Adding Conditional Control to Text-to-Image Diffusion Models

Presented By- Rittik Panda(MT2022090)

Adding Conditional Control to Text-to-Image Diffusion Models

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala
Stanford University
{lvmin, anyirao, maneesh}@cs.stanford.edu

Figure 1: Controlling Stable Diffusion with learned conditions. ControlNet allows users to add conditions like Canny edges (top), human pose (bottom), *etc*., to control the image generation of large pretrained diffusion models. The default results use the prompt “a high-quality, detailed, and professional image”. Users can optionally give prompts like the “chef in kitchen”.

Original Paper: https://openaccess.thecvf.com/content/ICCV2023/papers/Zhang_Adding_Conditional_Control_to_Text-to-Image_Diffusion_Models_ICCV_2023_paper.pdf

Key Contributions

- **ControlNet Introduction**: Integrates spatial controls into large text-to-image diffusion models using robust pretrained encoding layers.

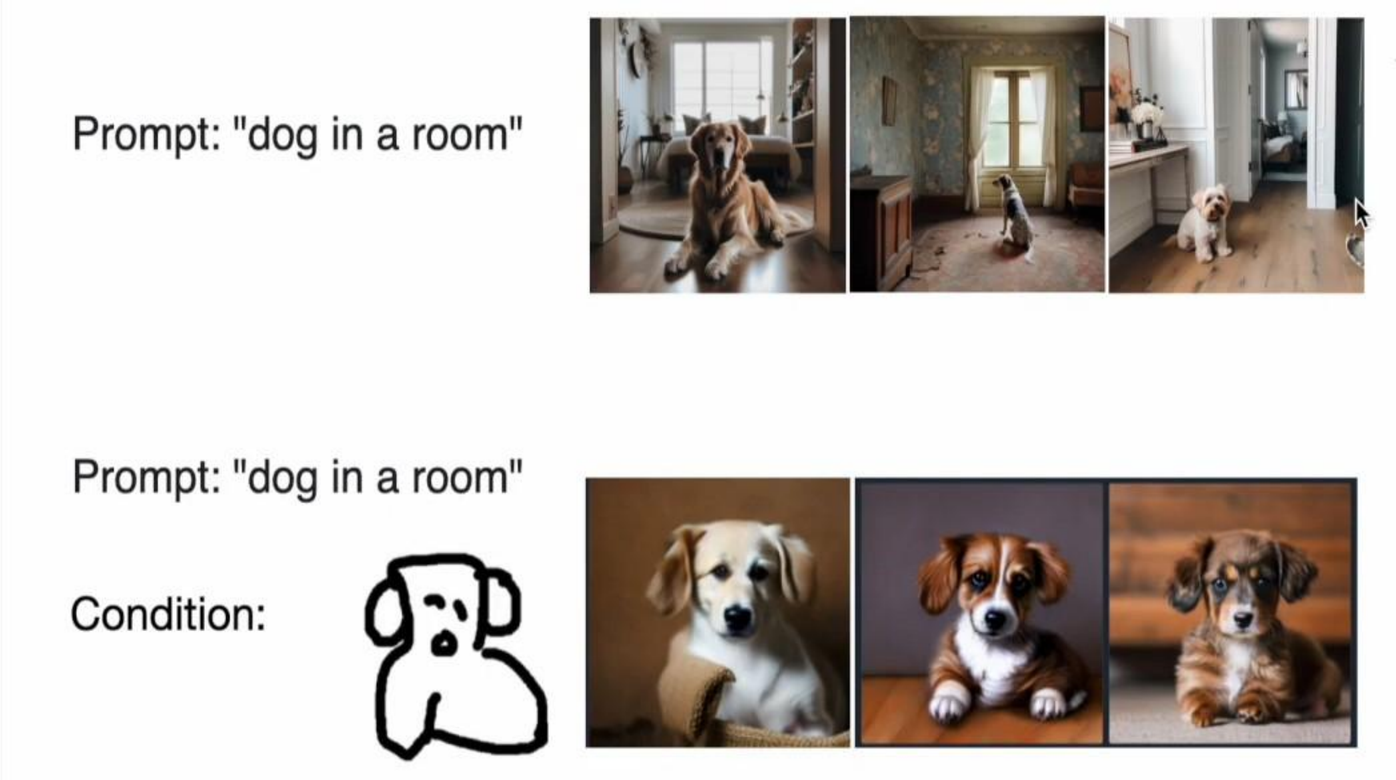
- **"Zero Convolutions" Implementation**: Initiates convolution layers from zero, progressively expanding parameters to prevent noise during fine-tuning.

- **Exploration of Conditioning Controls**: Experimentation with edges, depth, segmentation, and human poses, both singularly and combined with Stable Diffusion, with and without prompts.

- **Robust Training Demonstrations**: Exhibits robustness in training with small (<50k) and large (>1m) datasets.

- **Enhanced Model Potential**: Suggests ControlNet's capability to significantly enhance image diffusion models by enabling effective control mechanisms.

Example



Abstract

- To control pretrained large diffusion models to support additional input conditions.
- End-to-end architecture
- Robust on small dataset(<50k)
- As fast as fine-tuning
- Can be trained on personal devices
- Can scale to large amounts of data(millions to billions)

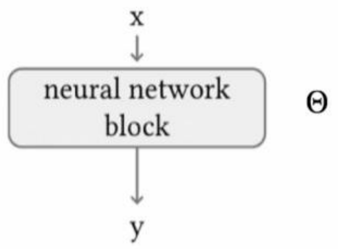
Introduction

- Can large models be applied to facilitate specific tasks?
- What kind of framework should we build to handle the wide range of problem conditions and user controls?
- Three findings:
  - The available data scale in a task-specific domain is not always as large as that in the general image-text domain
  - Large computation clusters are not always available
  - Various image processing problems have diverse forms of problem definitions,user controls or image annotations

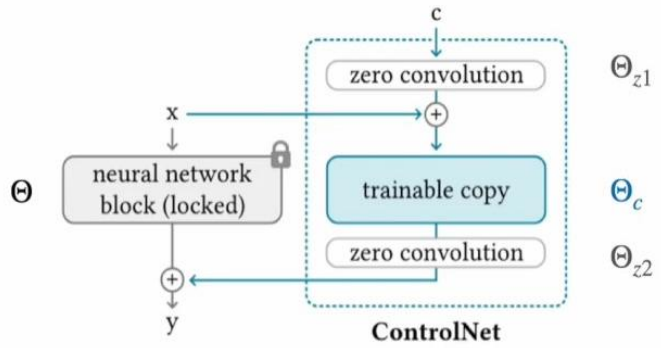
Method

ControlNet

- Manipulated the input conditions of neural network blocks to further control the overall behavior of an entire neural network
- $y = F(x, \Theta)$ => $y = F(x; \Theta) + Z(F(x + Z(c; \Theta_{z1}); \Theta_c); \theta_{z2})$
    - x: input feature map
    - y: output feature map
    - c: external condition vector
    - Θ_c: Trainable copy
    - Z: Zero convolution, 1 x 1 convolution layer with both weights and bias initialized with zeros



(a) Before



(b) After

$$y = F(x;\Theta) + Z(F(x + Z(c;\Theta_{z1});\Theta_c);\theta_{z2})$$

Method

ControlNet

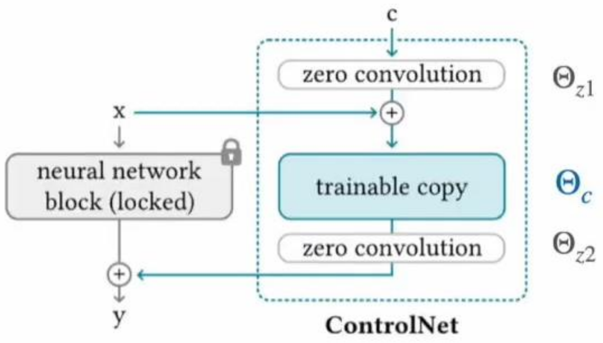
- Before any optimization, it will not cause any influence to the deep neural features

Because both the weight and bias of a zero convolution layer are initialized as zeros, in the first training step, we have

$$\begin{cases} \mathcal{Z}(\boldsymbol{c};\Theta_{z1}) = \boldsymbol{0} \\ \mathcal{F}(\boldsymbol{x} + \mathcal{Z}(\boldsymbol{c};\Theta_{z1});\Theta_c) = \mathcal{F}(\boldsymbol{x};\Theta_c) = \mathcal{F}(\boldsymbol{x};\Theta) \\ \mathcal{Z}(\mathcal{F}(\boldsymbol{x} + \mathcal{Z}(\boldsymbol{c};\Theta_{z1});\Theta_c);\Theta_{z2}) = \mathcal{Z}(\mathcal{F}(\boldsymbol{x};\Theta_c);\Theta_{z2}) = \boldsymbol{0} \end{cases} \tag{3}$$

and this can be converted to

$$\boldsymbol{y}_c = \boldsymbol{y} \tag{4}$$

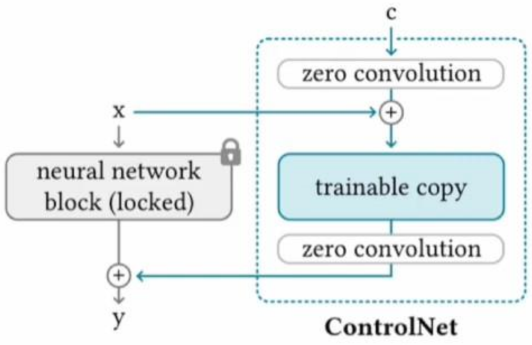


(b) After

Method

ControlNet

- Zero convolutions progressively grow from zeros to optimized parameters in a learned way



(b) After

We briefly deduce the gradient calculation of a zero convolution layer. Considering an 1×1 convolution layer with weight $\boldsymbol{W}$ and bias $\boldsymbol{B}$, at any spatial position p and channel-wise index i, given an input map $\boldsymbol{I} \in \mathbb{R}^{h \times w \times c}$, the forward pass can be written as

$$\mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i} = \boldsymbol{B}_i + \sum_j^c \boldsymbol{I}_{p,i} \boldsymbol{W}_{i,j} \tag{5}$$

and since zero convolution has $\boldsymbol{W} = \boldsymbol{0}$ and $\boldsymbol{B} = \boldsymbol{0}$ (before optimization), for anywhere with $\boldsymbol{I}_{p,i}$ being non-zero, the gradients become

$$\begin{cases} \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{B}_i} = 1 \\ \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{I}_{p,i}} = \sum_j^c \boldsymbol{W}_{i,j} = 0 \\ \dfrac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{W}_{i,j}} = \boldsymbol{I}_{p,i} \neq \boldsymbol{0} \end{cases} \tag{6}$$

and we can see that although a zero convolution can cause the gradient on the feature term $\boldsymbol{I}$ to become zero, the weight's and bias's gradients are not influenced. As long as the feature $\boldsymbol{I}$ is non-zero, the weight $\boldsymbol{W}$ will be optimized into non-zero matrix in the first gradient descent iteration. Notably, in our case, the feature term is input data or condition vectors sampled from datasets, which naturally ensures non-zero $\boldsymbol{I}$. For example, considering a classic gradient descent with an overall loss function $\mathcal{L}$ and a learning rate $\beta_{\text{lr}} \neq 0$, if the "outside" gradient $\partial \mathcal{L} / \partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})$ is not zero, we will have

$$\boldsymbol{W}^* = \boldsymbol{W} - \beta_{\text{lr}} \cdot \frac{\partial \mathcal{L}}{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})} \odot \frac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})}{\partial \boldsymbol{W}} \neq \boldsymbol{0} \tag{7}$$

where $\boldsymbol{W}^*$ is the weight after one gradient descent step; $\odot$ is Hadamard product. After this step, we will have

$$\frac{\partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}^*, \boldsymbol{B}\})_{p,i}}{\partial \boldsymbol{I}_{p,i}} = \sum_j^c \boldsymbol{W}^*_{i,j} \neq \boldsymbol{0} \tag{8}$$

Method

ControlNet in Image Diffusion Model

- Computationally efficient
- Overall learning objective:
  $\mathcal{L} = E_{z_0, t, c_t, c_f, \epsilon \sim \mathcal{N}(0,1)}[\epsilon - \epsilon_\theta(z_t, t, c_t, c_f)\|_2^2]$
  - z_0: original images
  - z_t: noisy images
  - t: time step
  - c_t: text prompts
  - c_f: task-specific conditions

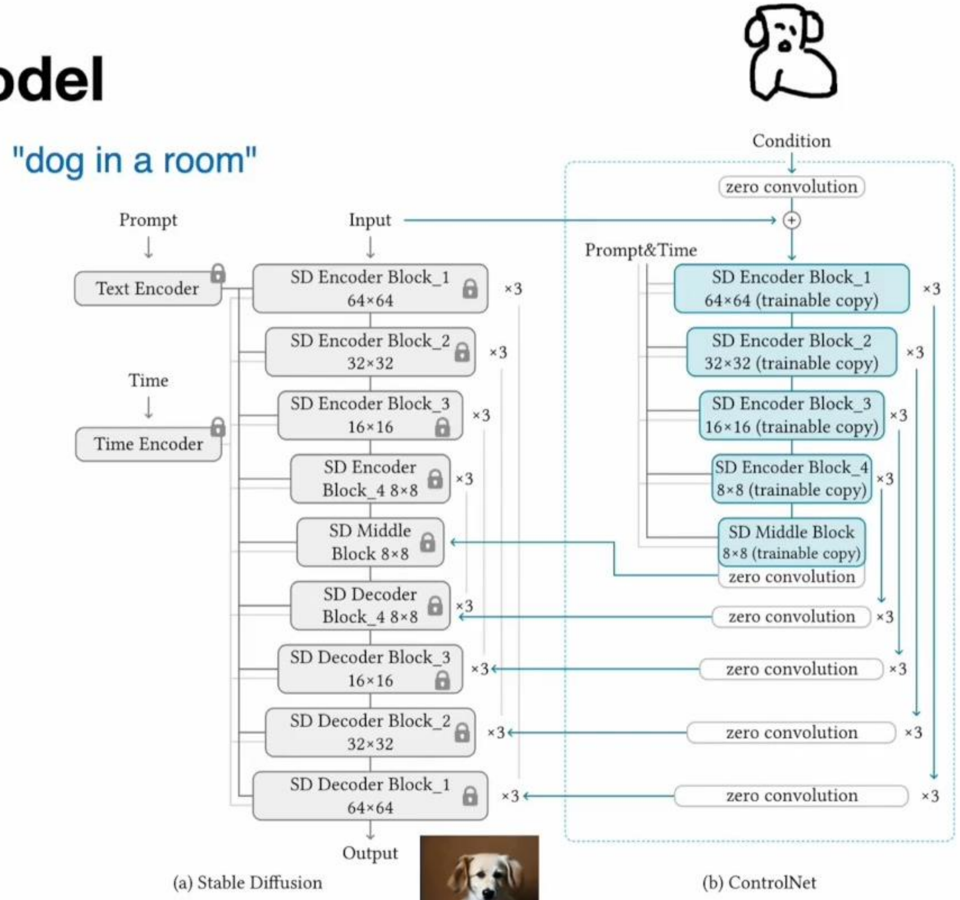


Figure 3: ControlNet in Stable Diffusi blocks are the structure of Stable Diffusion 1.5 (or SD V2.1, since they use the same U-Net architecture), while the blue blocks are ControlNet.

Method

Improved Training

- Small-Scale Training
  - Disconnecting the link to decoder 1,2,3,4 and only connecting the middle block
- Large-Scale Training
  - First train ControlNets for a large enough number of iterations
  - Then unlock all weights of the Stable Diffusion and jointly train the entire model as a whole

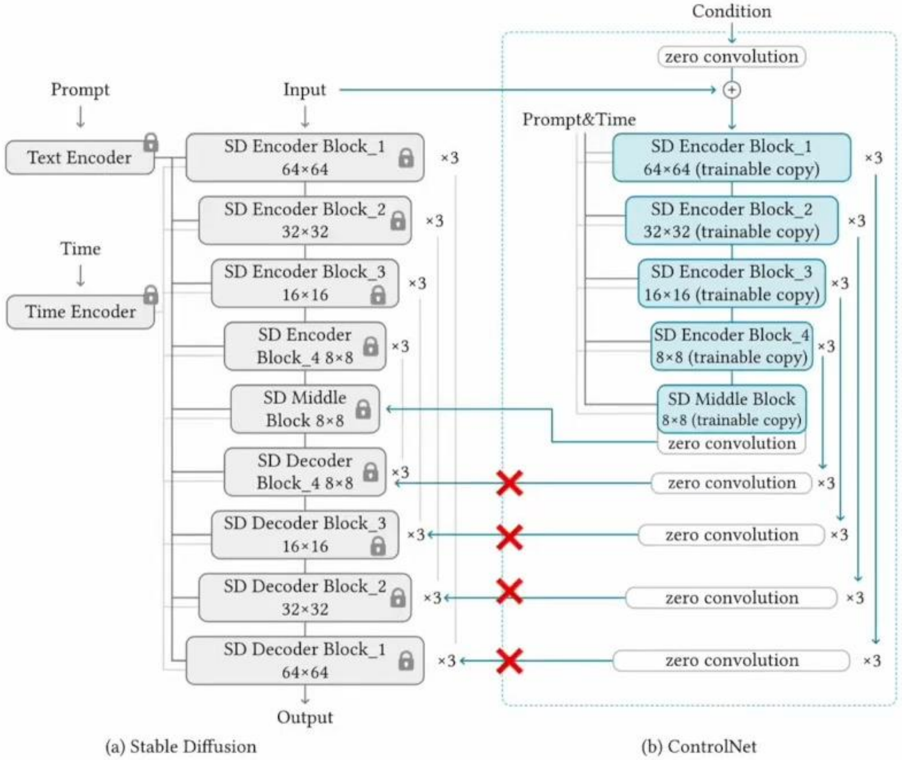


Figure 3: ControlNet in Stable Diffusion. The gray blocks are the structure of Stable Diffusion 1.5 (or SD V2.1, since they use the same U-Net architecture), while the blue blocks are ControlNet.

Experiment

- Large diffusion models like Stable Diffusion can be augmented with ControlNets to enable conditional input like:
    - Edge maps
    - Human scribbles
    - Segmentation maps
    - Keypoints
    - ...

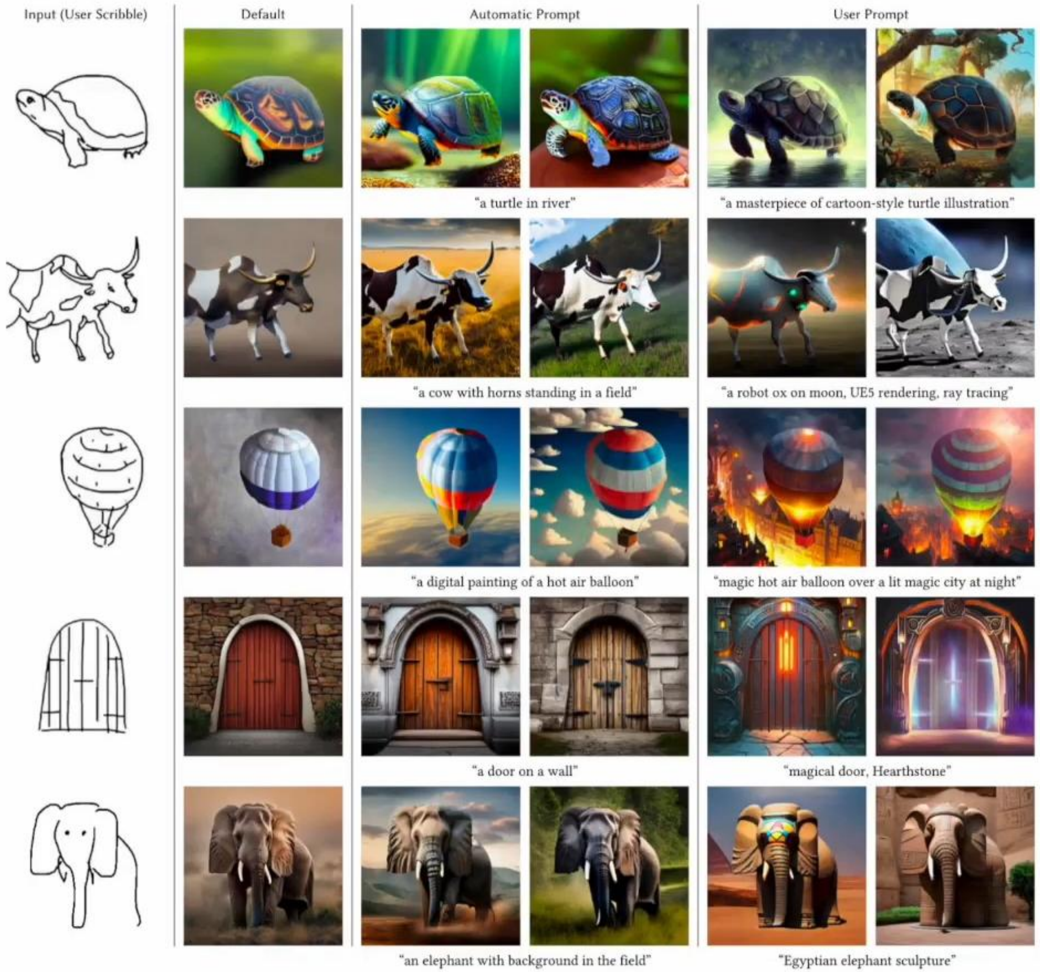


Figure 6: Controlling Stable Diffusion with Human scribbles. The "automatic prompts" are generated by BLIP based on the default result images without using user prompts. These scribbles are from [58].

Experiment

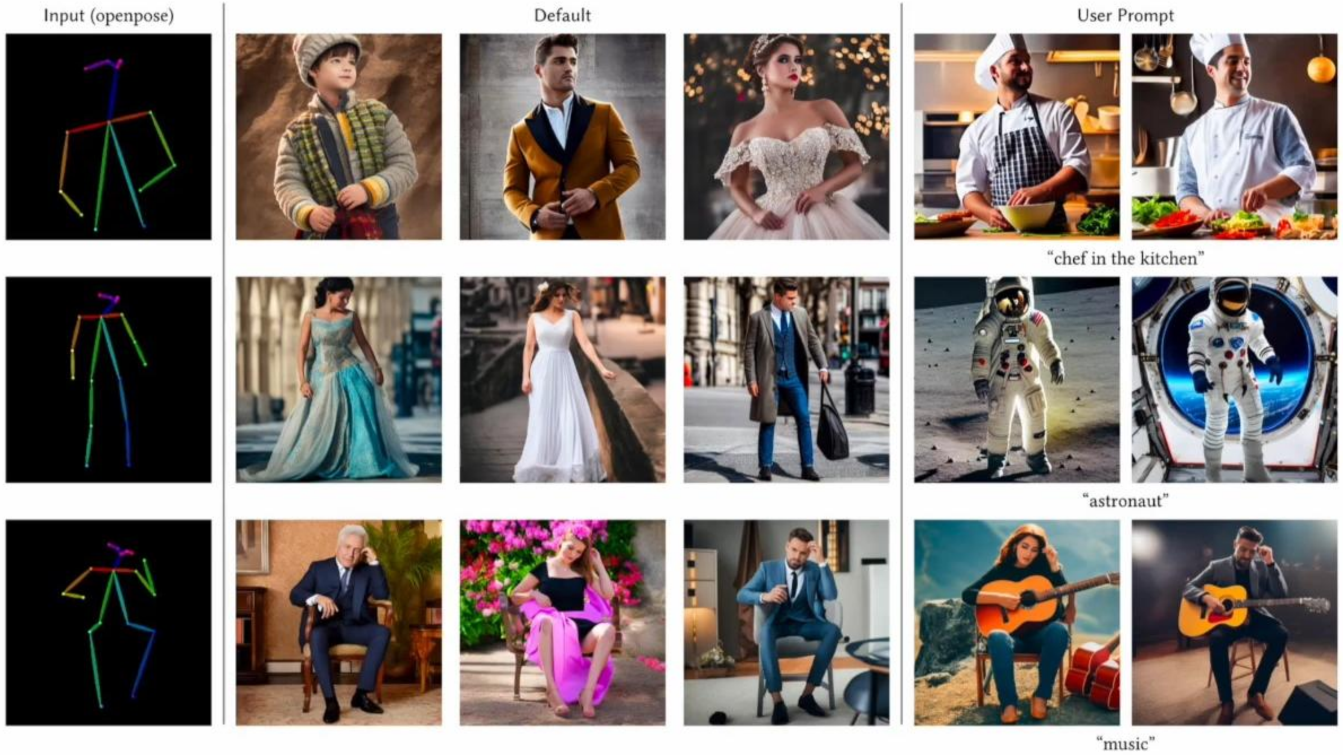


Figure 9: Controlling Stable Diffusion with Openpose. See also the Appendix for source images for Openpose pose detection.

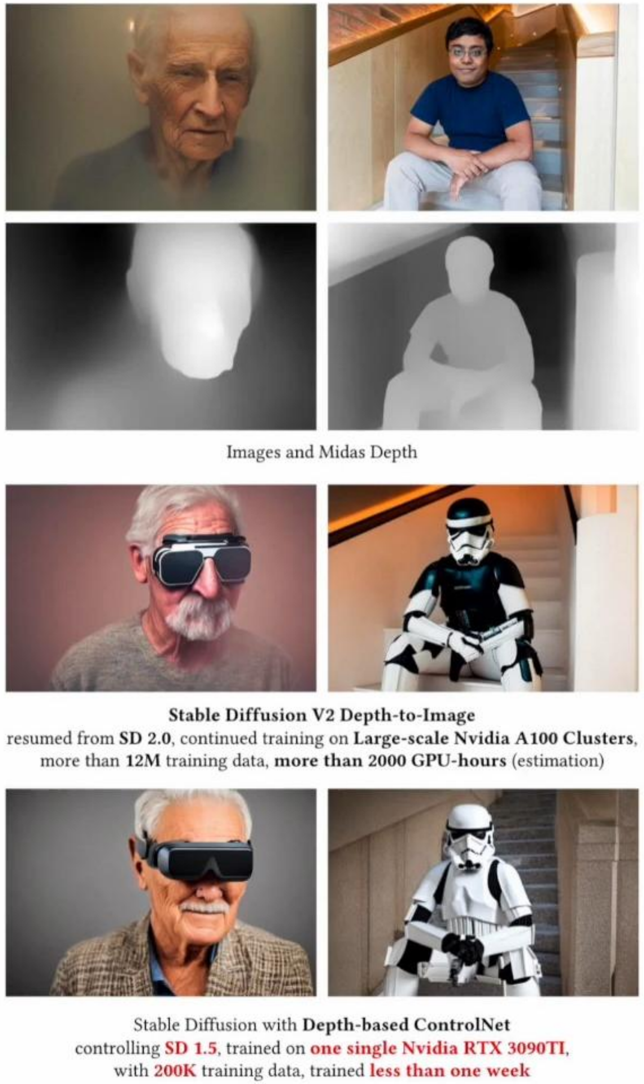


Figure 14: Comparison of Depth-based ControlNet and Stable Diffusion V2 Depth-to-Image. Note that in this experiment, the Depth-based ControlNet is trained at a relatively small scale to test minimal required computation resources. We also provide relatively stronger models that are trained at relatively large scale.

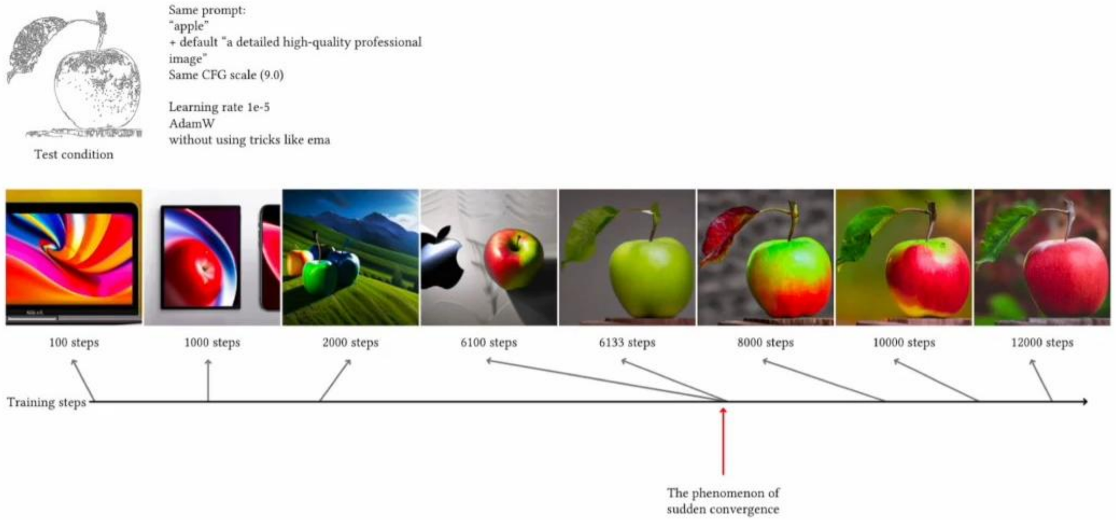


Figure 21: The sudden converge phenomenon. Because we use zero convolutions, the neural network always predict high-quality images during the entire training. At a certain point of training step, the model suddenly learns to adapt to the input conditions. We call this "sudden converge phenomenon".

Figure 22: Training on different scale. We show the Canny-edge-based ControlNet trained on different experimental settings with various dataset size.

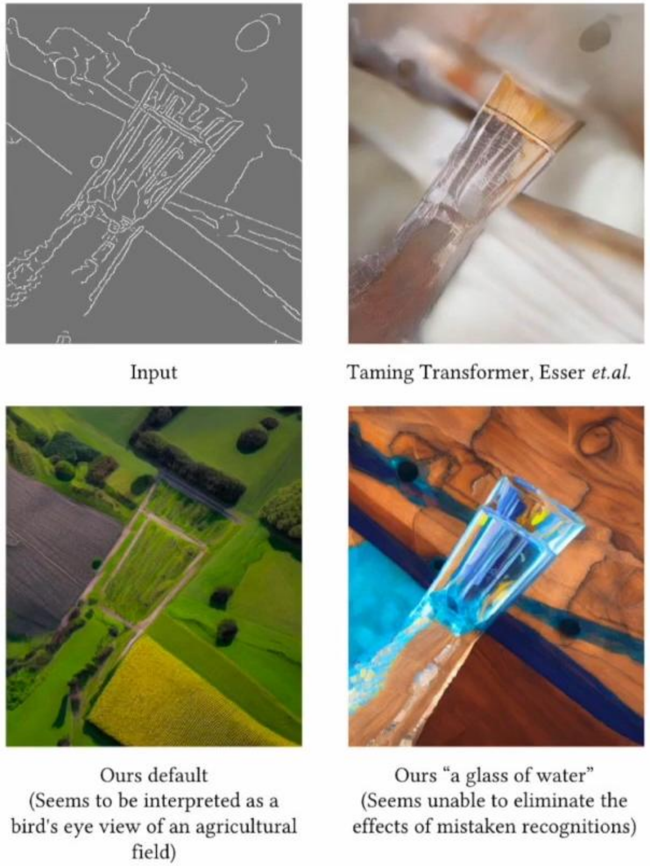


Figure 28: Limitation. When the semantic of input image is mistakenly recognized, the negative effects seem difficult to be eliminated, even if a strong prompt is provided.

Results & Comparison With Prev. Methods

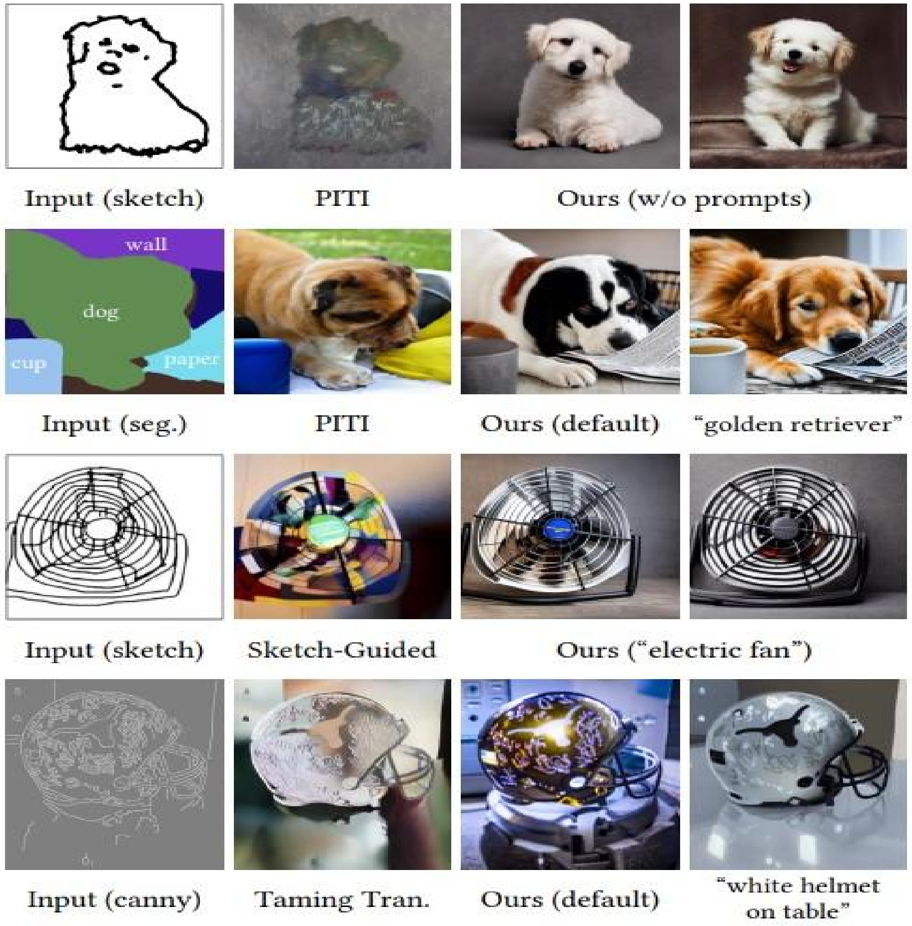


Figure 9: Comparison to previous methods.We present the qualitative comparisons to PITI [88], Sketch-Guided Diffusion [87], and Taming Transformers [19].

Method	FID ↓	CLIP-score ↑	CLIP-aes. ↑
Stable Diffusion	6.09	0.26	6.32
VQGAN [19](seg.)*	26.28	0.17	5.14
LDM [71](seg.)*	25.35	0.18	5.15
PIPT [88](seg.)	19.74	0.20	5.77
ControlNet-lite	17.92	0.26	6.30
ControlNet	15.27	0.26	6.31

Table 3: Evaluation for image generation conditioned by semantic segmentation. We report FID, CLIP text-image score, and CLIP aesthetic scores for our method and other baselines. We also report the performance of Stable Diffusion without segmentation conditions. Methods marked with "*" are trained from scratch.

Ablative Study

Figure 8: Ablative study of different architectures on a sketch condition and different prompt settings. For each setting, we show a random batch of 6 samples without cherry-picking. Images are at 512 $\times$ 512 and best viewed when zoomed in. The green "conv" blocks on the left are standard convolution layers initialized with Gaussian weights.

THANK YOU