Diffusion Models for Visual Content Generation

Guidance

Presenter: Minhyuk Sung

Eurographics 2024 Tutorial

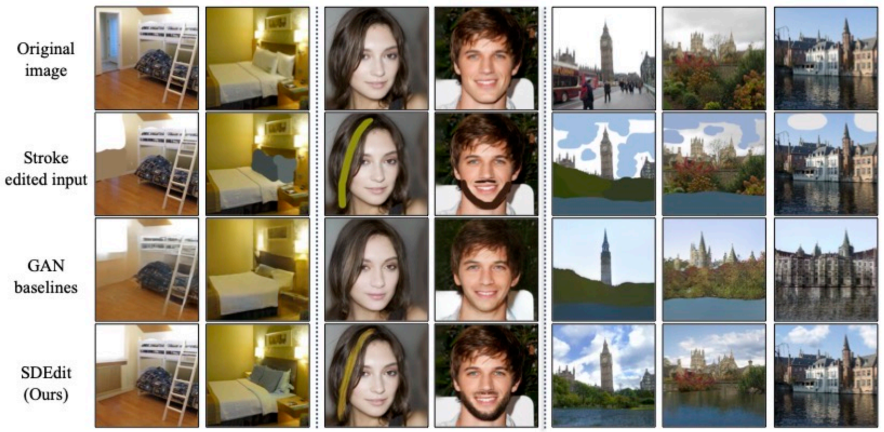
Guidance Using Additional Information

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?

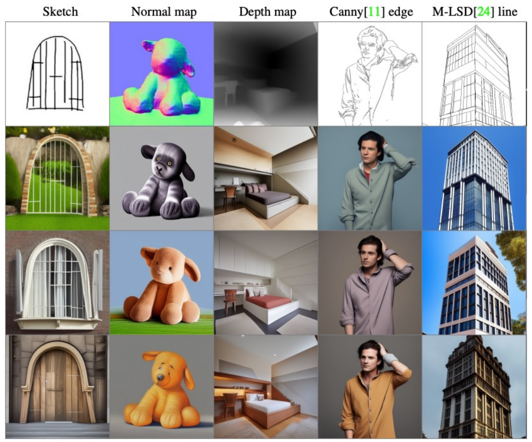
Ho and Salimans, Classifier-Free Diffusion Guidance, NeurIPS 2021 Workshop.

Zero-Shot / Few-Shot Adaptations

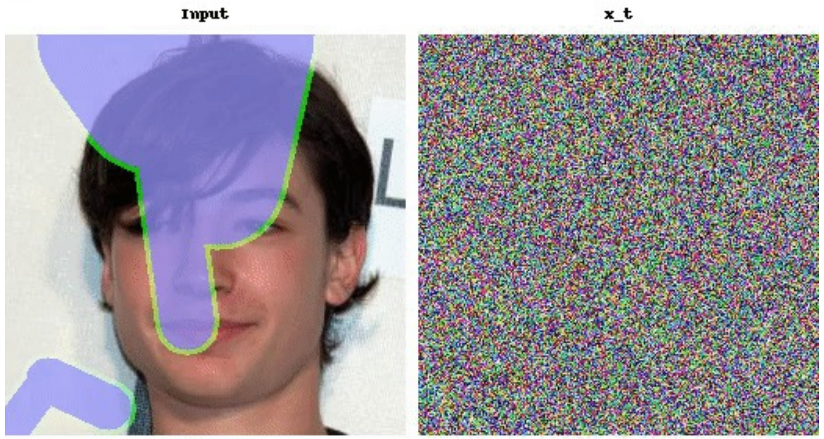
Can we apply a pretrained diffusion model to various conditional generation setups in a zero-shot or few-shot manner?



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?

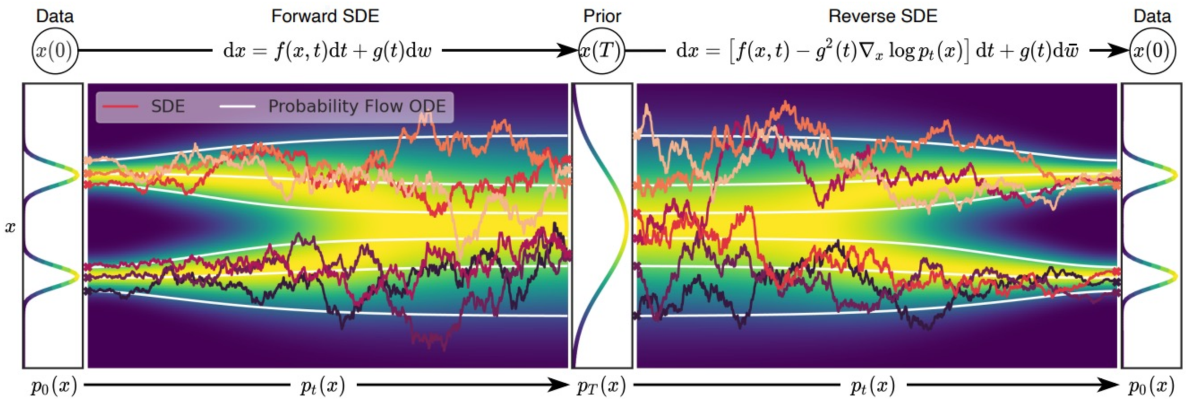
Ho and Salimans, Classifier-Free Diffusion Guidance, NeurIPS 2021 Workshop.

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.

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

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

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 $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t, y)$:

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

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

$$= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \left(\underbrace{\boxed{\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, t)}}_{\text{Noise predictor}} - \sqrt{1-\bar{\alpha}_t} \underbrace{\boxed{\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t)}}_{\text{How to compute this?}} \right)$$

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:

$$\hat{\boldsymbol{\varepsilon}}_\theta(\mathbf{x}_t, y, t) = \underbrace{(1+w)\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, y, t)}_{\text{Increase}\uparrow\text{ the conditional noise}} - \underbrace{w\boldsymbol{\varepsilon}_\theta(\mathbf{x}_t, \emptyset, t)}_{\text{Decrease}\downarrow\text{ the unconditional noise}}$$

$$w \geq 0$$

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 ($T = 256$)
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

ImageNet Results

Classifier-Free Guidance (CFG)

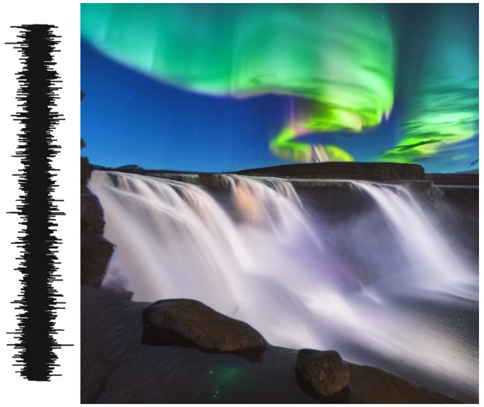
- $(+)$ The classifier does not need to be trained.
- $(+)$ It is more versatile and can be used not only for class labels but also for any additional information (e.g., text descriptions).
- $(-)$ In the generation process, the noise predictor needs to be evaluated twice.
- $(-)$ Determining the optimal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

Example: Audio Conditioned Generation

Example: Audio Conditioned Generation

Text Prompt: *"Aurora Borealis Lights"*

Waterfall Burbling

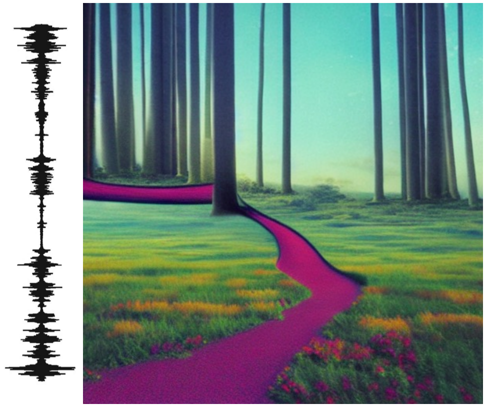


Text Prompt: *"Lego"*

Snow

Text Prompt: *"Surreal Dreamscapes"*

Forest



Example: Two-Hand Interaction Generation

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

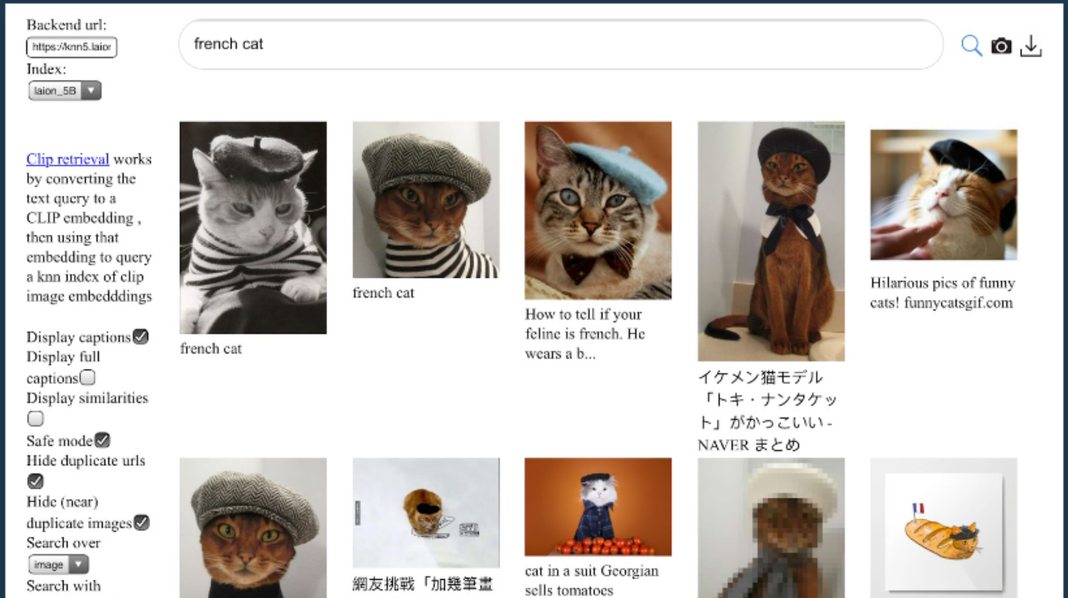
ControlNet:
Few-Shot Adaptation

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

by: Romain Beaumont, 31 Mar, 2022

We present a dataset of 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



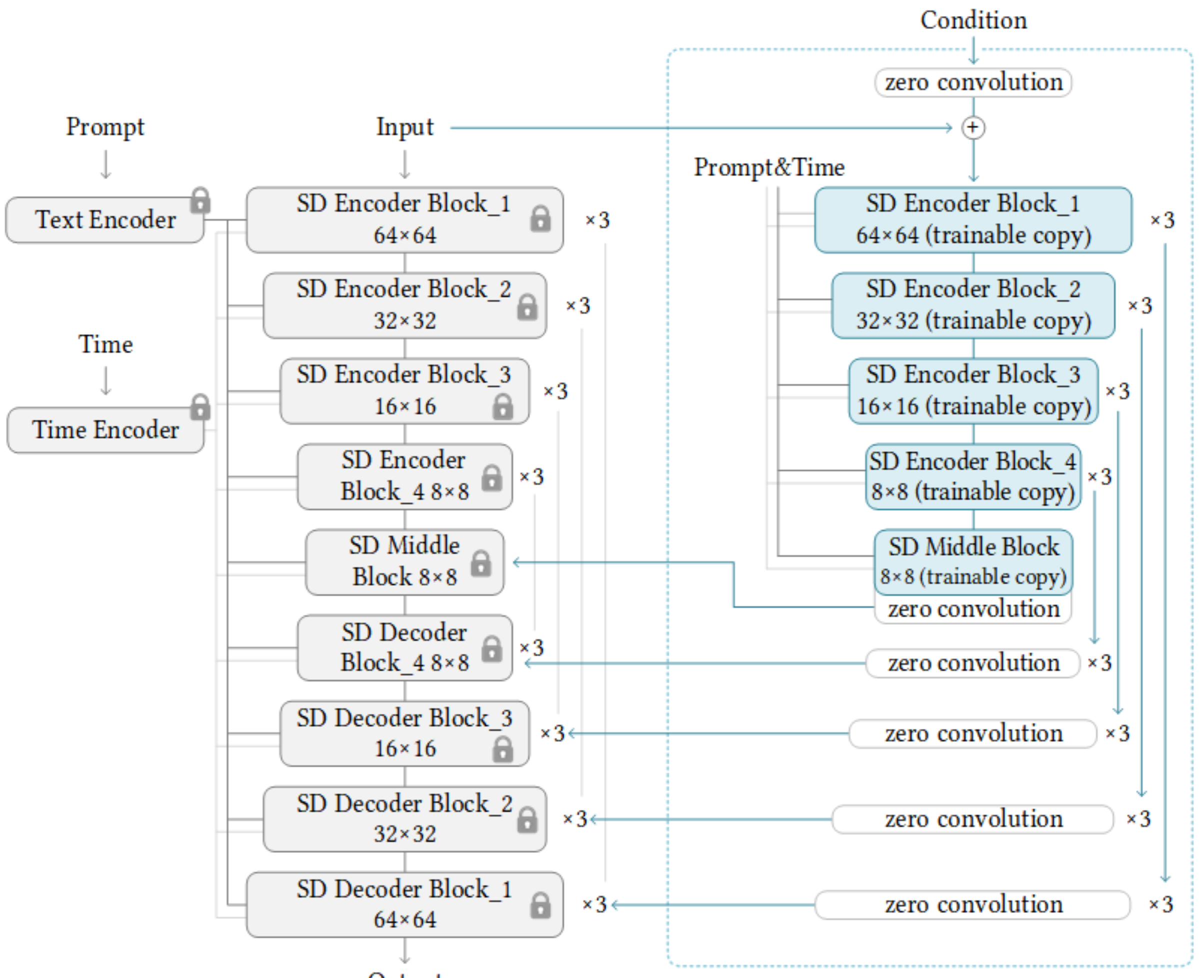
ControlNet [Zhang et al., 2023]

- 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?

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 ($\sim 100k$)?

ControlNet [Zhang et al., 2023]

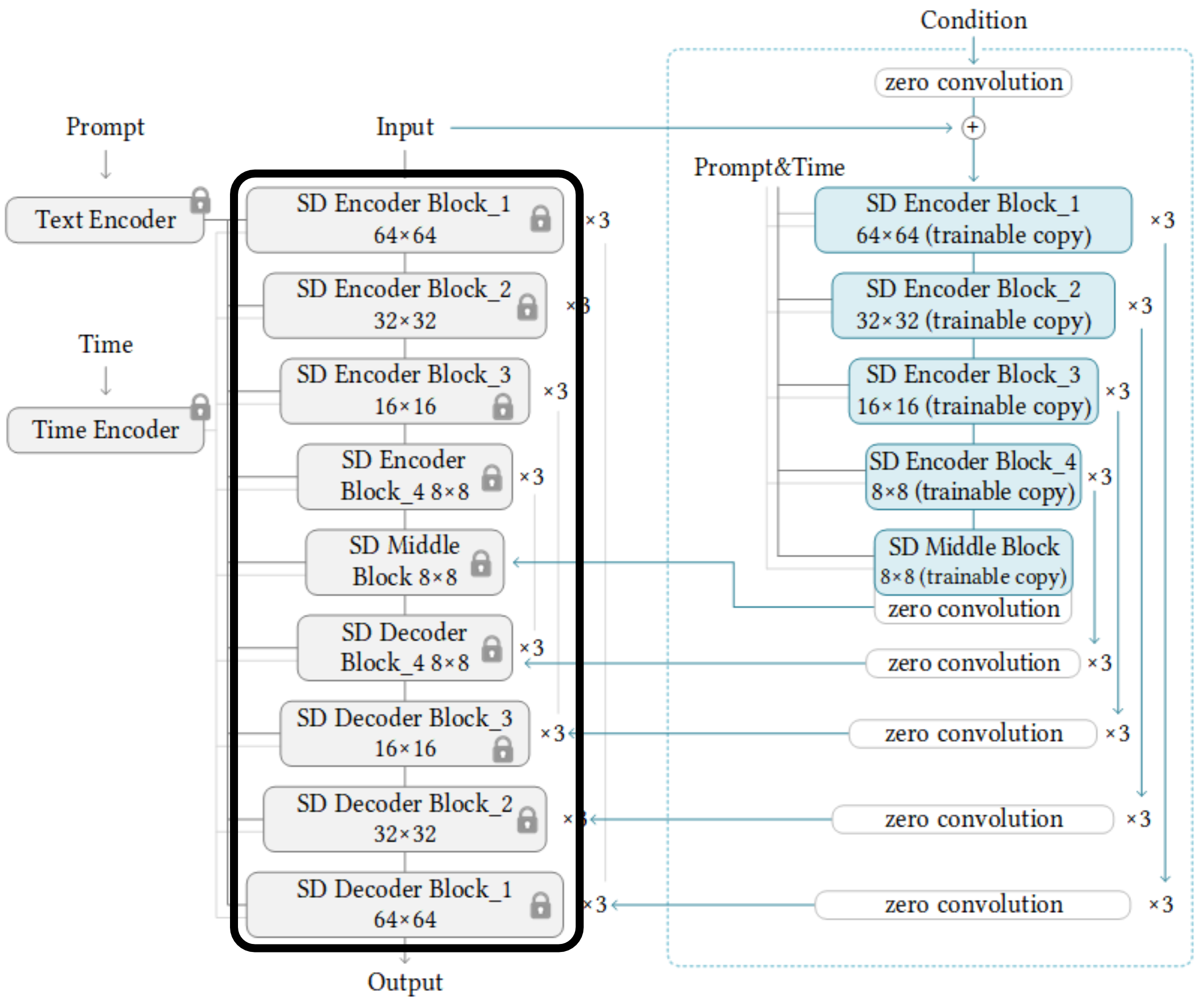


(a) Stable Diffusion (b) ControlNet

Key Idea:

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

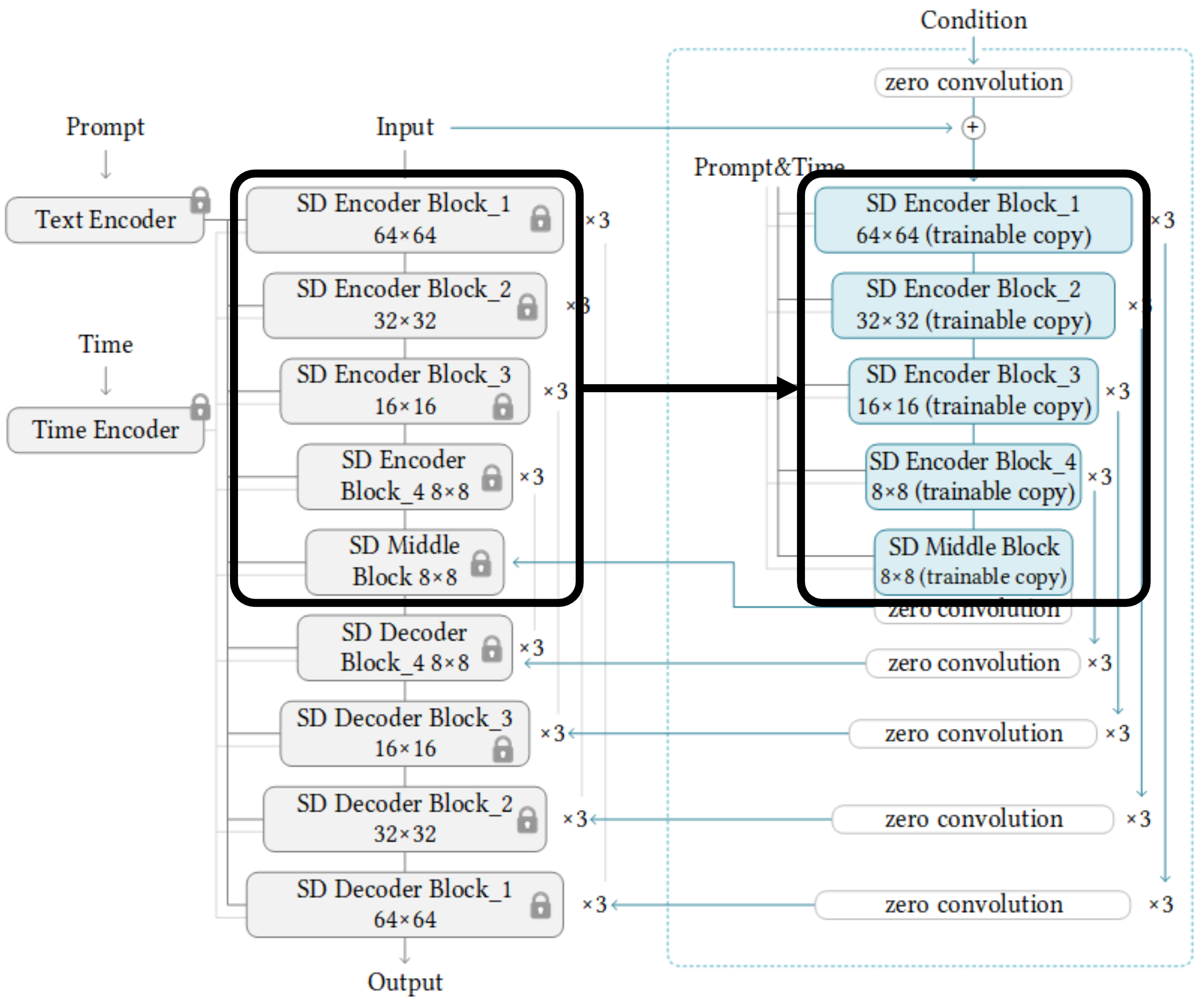
ControlNet [Zhang et al., 2023]



(a) Stable Diffusion (b) ControlNet

1. Freeze the noise prediction network.

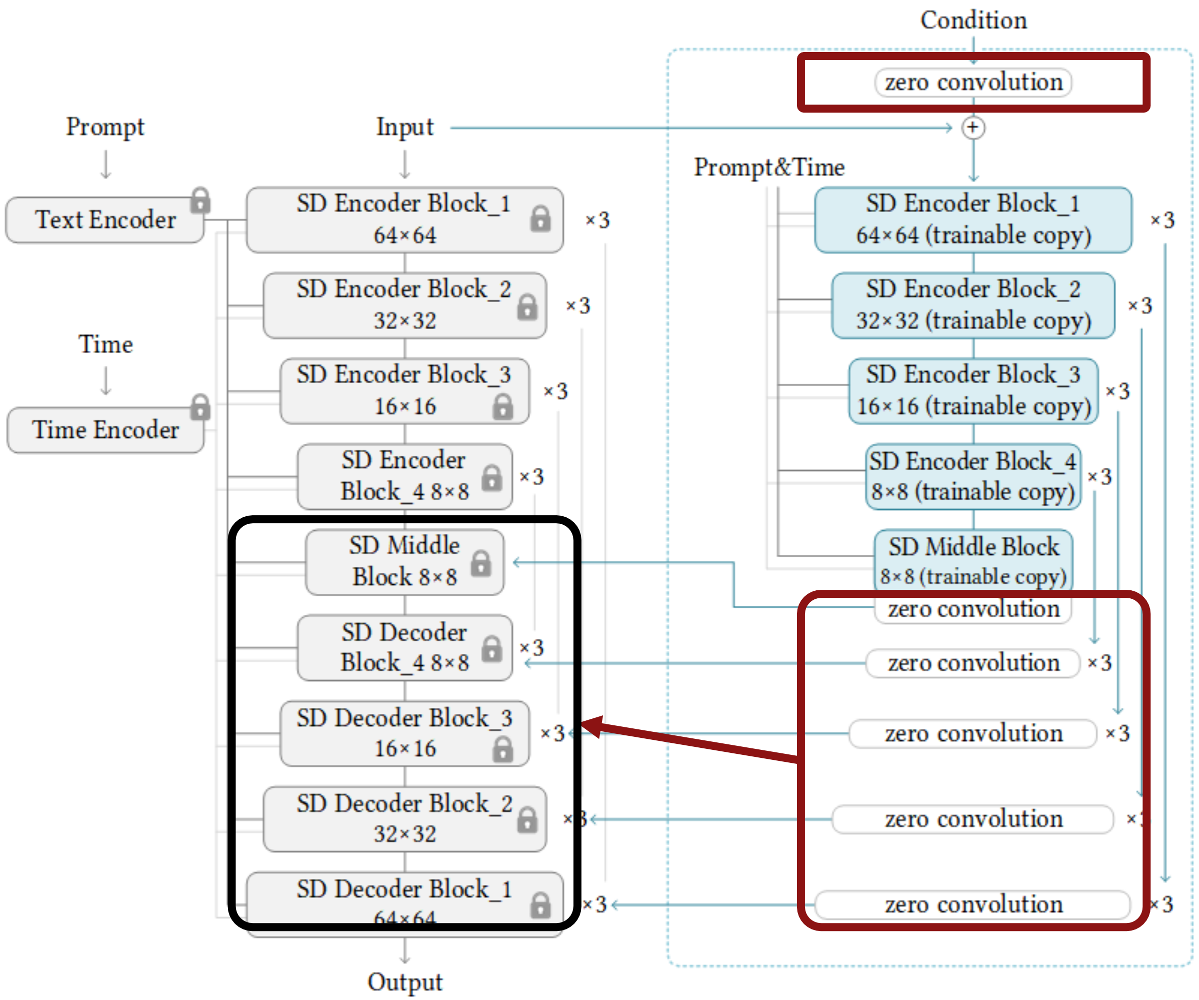
ControlNet [Zhang et al., 2023]



(a) Stable Diffusion (b) ControlNet

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]



(a) Stable Diffusion (b) ControlNet

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:

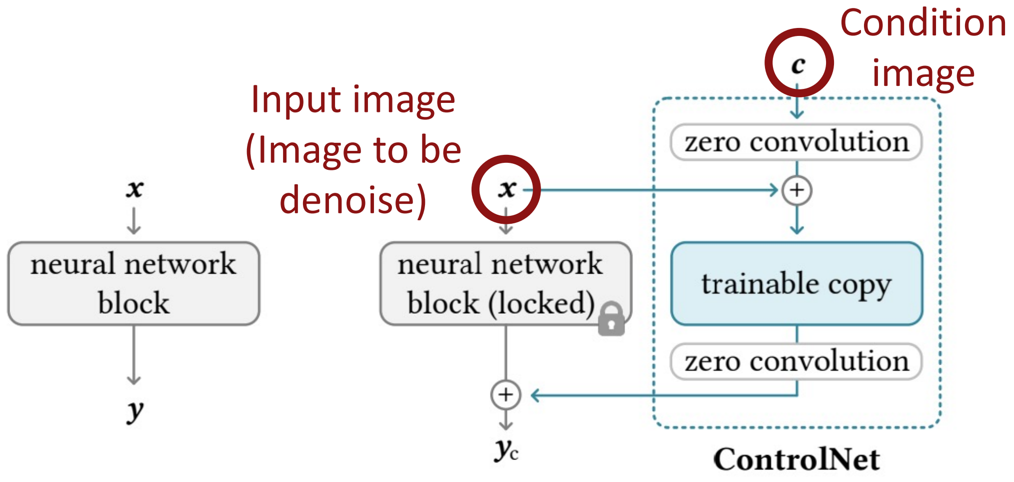
$$\boxed{Z(\boldsymbol{x}; \boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a} \cdot \boldsymbol{x} + \boldsymbol{b}}$$

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:

$$\boldsymbol{y}_c = F(\boldsymbol{x};\Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c};\boldsymbol{a}_1,\boldsymbol{b}_1);\Theta_c);\boldsymbol{a}_2,\boldsymbol{b}_2)$$



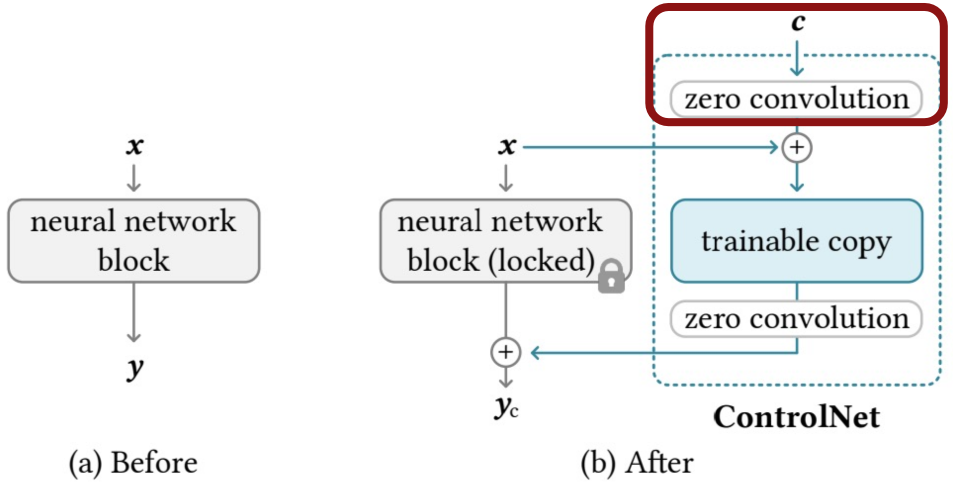
(a) Before (b) After

ControlNet [Zhang et al., 2023]

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

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

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

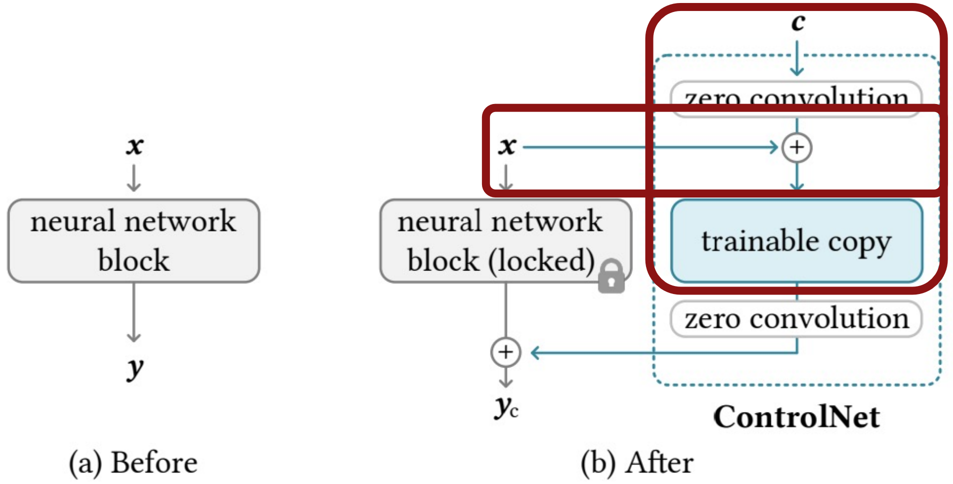


ControlNet [Zhang et al., 2023]

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

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

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

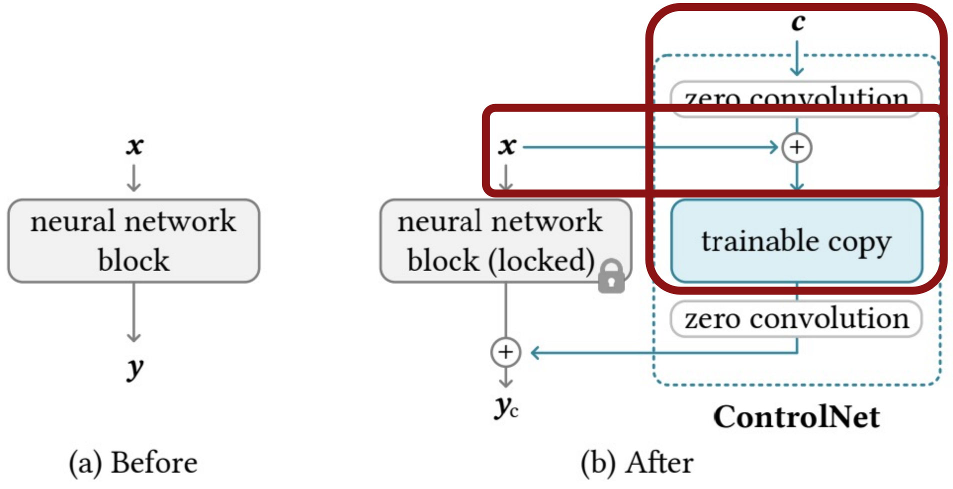


ControlNet [Zhang et al., 2023]

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 $\Theta_c = \Theta$.

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

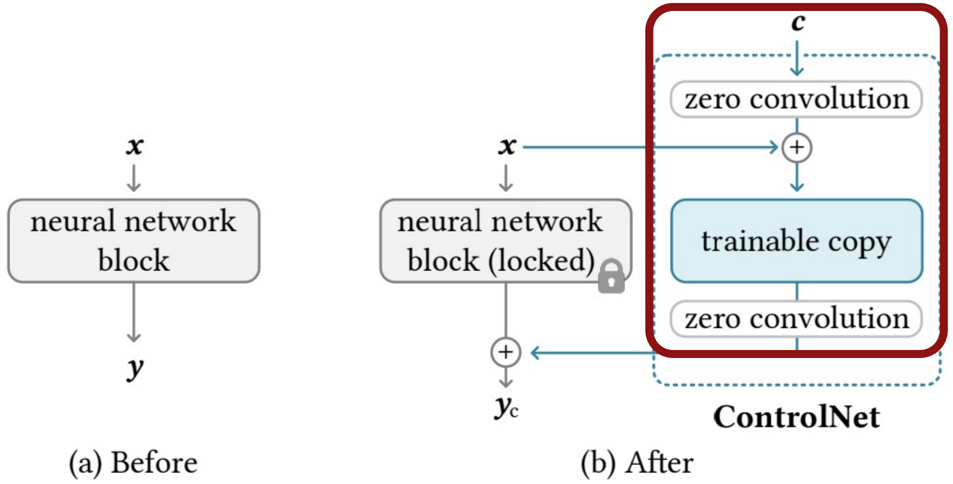


ControlNet [Zhang et al., 2023]

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

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

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

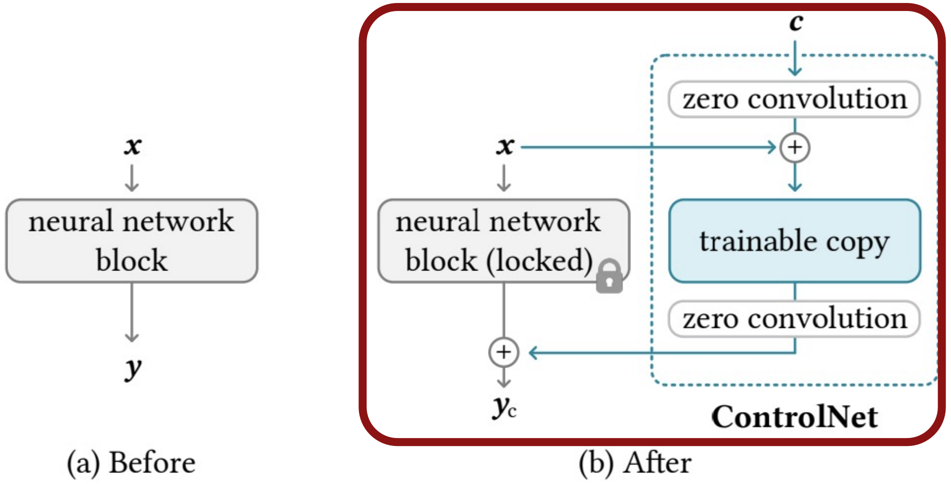


ControlNet [Zhang et al., 2023]

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

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

$$\boldsymbol{y}_c = \boxed{F(\boldsymbol{x};\Theta) + Z(F(\boldsymbol{x} + Z(\boldsymbol{c};\boldsymbol{a}_1,\boldsymbol{b}_1);\Theta_c);\boldsymbol{a}_2,\boldsymbol{b}_2)}$$

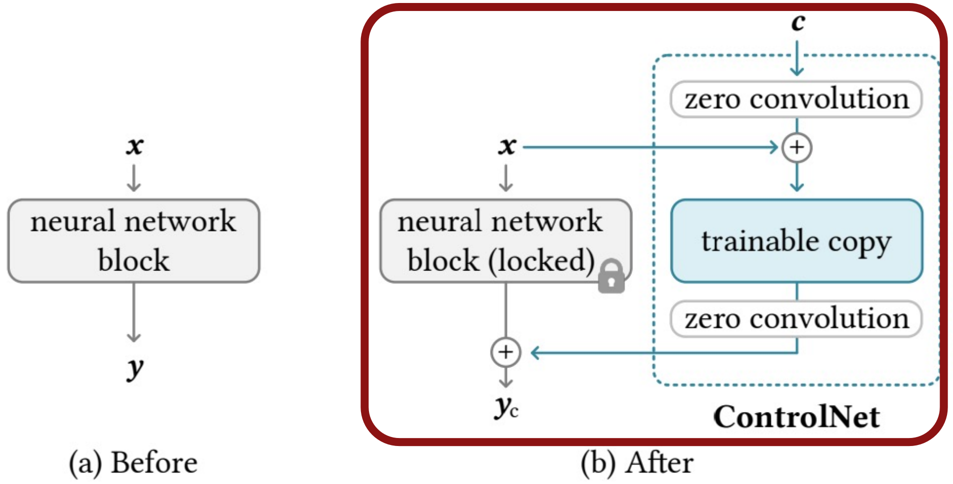


ControlNet [Zhang et al., 2023]

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

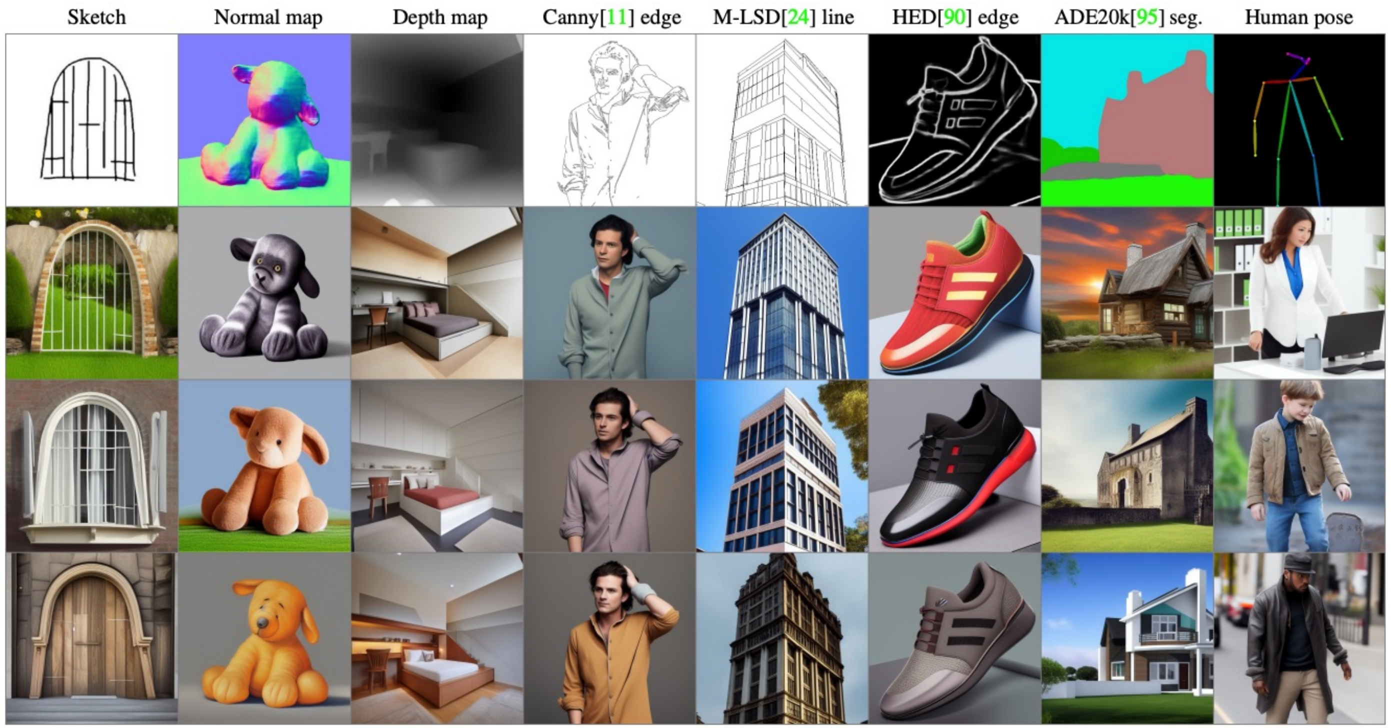
Same as $\mathrm{F}(\mathrm{x}; \Theta)$ in the beginning.

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



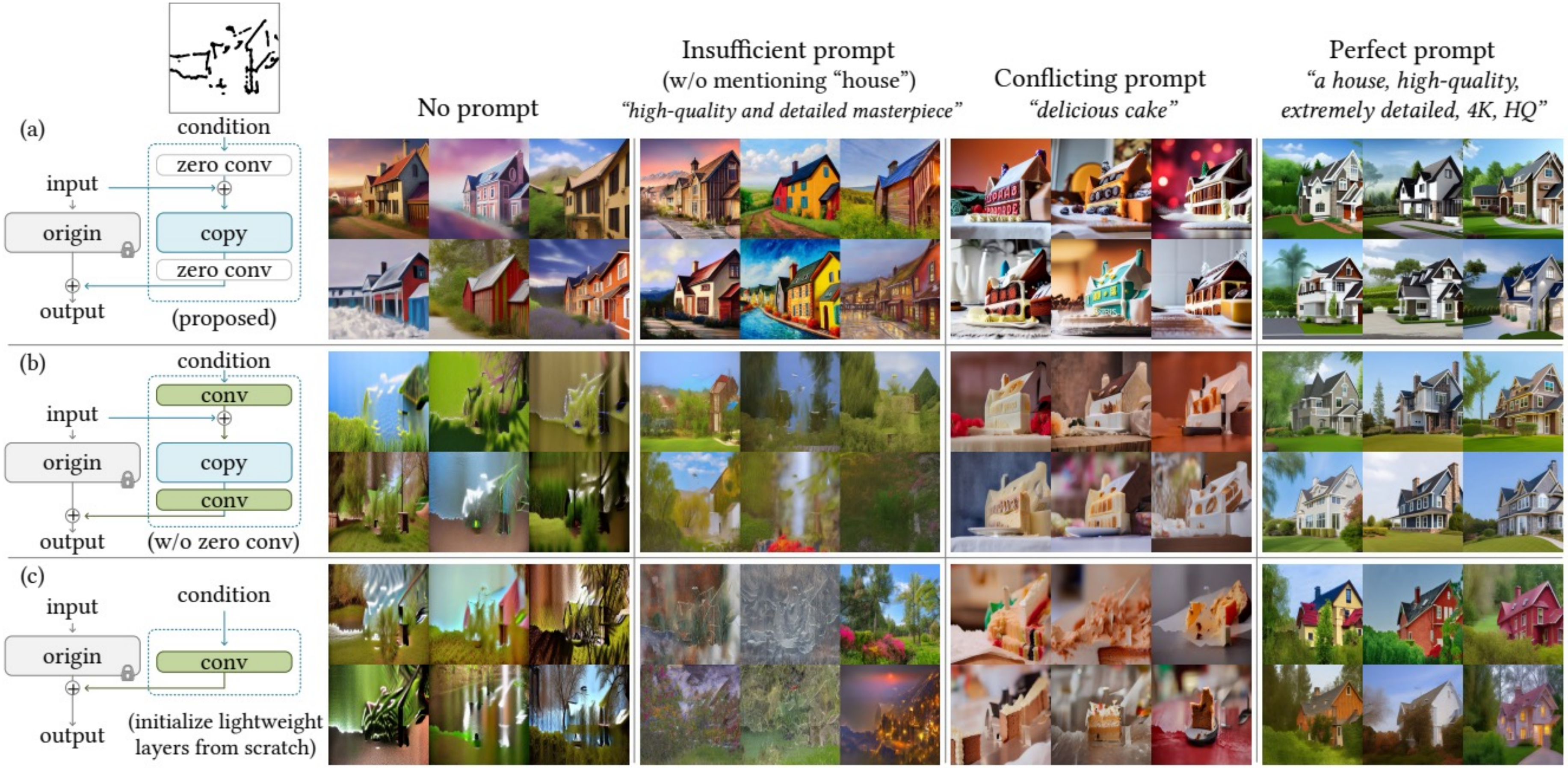
ControlNet [Zhang et al., 2023]

- 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."

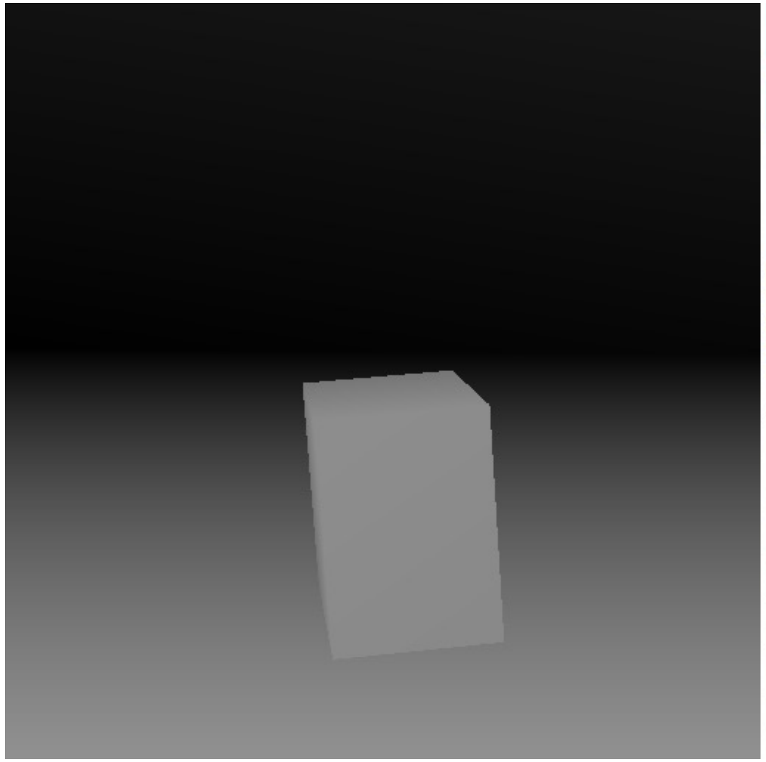


ControlNet [Zhang et al., 2023]

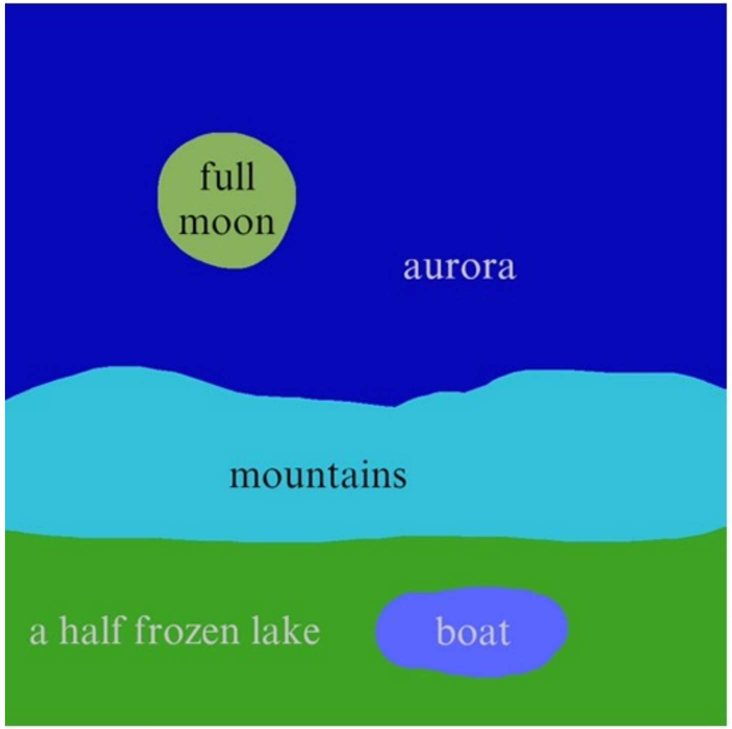
Ablative study of different architectures.

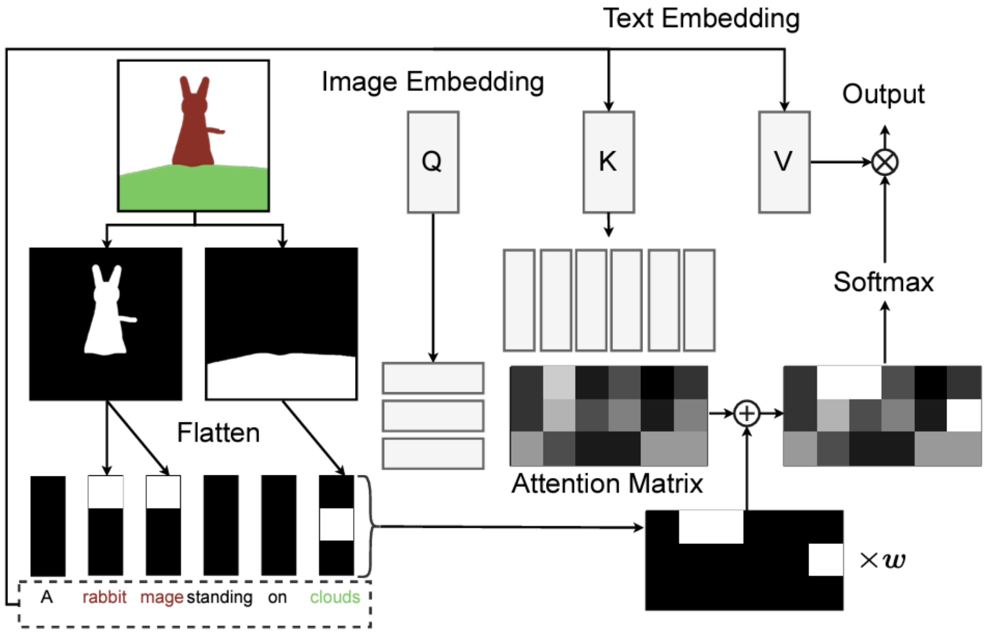


Adherence of Conditioning Signal



Regional Control of Conditioning Signal





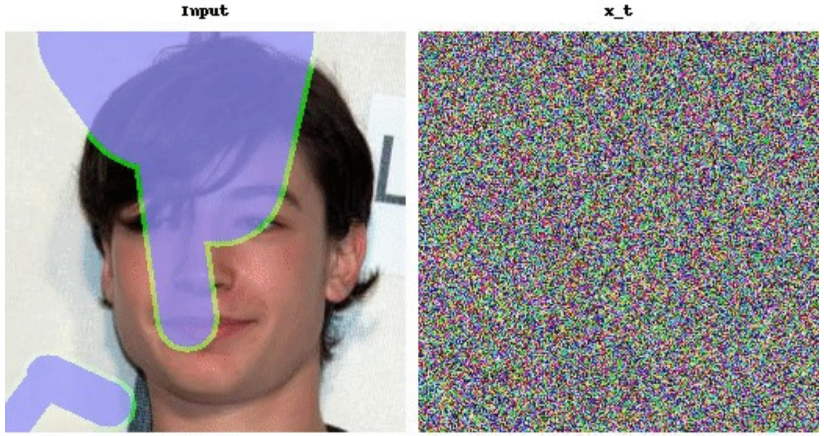
cross attention layer

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.

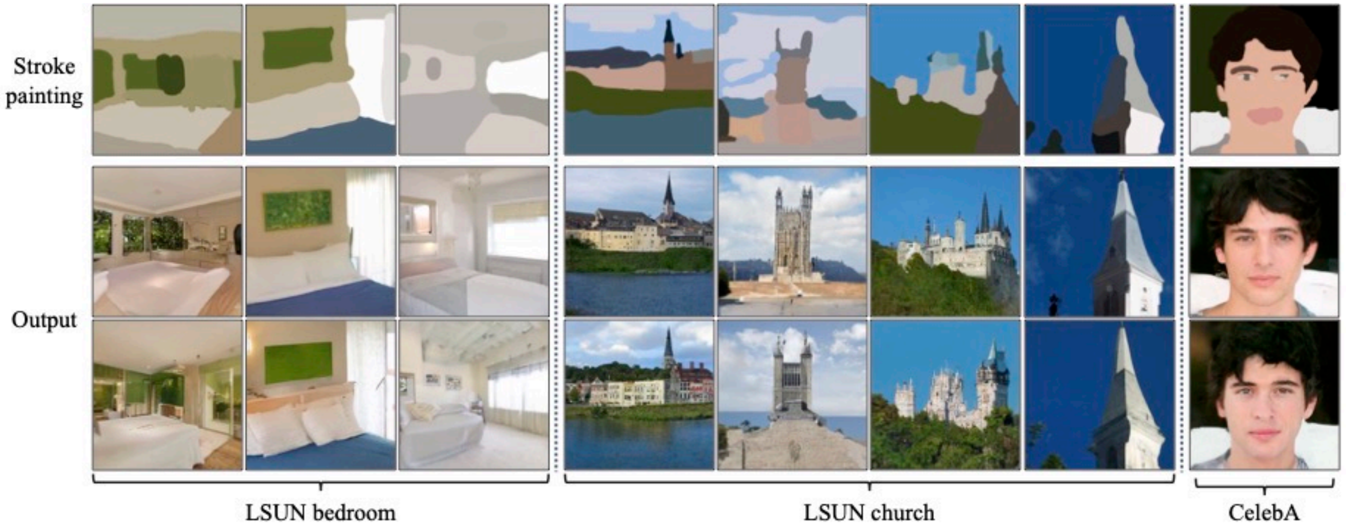


Lugmayr et al., RePaint: Inpainting using Denoising Diffusion Probabilistic Models, CVPR 2022.

SDEdit [Meng et al., 2022]

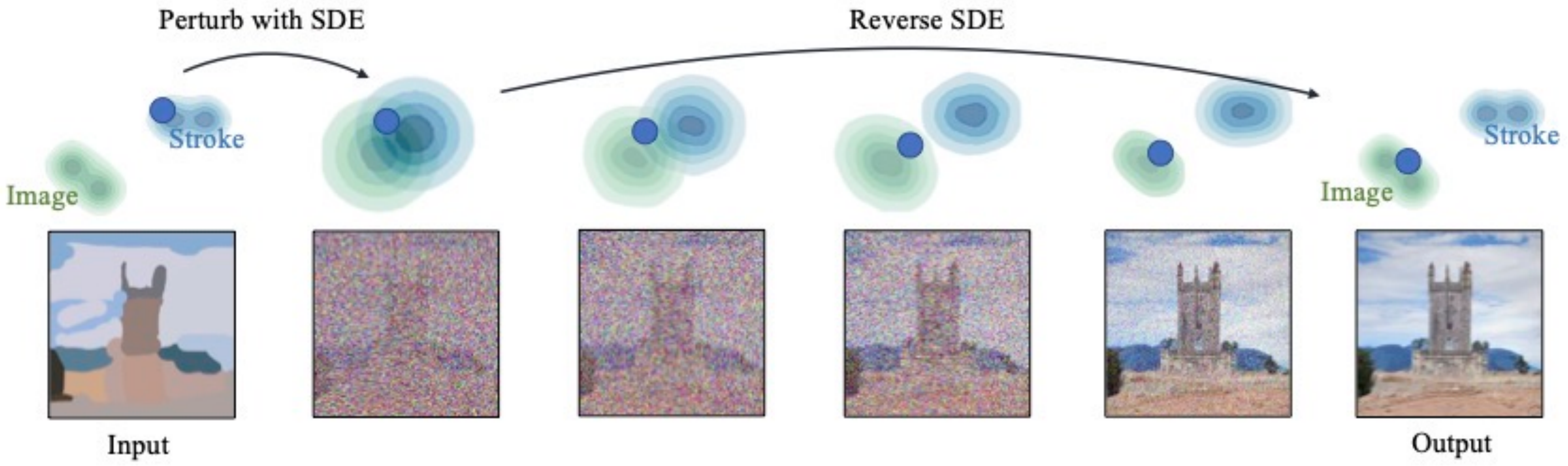
Image generation/editing through user interaction

- Image generation from sketches
- Image editing from scribbles



SDEdit [Meng et al., 2022]

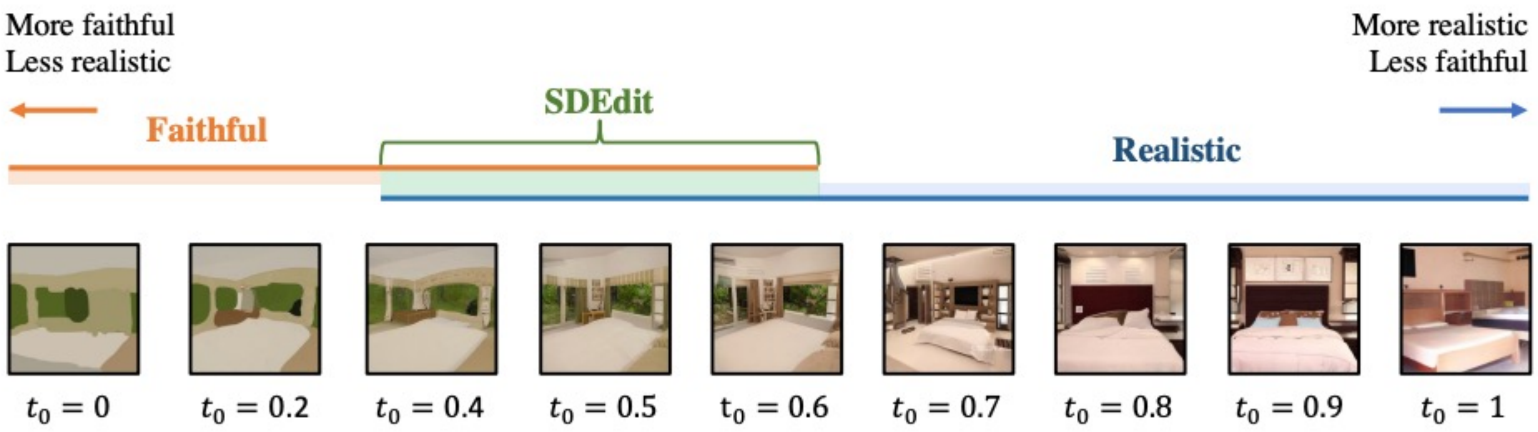
Perform the forward process for a bit and then reverse the process.



SDEdit [Meng et al., 2022]

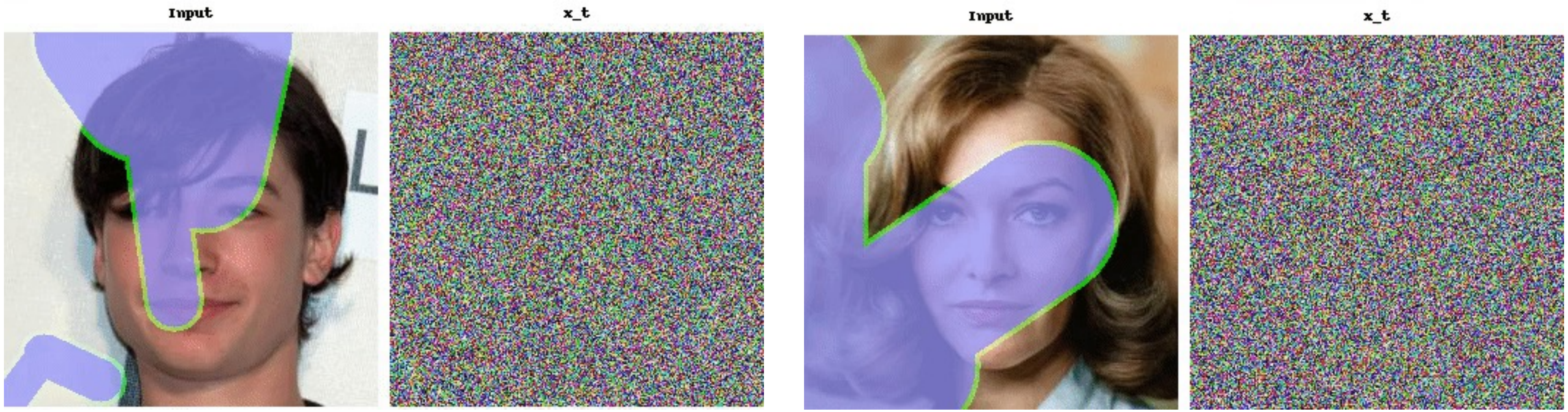
Realism vs. faithfulness

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



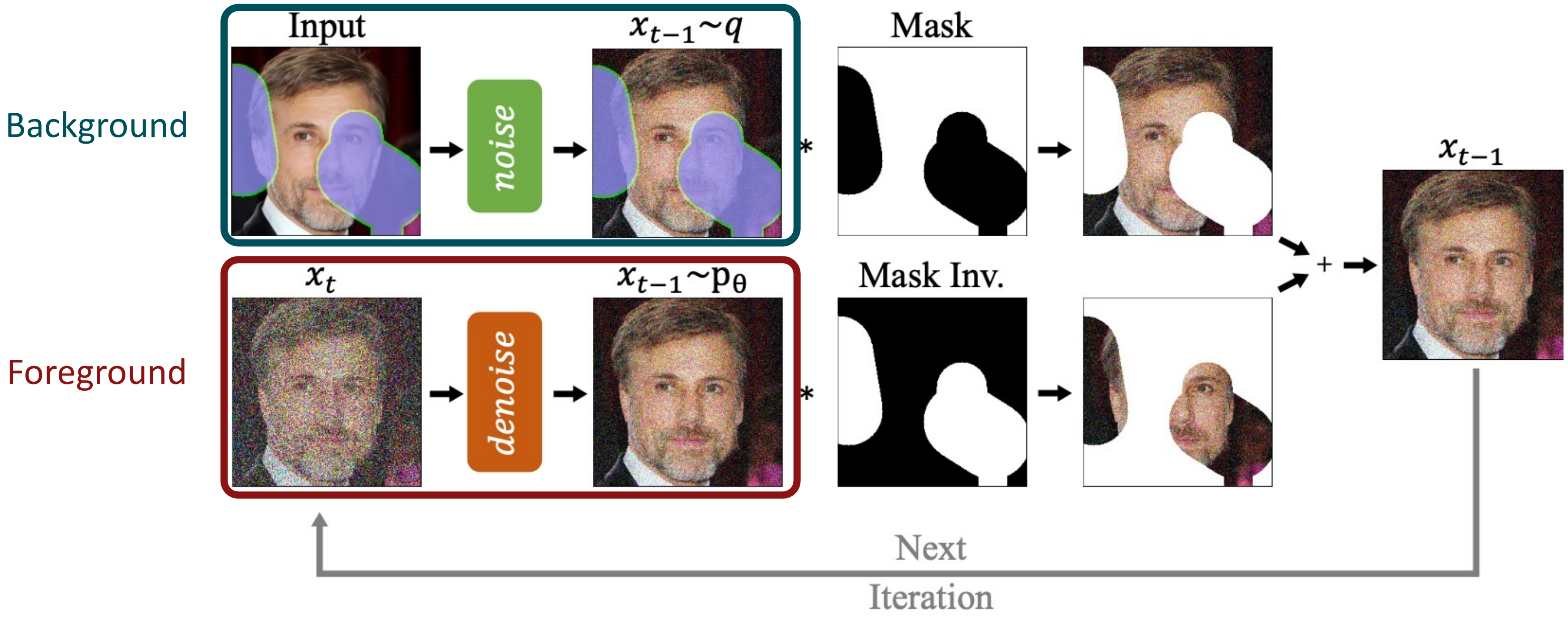
RePaint [Lugmayr et al., 2022]

- 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]

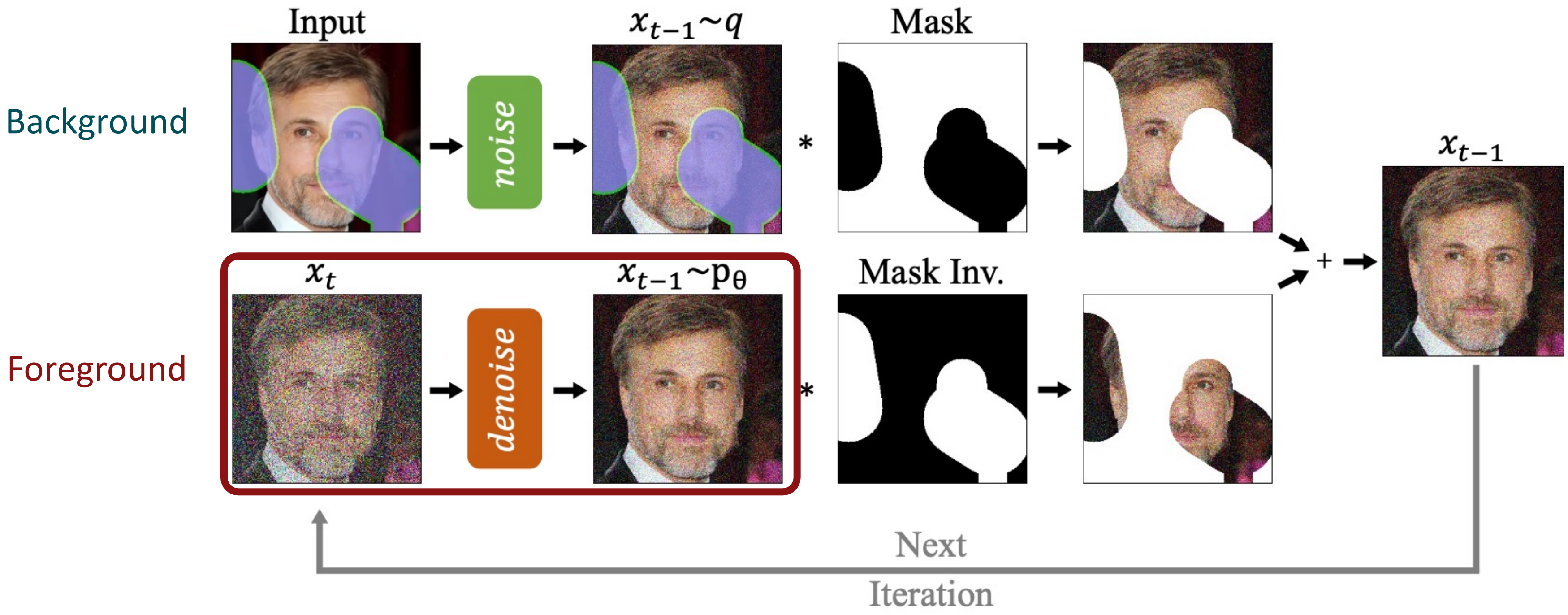
Combine denoised foreground images (to be filled) and noisy background (to be fixed) images at each iteration of the reverse process.



RePaint [Lugmayr et al., 2022]

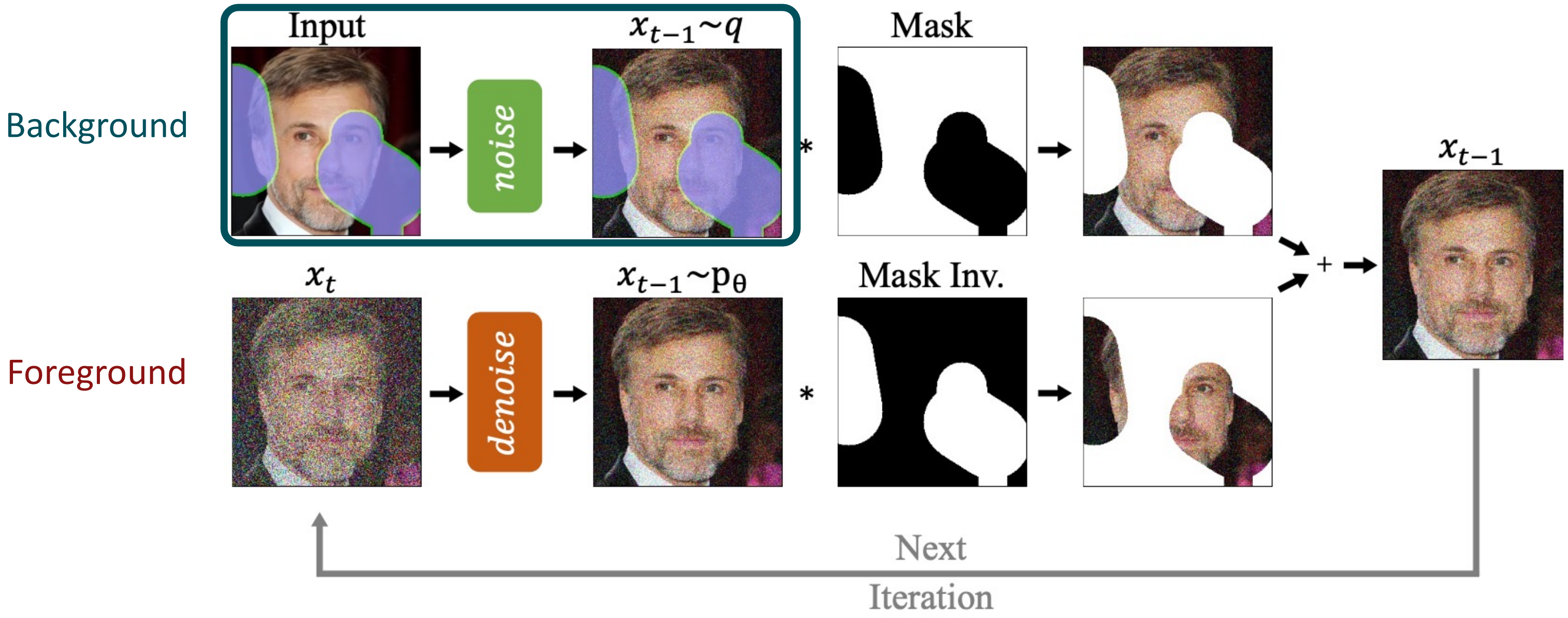
1. Starting from $\boldsymbol{x}_T$, at each timestep t,

   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 $\boldsymbol{x}_{t-1}^b$.



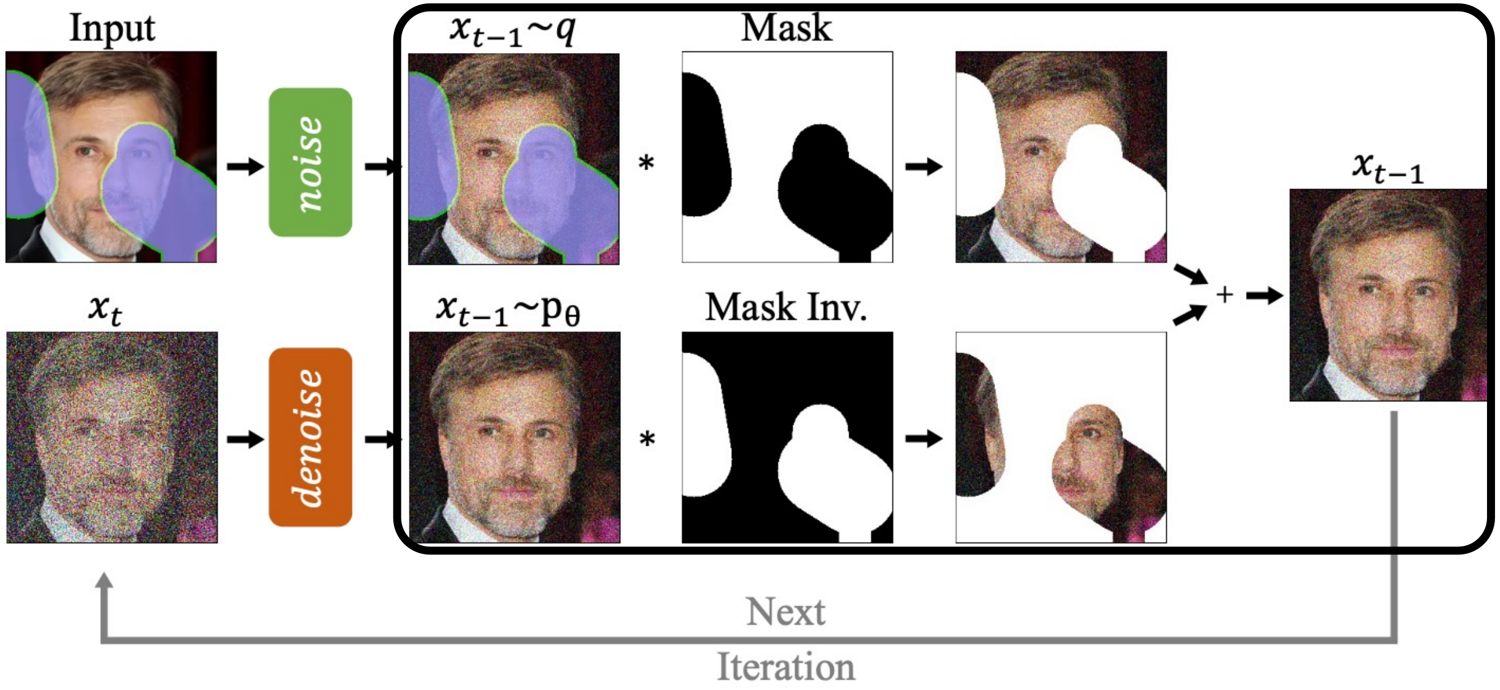
RePaint [Lugmayr et al., 2022]

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

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

Background

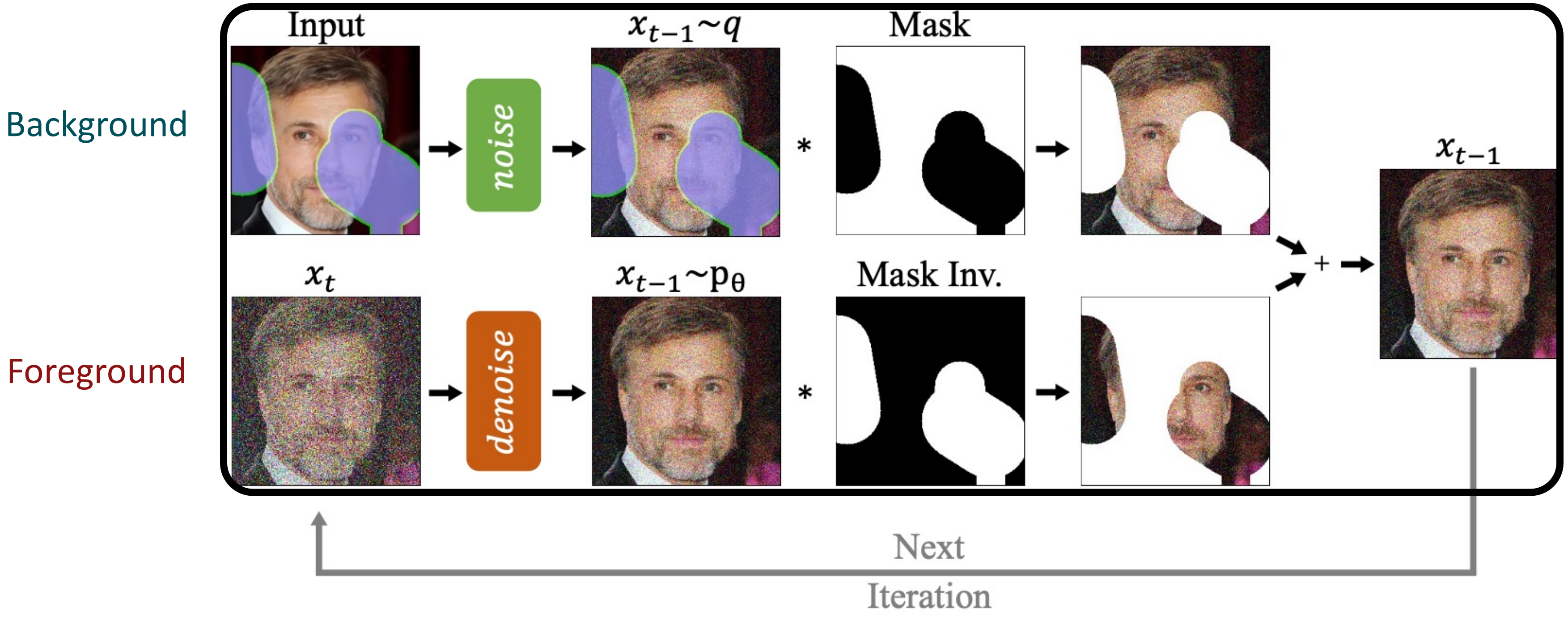
Foreground



RePaint [Lugmayr et al., 2022]

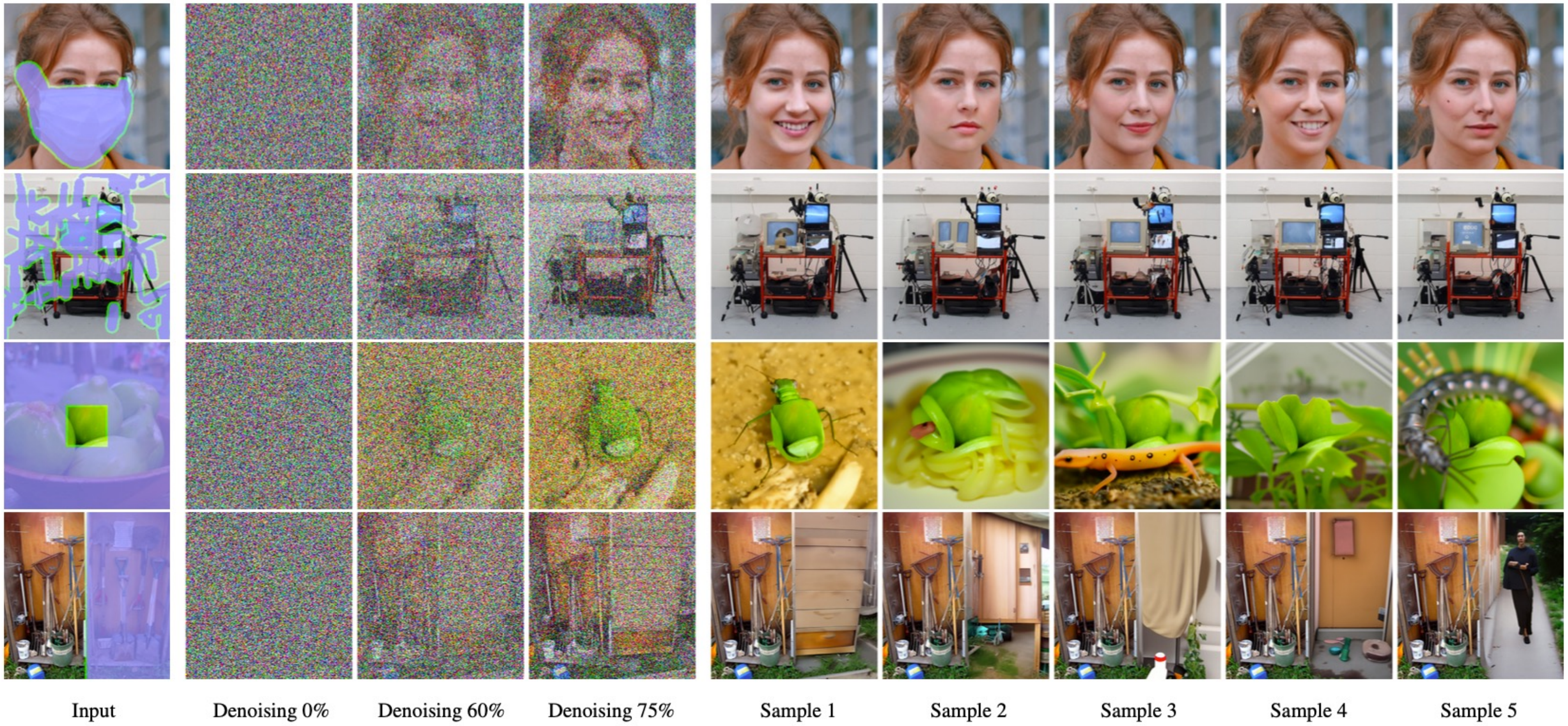
4. Repeat this process for $\mathrm{t} = T, \ldots, 1$.

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



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.