du, et al. "Compositional Visual Generation." ECCV 2022

Zero shot image manipulation via diffusion: summary

- Image translation by denoising from intermediate noise levels.
- Manipulate cross-attention for prompt-based image editing.
- Fill holes by constraining pixels during denoising.
- This approach generalizes to other inverse problems.
- Compose diffusion models by averaging their noise estimators.



"Photo of a cat riding on a bicycle."

an oil painting of people around a campfire

Have a great spring break!