Adding Conditional Control to Text-to-Image Diffusion Models

Lvmin Zhang and Maneesh Agrawala
Stanford University

We present a neural network structure, ControlNet, to control pretrained large diffusion models to support additional input conditions. The ControlNet learns task-specific conditions in an end-to-end way, and the learning is robust even when the training dataset is small (< 50k). Moreover, training a ControlNet is as fast as fine-tuning a diffusion model, and the model can be trained on a personal devices. Alternatively, if powerful computation clusters are available, the model can scale to large amounts (millions to billions) of data. We report that large diffusion models like Stable Diffusion can be augmented with ControlNets to enable conditional inputs like edge maps, segmentation maps, keypoints, *etc*. This may enrich the methods to control large diffusion models and further facilitate related applications.

https://github.com/lllyasviel/ControlNet

1 Introduction

With the presence of large text-to-image models, generating a visually appealing image may require only a short descriptive prompt entered by users. After typing some texts and getting the images, we may naturally come up with several questions: does this prompt-based control satisfy our needs? For example in image processing, considering many long-standing tasks with clear problem formulations, can these large models be applied to facilitate these specific tasks? What kind of framework should we build to handle the wide range of problem conditions and user controls? In specific tasks, can large models preserve the advantages and capabilities obtained from billions of images?

To answer these questions, we investigate various image processing applications and have three findings. First, the available data scale in a task-specific domain is not always as large as that in the general image-text domain. The largest dataset size of many specific problems (*e.g*., object shape/normal, pose understanding, *etc*.) is often under 100k, *i.e*., 5×10^4 times smaller than LAION-5B. This would require robust neural network training method to avoid overfitting and to preserve generalization ability when the large models are trained for specific problems.

Second, when image processing tasks are handled with data-driven solutions, large computation clusters are not always available. This makes fast training methods important for optimizing large models to specific tasks within an acceptable amount of time and memory space (*e.g*., on personal devices). This would further require the utilization of pretrained weights, as well as fine-tuning strategies or transfer learning.

Third, various image processing problems have diverse forms of problem definitions, user controls, or image annotations. When addressing these problems, although an image diffusion algorithm can be regulated in a "procedural" way, *e.g*., constraining denoising process, editing multi-head attention activations, *etc*., the behaviors of these hand-crafted rules are fundamentally prescribed by human directives. Considering some specific tasks like depth-to-image, pose-to-human, *etc*., these problems essentially require the interpretation of raw inputs into object-level or scene-level understandings, making hand-crafted procedural methods less feasible. To achieve learned solutions in many tasks, the end-to-end learning is indispensable.

Figure 1: Control Stable Diffusion with Canny edge map. The canny edge map is input, and the source image is not used when we generate the images on the right. The outputs are achieved with a default prompt *"a high-quality, detailed, and professional image"*. This prompt is used in this paper as a default prompt that does not mention anything about the image contents and object names. Most of figures in this paper are high-resolution images and best viewed when zoomed in.

This paper presents ControlNet, an end-to-end neural network architecture that controls large image diffusion models (like Stable Diffusion) to learn task-specific input conditions. The ControlNet clones the weights of a large diffusion model into a "trainable copy" and a "locked copy": the locked copy preserves the network capability learned from billions of images, while the trainable copy is trained on task-specific datasets to learn the conditional control. The trainable and locked neural network blocks are connected with an unique type of convolution layer called "zero convolution", where the convolution weights progressively grow from zeros to optimized parameters in a learned manner. Since the production-ready weights are preserved, the training is robust at datasets of different scale. Since the zero convolution does not add new noise to deep features, the training is as fast as fine tuning a diffusion model, compared to training new layers from scratch.

We train several ControlNets with various datasets of different conditions, *e.g.*, Canny edges, Hough lines, user scribbles, human key points, segmentation maps, shape normals, depths, *etc*. We also experiment ControlNets with both small datasets (with samples less than 50k or even 1k) and large datasets (millions of samples). We also show that in some tasks like depth-to-image, training ControlNets on a personal computer (one Nvidia RTX 3090TI) can achieve competitive results to commercial models trained on large computation clusters with terabytes of GPU memory and thousands of GPU hours.

2 Related Work

2.1 HyperNetwork and Neural Network Structure

HyperNetwork originates from a neural language processing method [14] to train a small recurrent neural network to influence the weights of a larger one. Successful results of HyperNetwork are also reported in image generation using generative adversarial networks [1, 10] and other machine learning tasks [51]. Inspired by these ideas, [15] provided a method to attach a smaller neural network to Stable Diffusion [44] so as to change the artistic style of its output images. This approach gained more popularity after [28] provided the pretrained weights of several HyperNetworks. ControlNet and HyperNetwork have similarities in the way they influence the behaviors of neural networks.

ControlNet uses a special type of convolution layer called "zero convolution". Early neural network studies [31, 47, 32] have extensively discussed the initialization of network weights, including the rationality of initializing the weights with Gaussian distributions and the risks that may incur by initializing the weights with zeros. More recently, [37] discussed a method to scale the initial weight of several convolution layers in a diffusion model to improve the training, which shares similarity with the idea of zero convolution (and their codes contain a function called "zero_module"). Manipulating the initial convolution weights is also discussed in ProGAN [21] and StyleGAN [22], as well as Noise2Noise [33] and [65]. Stability's model cards [55] also mention the use of zero weights in neural layers.

2.2 Diffusion Probabilistic Model

Diffusion probabilistic model was proposed in [52]. Successful results of image generation are first reported at small scale [25] and then relatively large scale [9]. This architecture was improved by important training and sampling methods like Denoising Diffusion Probabilistic Model (DDPM) [17], Denoising Diffusion Implicit Model (DDIM) [53], and score-based diffusion [54]. Image diffusion methods can directly use pixel colors as training data, and in that case, researches often consider strategies to save computation powers when handling high-resolution images [53, 50, 26], or directly use pyramid-based or multiple-stage methods [18, 43]. These methods essentially use U-net [45] as their neural network architecture. In order to reduce the computation power required for training a diffusion model, based on the idea of latent image [11], the approach Latent Diffusion Model (LDM) [44] was proposed and further extended to Stable Diffusion.

2.3 Text-to-Image Diffusion

Diffusion models can be applied to text-to-image generating tasks to achieve state-of-the-art image generating results. This is often achieved by encoding text inputs into latent vectors using pretrained language models like CLIP [41]. For instances, Glide [38] is a text-guided diffusion models supporting both image generating and editing. Disco Diffusion is a clip-guided implementation of [9] to process text prompts. Stable Diffusion is a large scale implementation of latent diffusion [44] to achieve text-to-image generation. Imagen [49] is a text-to-image structure that does not use latent images and directly diffuse pixels using a pyramid structure.

2.4 Personalization,Customization, and Control of Pretrained Diffusion Model

Because state-of-the-art image diffusion models are dominated by text-to-image methods, the most straight-forward ways to enhance the control over a diffusion model are often text-guided [38, 24, 2, 3, 23, 43, 16]. This type of control can also be achieved by manipulating CLIP features [43]. The image diffusion process by itself can provide some functionalities to achieve color-level detail variations [35] (the community of Stable Diffusion call it img2img). Image diffusion algorithms naturally supports inpainting as an important way to control the results [43, 2]. Textual Inversion [12] and DreamBooth [46] are proposed to customize (or personalize) the contents in the generated results using a small set of images with same topics or objects.

2.5 Image-to-Image Translation

We would like to point out that, although the ControlNet and image-to-image translation may have several overlapped applications, their motivations are essentially different. Image-to-image translation

is targeted to learn a mapping between images in different domains, while a ControlNet is targeted to control a diffusion model with task-specific conditions.

Pix2Pix [20] presented the concept of image-to-image translation, and early methods are dominated by conditional generative neural networks [20, 69, 60, 39, 8, 63, 68]. After transformers and Vision Transformers (ViTs) gained popularity, successful results have been reported using autoregressive methods [42, 11, 7]. Some researches also show that multi-model methods can learn a robust generator from various translation tasks [64, 29, 19, 40].

We discuss the current strongest methods in image-to-image translation. Taming Transformer [11] is a vision transformer with the capability to both generate images and perform image-to-image translations. Palette [48] is an unified diffusion-based image-to-image translation framework. PITI [59] is a diffusion-based image-to-image translation method that utilizes large-scale pretraining as a way to improve the quality of generated results. In specific fields like sketch-guided diffusion, [58] is a optimization-based method that manipulates the diffusion process. These methods are tested in the experiments.

3 Method

ControlNet is a neural network architecture that can enhance pretrained image diffusion models with task-specific conditions. We introduce ControlNet's essential structure and motivate of each part in Section 3.1. We detail the method to apply ControlNets to image diffusion models using the example of Stable Diffusion in Section 3.2. We elaborate the learning objective and general training method in Section 3.3, and then describe several approaches to improve the training in extreme cases such as training with one single laptop or using large-scale computing clusters in Section 3.4. Finally, we include the details of several ControlNet implementations with different input conditions in Section 3.5.

3.1 ControlNet

ControlNet manipulates the input conditions of neural network blocks so as to further control the overall behavior of an entire neural network. Herein, a "network block" refers to a set of neural layers that are put together as a frequently used unit to build neural networks, *e.g.*, "resnet" block, "conv-bn-relu" block, multi-head attention block, transformer block, *etc*.

Using 2D feature as an example, given a feature map $\boldsymbol{x} \in \mathbb{R}^{h \times w \times c}$ with $\{h, w, c\}$ being height, width, and channel numbers, a neural network block $\mathcal{F}(\cdot; \Theta)$ with a set of parameters Θ transforms $\boldsymbol{x}$ into another feature map $\boldsymbol{y}$ with

$$\boldsymbol{y} = \mathcal{F}(\boldsymbol{x}; \Theta) \tag{1}$$

and this procedure is visualized in Fig. 2-(a).

We lock all parameters in Θ and then clone it into a trainable copy Θ_{c}. The copied Θ_{c} is trained with an external condition vector $\boldsymbol{c}$. In this paper, we call the original and new parameters "locked copy" and "trainable copy". The motivation of making such copies rather than directly training the original weights is to avoid overfitting when dataset is small and to preserve the production-ready quality of large models learned from billions of images.

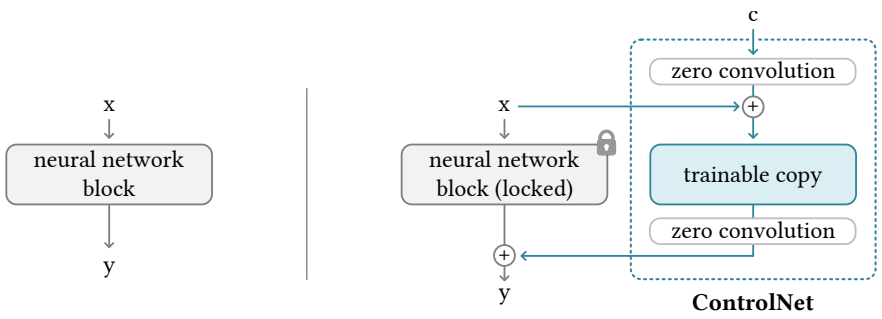
The neural network blocks are connected by an unique type of convolution layer called "zero convolution", *i.e.*, 1×1 convolution layer with both weight and bias initialized with zeros. We denote the zero convolution operation as $\mathcal{Z}(\cdot; \cdot)$ and use two instances of parameters $\{\Theta_{\text{z1}}, \Theta_{\text{z2}}\}$ to compose the ControlNet structure with

$$\boldsymbol{y}_{\text{c}} = \mathcal{F}(\boldsymbol{x}; \Theta) + \mathcal{Z}(\mathcal{F}(\boldsymbol{x} + \mathcal{Z}(\boldsymbol{c}; \Theta_{\text{z1}}); \Theta_{\text{c}}); \Theta_{\text{z2}}) \tag{2}$$

where $\boldsymbol{y}_{\text{c}}$ becomes the output of this neural network block, as visualized in Fig. 2-(b).

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

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



(a) Before (b) After

Figure 2: ControlNet. We show the approach to apply a ControlNet to an arbitrary neural network block. The x, y are deep features in neural networks. The "+" refers to feature addition. The "c" is an extra condition that we want to add to the neural network. The "zero convolution" is an 1×1 convolution layer with both weight and bias initialized as zeros.

and this can be converted to

$$\boldsymbol{y}_{\mathrm{c}} = \boldsymbol{y} \tag{4}$$

and Eq-(2,3,4) indicate that, in the first training step, all the inputs and outputs of both the trainable and locked copy of neural network blocks are consistent with what they would be as if the ControlNet does not exist. In other words, when a ControlNet is applied to some neural network blocks, before any optimization, it will not cause any influence to the deep neural features. The capability, functionality, and result quality of any neural network block is perfectly preserved, and any further optimization will become as fast as fine tuning (compared to train those layers from scratch).

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

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

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 $\beta_{\mathrm{lr}} \neq 0$, if the "outside" gradient $\partial \mathcal{L} / \partial \mathcal{Z}(\boldsymbol{I}; \{\boldsymbol{W}, \boldsymbol{B}\})$ is not zero, we will have

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

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

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

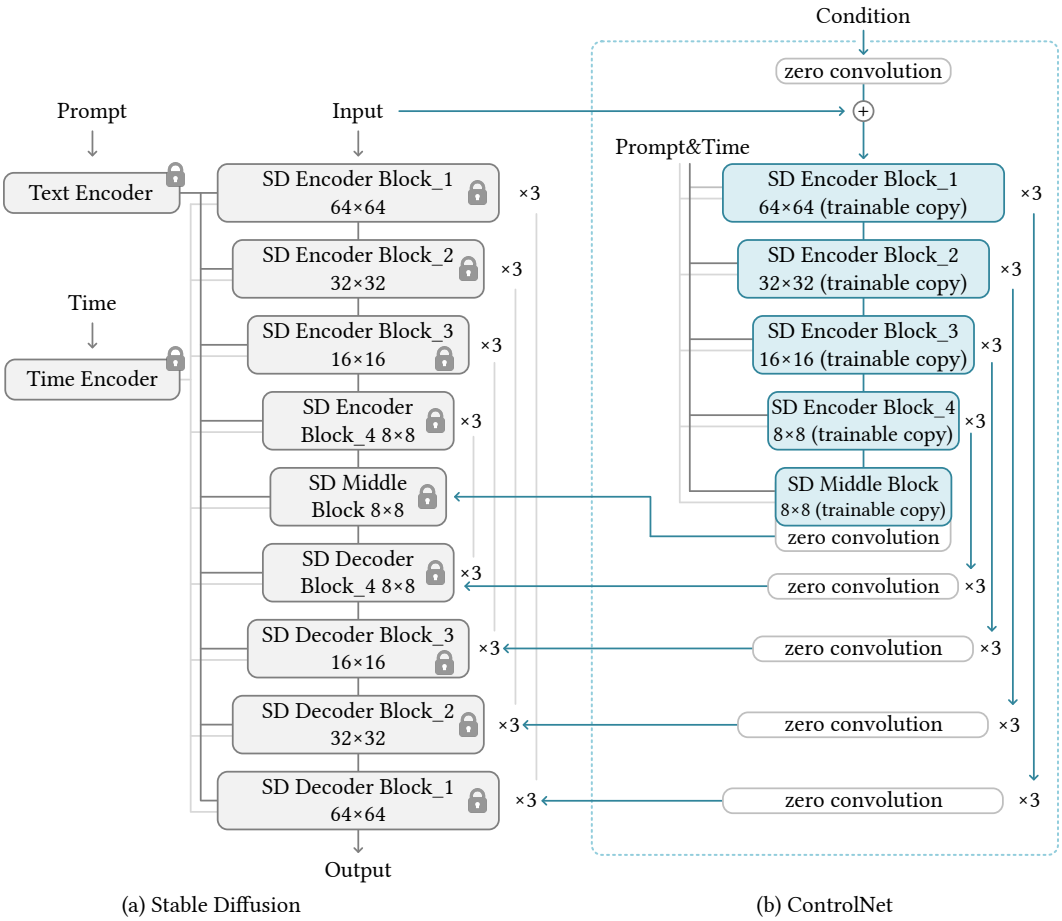


Figure 3: ControlNet in Stable Diffusion. The gray blocks are the structure of Stable Diffusion 1.5 (or SD V2.1, since they use the same U-Net architecture), while the blue blocks are ControlNet.

where non-zero gradients are obtained and the neural network begins to learn. In this way, the zero convolutions become an unique type of connection layer that progressively grow from zeros to optimized parameters in a learned way.

3.2 ControlNet in Image Diffusion Model

We use the Stable Diffusion [44] as an example to introduce the method to use ControlNet to control a large diffusion model with task-specific conditions.

Stable Diffusion is a large text-to-image diffusion model trained on billions of images. The model is essentially an U-net with an encoder, a middle block, and a skip-connected decoder. Both the encoder and decoder have 12 blocks, and the full model has 25 blocks (including the middle block). In those blocks, 8 blocks are down-sampling or up-sampling convolution layers, 17 blocks are main blocks that each contains four resnet layers and two Vision Transformers (ViTs). Each Vit contains several cross-attention and/or self-attention mechanisms. The texts are encoded by OpenAI CLIP, and diffusion time steps are encoded by positional encoding.

Stable Diffusion uses a pre-processing method similar to VQ-GAN [11] to convert the entire dataset of 512×512 images into smaller 64×64 "latent images" for stabilized training. This requires ControlNets to convert image-based conditions to 64×64 feature space to match the convolution size. We use a tiny network $\mathcal{E}(\cdot)$ of four convolution layers with 4×4 kernels and 2×2 strides (activated by ReLU, channels are 16, 32, 64, 128, initialized with Gaussian weights, trained jointly with the full model) to encode image-space conditions $\boldsymbol{c}_\text{i}$ into feature maps with

$$\boldsymbol{c}_\text{f} = \mathcal{E}(\boldsymbol{c}_\text{i}) \tag{9}$$

where $\boldsymbol{c}_\text{f}$ is the converted feature map. This network convert 512×512 image conditions to 64×64 feature maps.

As shown in Fig. 3, we use ControlNet to control each level of the U-net. Note that the way we connect the ControlNet is computationally efficient: since the original weights are locked, no gradient computation on the original encoder is needed for training. This can speed up training and save GPU memory, as half of the gradient computation on the original model can be avoided. Training a stable diffusion model with ControlNet requires only about 23% more GPU memory and 34% more time in each training iteration (as tested on a single Nvidia A100 PCIE 40G).

To be specific, we use ControlNet to create the trainable copy of the 12 encoding blocks and 1 middle block of Stable Diffusion. The 12 blocks are in 4 resolutions ($64 \times 64, 32 \times 32, 16 \times 16, 8 \times 8$) with each having 3 blocks. The outputs are added to the 12 skip-connections and 1 middle block of the U-net. Since SD is a typical U-net structure, this ControlNet architecture is likely to be usable in other diffusion models.

3.3 Training

Image diffusion models learn to progressively denoise images to generate samples. The denoising can happen in pixel space or a "latent" space encoded from training data. Stable Diffusion uses latent images as the training domain. In this context, the terminology "image", "pixel", and "denoising" all refers to corresponding concepts in the "perceptual latent space" [44].

Given an image $\boldsymbol{z}_0$, diffusion algorithms progressively add noise to the image and produces a noisy image $\boldsymbol{z}_t$, with t being how many times the noise is added. When t is large enough, the image approximates pure noise. Given a set of conditions including time step t, text prompts $\boldsymbol{c}_t$, as well as a task-specific conditions $\boldsymbol{c}_\text{f}$, image diffusion algorithms learn a network ϵ_θ to predict the noise added to the noisy image $\boldsymbol{z}_t$ with

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{z}_0, t, \boldsymbol{c}_t, \boldsymbol{c}_\text{f}, \epsilon \sim \mathcal{N}(0,1)} \Big[\| \epsilon - \epsilon_\theta(z_t, t, \boldsymbol{c}_t, \boldsymbol{c}_\text{f})) \|_2^2 \Big] \tag{10}$$

where $\mathcal{L}$ is the overall learning objective of the entire diffusion model. This learning objective can be directly used in fine tuning diffusion models.

During the training, we randomly replace 50% text prompts $\boldsymbol{c}_t$ with empty strings. This facilitates ControlNet's capability to recognize semantic contents from input condition maps, *e.g.*, Canny edge maps or human scribbles, *etc*. This is mainly because when the prompt is not visible for the SD model, the encoder tends to learn more semantics from input control maps as a replacement for the prompt.

3.4 Improved Training

We discuss several strategies to improve the training of ControlNets, especially in extreme cases when the computation device is very limited (*e.g.*, on a laptop) or very powerful (*e.g.*, on a computation cluster with large-scale GPUs available). In our experiments, if any of these strategies are used, we will mention in the experimental settings.

Small-Scale Training When computation device is limited, we find that partially breaking the connection between a ControlNet and the Stable Diffusion can accelerate convergence. By default, we connect the ControlNet to "SD Middle Block" and "SD Decoder Block 1,2,3,4" as shown in Fig. 3. We find that disconnecting the link to decoder 1,2,3,4 and only connecting the middle block can improve the training speed by about a factor of 1.6 (tested on RTX 3070TI laptop GPU). When the model shows reasonable association between results and conditions, those disconnected links can be connected again in a continued training to facilitate accurate control.

Large-Scale Training Herein, the large-scale training refers to the situation where both powerful computation clusters (at least 8 Nvidia A100 80G or equivalent) and large dataset (at least 1 million of training image pairs) are available. This usually applies to tasks where data is easily available, *e.g.*, edge maps detected by Canny. In this case, since the risk of over-fitting is relatively low, we can first train ControlNets for a large enough number of iterations (usually more than 50k steps), and then unlock all weights of the Stable Diffusion and jointly train the entire model as a whole. This would lead to a more problem-specific model.

3.5 Implementation

We present several implementations of ControlNets with different image-based conditions to control large diffusion models in various ways.

Canny Edge We use Canny edge detector [5] (with random thresholds) to obtain 3M edge-image-caption pairs from the internet. The model is trained with 600 GPU-hours with Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 4.)

Canny Edge (Alter) We rank the image resolutions of the above Canny edge dataset and sampled several sub-set with 1k, 10k, 50k, 500k samples. We use the same experimental setting to test the effect of dataset scale. (See also Fig. 22.)

Hough Line We use a learning-based deep Hough transform [13] to detect straight lines from Places2 [66], and then use BLIP [34] to generate captions. We obtain 600k edge-image-caption pairs. We use the above Canny model as a starting checkpoint and train the model with 150 GPU-hours with Nvidia A100 80G. (See also Fig. 5.)

HED Boundary We use HED boundary detection [62] to obtain 3M edge-image-caption pairs from internet. The model is trained with 300 GPU-hours with Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 7.)

User Sketching We synthesize human scribbles from images using a combination of HED boundary detection [62] and a set of strong data augmentations (random thresholds, randomly masking out a random percentage of scribbles, random morphological transformations, and random non-maximum suppression). We obtain 500k scribble-image-caption pairs from internet. We use the above Canny model as a starting checkpoint and train the model with 150 GPU-hours with Nvidia A100 80G. Note that we also tried a more "human-like" synthesizing method [57] but the method is much slower than a simple HED and we do not notice visible improvements. (See also Fig. 6.)

Human Pose (Openpifpaf) We use learning-based pose estimation method [27] to "find" humans from internet using a simple rule: an image with human must have at least 30% of the key points of the whole body detected. We obtain 80k pose-image-caption pairs. Note that we directly use visualized pose images with human skeletons as training condition. The model is trained with 400 GPU-hours on Nvidia RTX 3090TI. The base model is Stable Diffusion 2.1. (See also Fig. 8.)

Human Pose (Openpose) We use learning-based pose estimation method [6] to find humans from internet using the same rule in the above Openpifpaf setting. We obtain 200k pose-image-caption pairs. Note that we directly use visualized pose images with human skeletons as training condition. The model is trained with 300 GPU-hours with Nvidia A100 80G. Other settings are same with the above Openpifpaf. (See also Fig. 9.)

Semantic Segmentation (COCO) The COCO-Stuff dataset [4] captioned by BLIP [34]. We obtain 164K segmentation-image-caption pairs. The model is trained with 400 GPU-hours on Nvidia RTX 3090TI. The base model is Stable Diffusion 1.5. (See also Fig. 12.)

Semantic Segmentation (ADE20K) The ADE20K dataset [67] captioned by BLIP [34]. We obtain 164K segmentation-image-caption pairs. The model is trained with 200 GPU-hours on Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 11.)

Depth (large-scale) We use the Midas [30] to obtain 3M depth-image-caption pairs from internet. The model is trained with 500 GPU-hours with Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 23,24,25.)

Depth (small-scale) We rank the image resolutions of the above depth dataset to sample a subset of 200k pairs. This set is used in experimenting the minimal required dataset size to train the model. (See also Fig. 14.)

Normal Maps The DIODE dataset [56] captioned by BLIP [34]. We obtain 25,452 normal-image-caption pairs. The model is trained with 100 GPU-hours on Nvidia A100 80G. The base model is Stable Diffusion 1.5. (See also Fig. 13.)

Normal Maps (extended) We use the Midas [30] to compute depth map and then perform normal-from-distance to achieve "coarse" normal maps. We use the above Normal model as a starting checkpoint and train the model with 200 GPU-hours with Nvidia A100 80G. (See also Fig. 23,24,25.)

Cartoon Line Drawing We use a cartoon line drawing extracting method [61] to extract line drawings from cartoon illustration from internet. By sorting the cartoon images with popularity, we obtain the top 1M lineart-cartoon-caption pairs. The model is trained with 300 GPU-hours with Nvidia A100 80G. The base model is Waifu Diffusion (an interesting community-developed variation model from stable diffusion [36]). (See also Fig. 15.)

4 Experiment

4.1 Experimental Settings

All results in this paper is achieved with CFG-scale at 9.0. The sampler is DDIM. We use 20 steps by default. We use three types of prompts to test the models:

(1) No prompt: We use empty string "" as prompt.

(2) Default prompt: Since Stable diffusion is essentially trained with prompts, the empty string might be an unexpected input for the model, and SD tends to generate random texture maps if no prompt is provided. A better setting is to use meaningless prompts like "an image", "a nice image", "a professional image", *etc*. In our setting, we use "a professional, detailed, high-quality image" as default prompt.

(3) Automatic prompt: In order to test the state-of-the-art maximized quality of a fully automatic pipeline, we also try to use automatic image captioning methods (*e.g*., BLIP [34]) to generate prompts using the results obtained by "default prompt" mode. We use the generated prompt to diffusion again.

(4) User prompt: Users give the prompts.

4.2 Qualitative Results

We present qualitative results in Fig. 4, 5,6,7,8,9,10,11,12,13,14,15.

4.3 Ablation Study

Fig. 20 shows a comparison to a model trained without using ControlNet. That model is trained with exactly same method with Stability's Depth-to-Image model (Adding a channel to the SD and continue the training).

Fig. 21 shows the training process. We would like to point out a "sudden convergence phenomenon" where the model suddenly be able to follow the input conditions. This can happen during the training process from 5000 to 10000 steps when using 1e-5 as the learning rate.

Fig. 22 shows Canny-edge-based ControlNets trained with different dataset scales.

4.4 Comparison to previous methods

Fig. 14 shows the comparison to Stability's Depth-to-Image model.

Fig. 17 shows a comparison to PITI [59].

Fig. 18 shows a comparison to sketch-guided diffusion [58].

Fig. 19 shows a comparison to Taming transformer [11].

4.5 Comparison of pre-trained models

We show comparisons of different pre-trained models in Fig. 23, 24, 25.

4.6 More Applications

Fig. 16 show that if the diffusion process is masked, the models can be used in pen-based image editing.

Fig. 26 show that when object is relatively simple, the model can achieve relatively accurate control of the details.

Fig. 27 shows that when ControlNet is only applied to 50% diffusion iterations, users can get results that do not follow the input shapes.

5 Limitation

Fig. 28 shows that when the semantic interpretation is wrong, the model may have difficulty to generate correct contents.

Appendix

Fig. 29 shows all source images in this paper for edge detection, pose extraction, *etc*.

References

[1] Y. Alaluf, O. Tov, R. Mokady, R. Gal, and A. H. Bermano. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. 2021.

[2] O. Avrahami, D. Lischinski, and O. Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.

[3] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions, 2022.

[4] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context, 2016.

[5] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986.

[6] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[7] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021.

[8] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[9] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *CoRR*, 2105, 2021.

[10] T. M. Dinh, A. T. Tran, R. Nguyen, and B.-S. Hua. Hyperinverter: Improving stylegan inversion via hypernetwork. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11389–11398, 2022.

[11] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. *CoRR*, 2012, 2020.

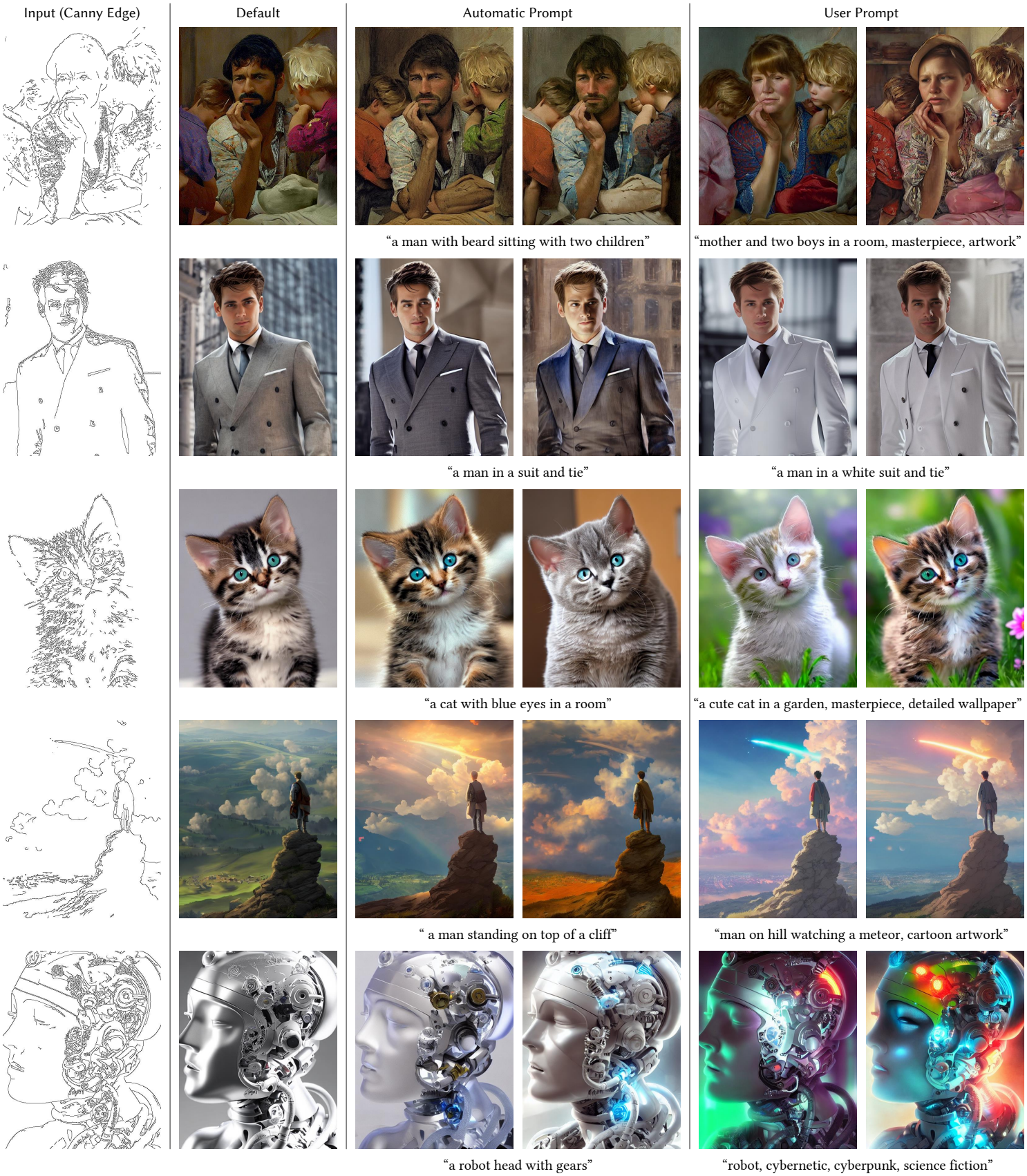


Figure 4: Controlling Stable Diffusion with Canny edges. The “automatic prompts” are generated by BLIP based on the default result images without using user prompts. See also the Appendix for source images for canny edge detection.

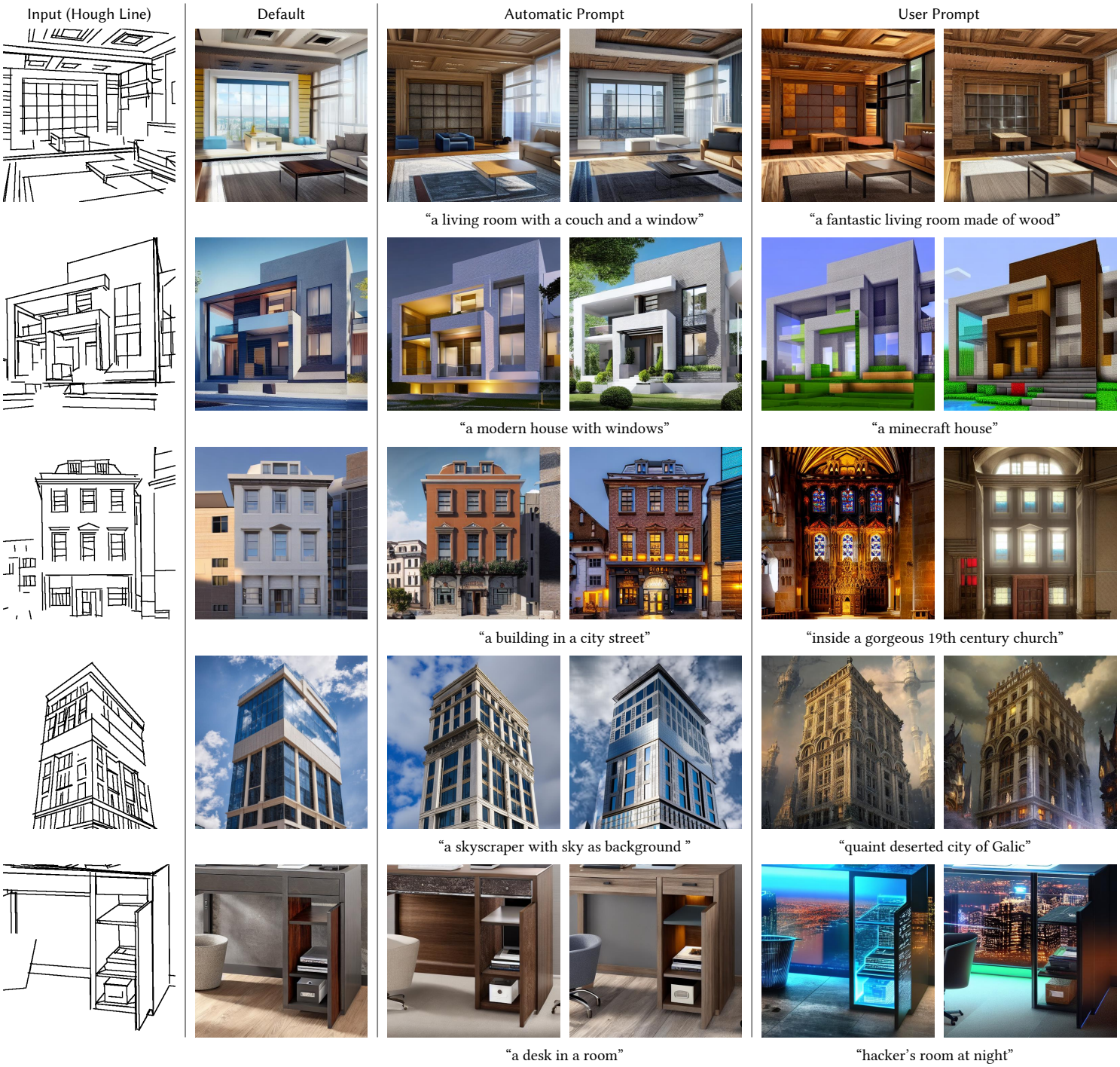


Figure 5: Controlling Stable Diffusion with Hough lines (M-LSD). The “automatic prompts” are generated by BLIP based on the default result images without using user prompts. See also the Appendix for source images for line detection.

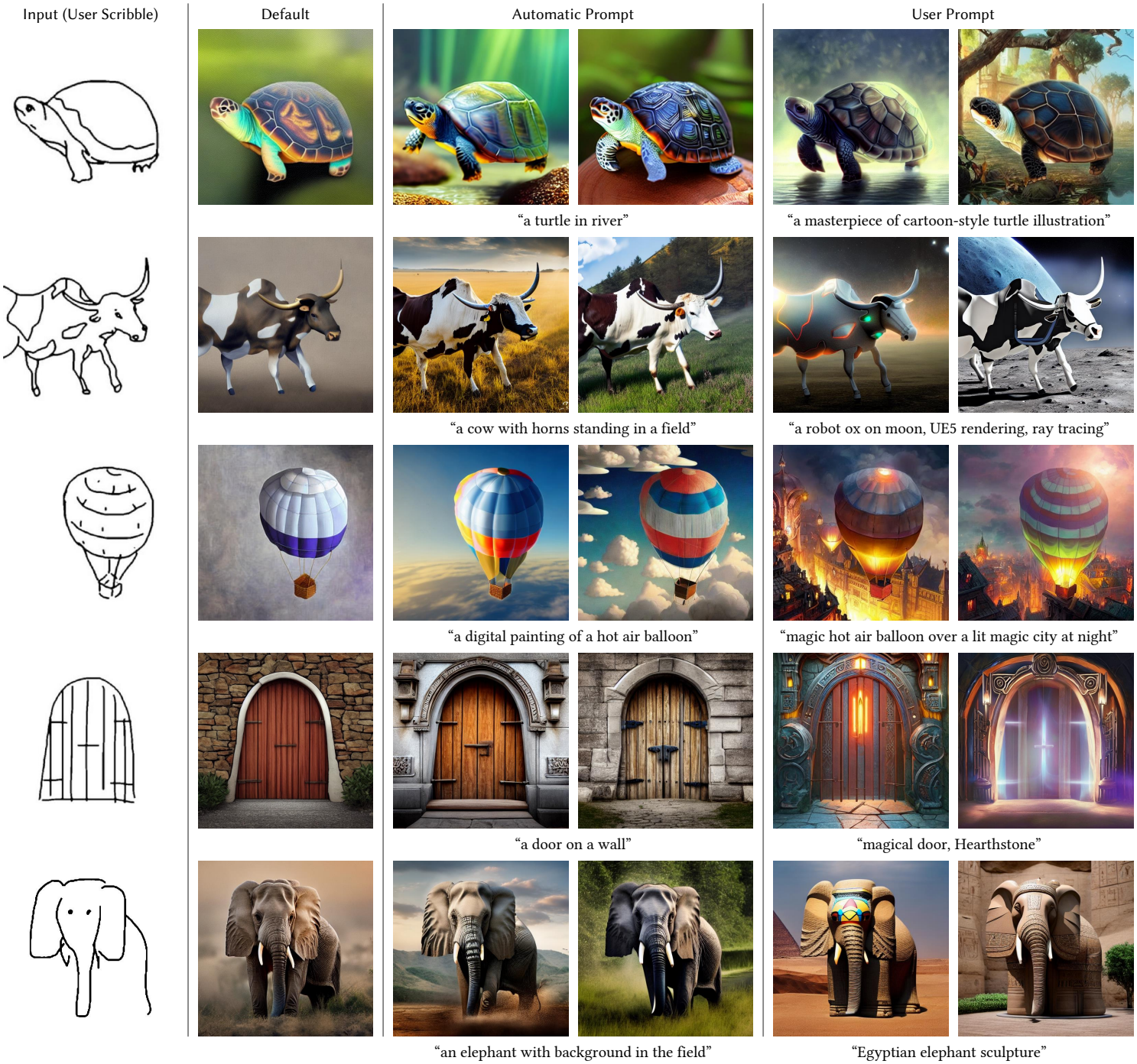


Figure 6: Controlling Stable Diffusion with Human scribbles. The “automatic prompts” are generated by BLIP based on the default result images without using user prompts. These scribbles are from [58].

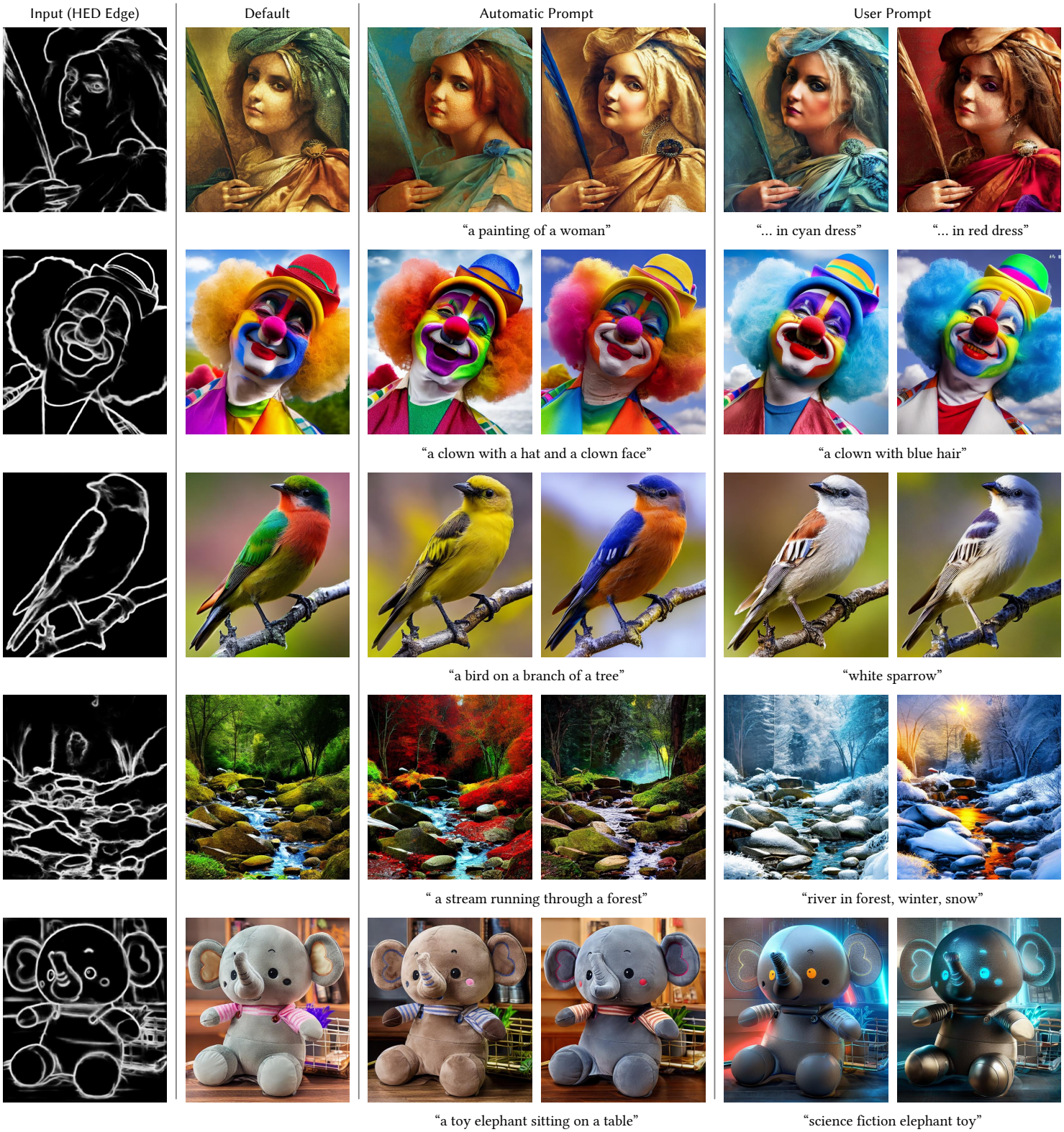


Figure 7: Controlling Stable Diffusion with HED boundary map. The “automatic prompts” are generated by BLIP based on the default result images without using user prompts. See also the Appendix for source images for HED boundary detection.

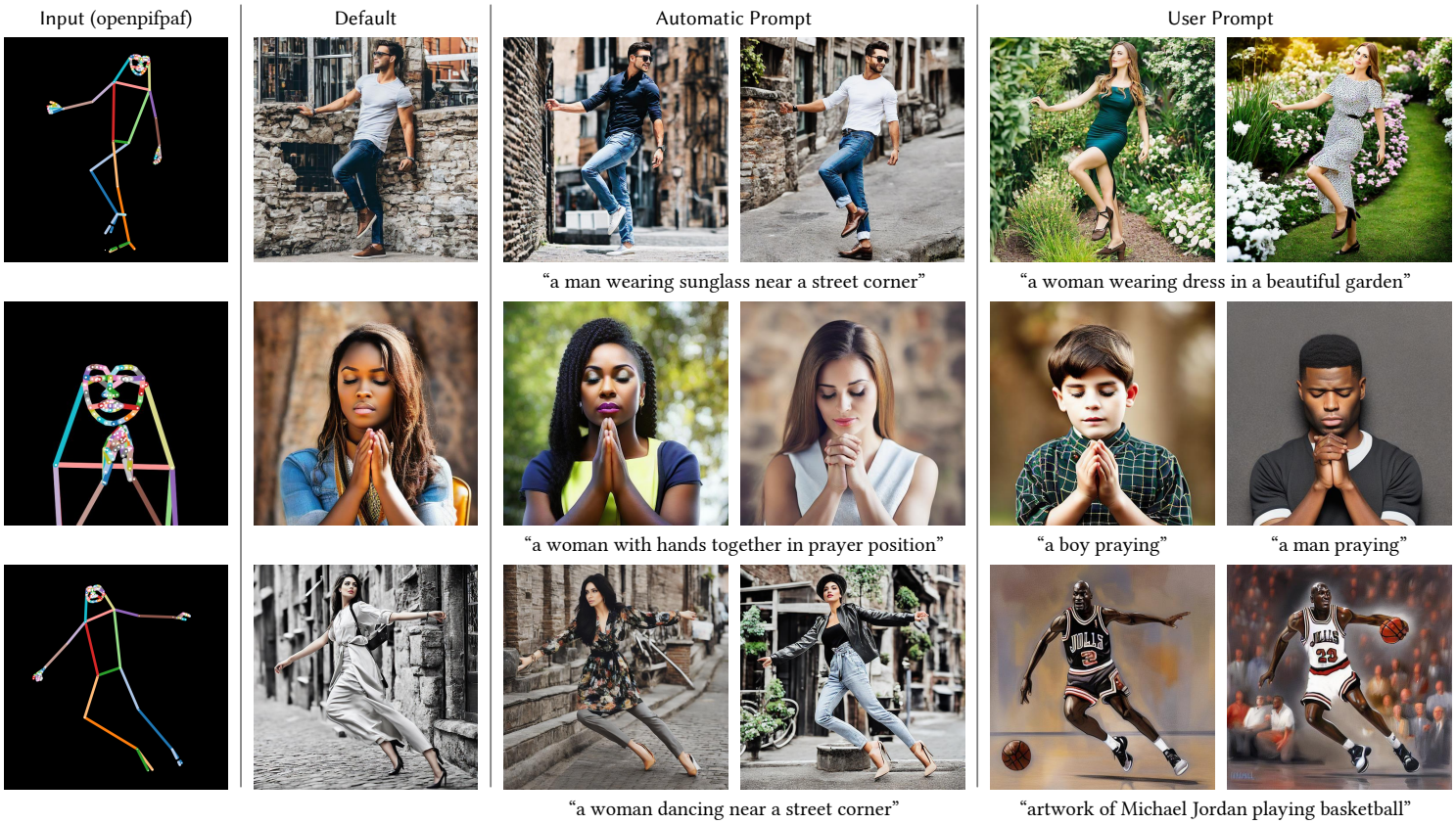


Figure 8: Controlling Stable Diffusion with Openpifpaf pose. See also the Appendix for source images for Openpifpaf pose detection.

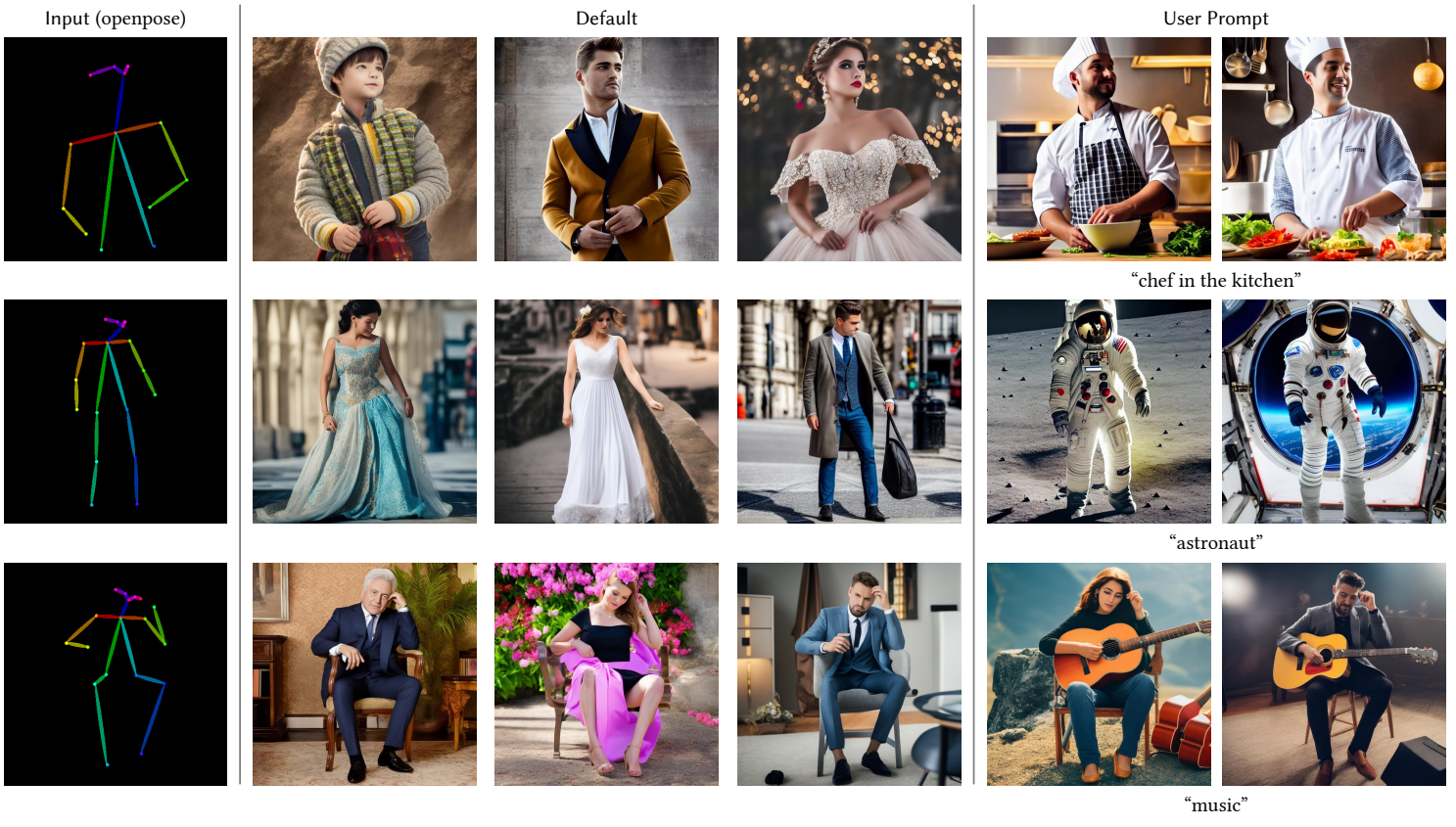


Figure 9: Controlling Stable Diffusion with Openpose. See also the Appendix for source images for Openpose pose detection.

“Michael Jackson's concert”

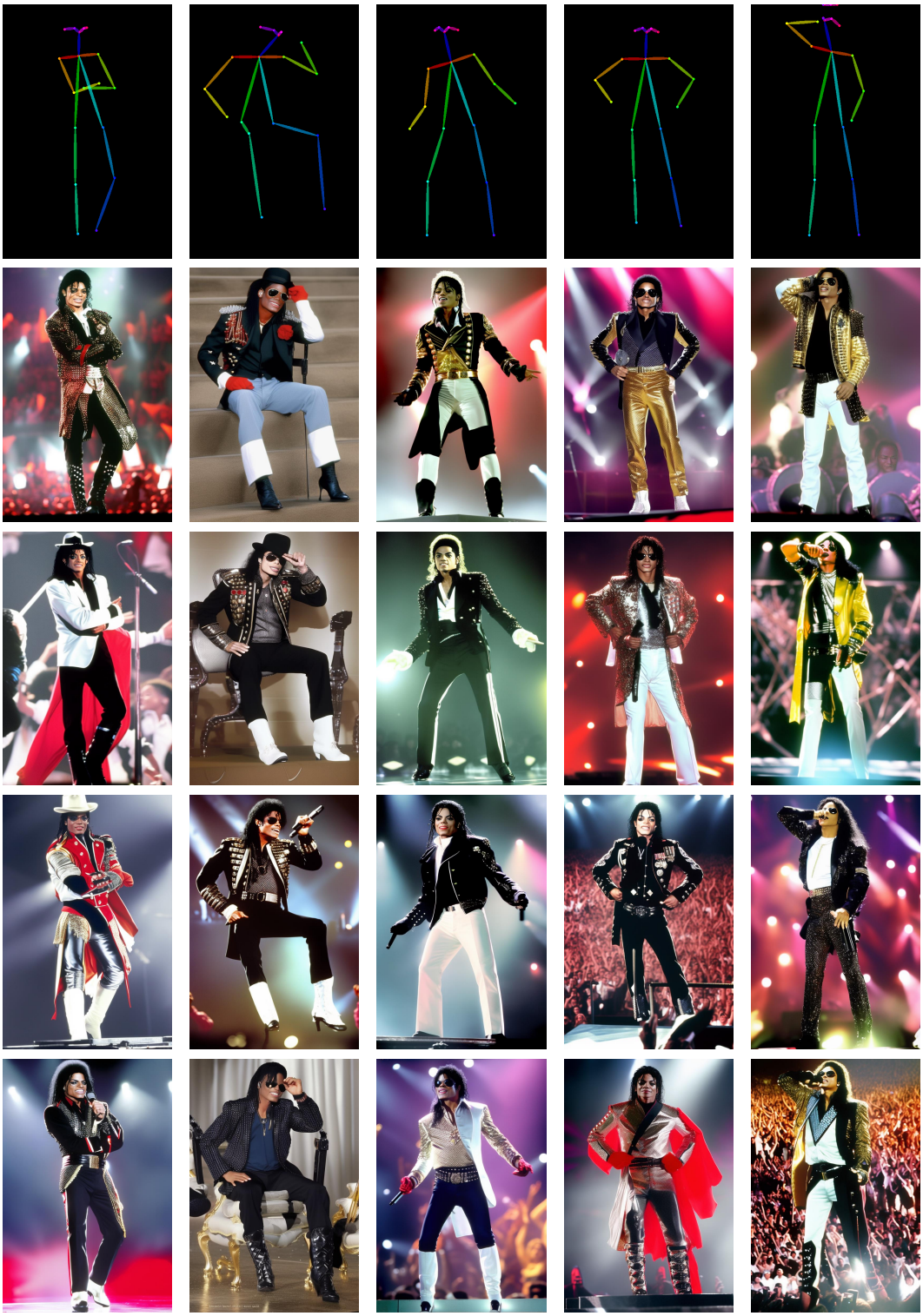


Figure 10: Controlling Stable Diffusion with human pose to generate different poses for a same person (“Michael Jackson’s concert”). Images are not cherry picked. See also the Appendix for source images for Openpose pose detection.

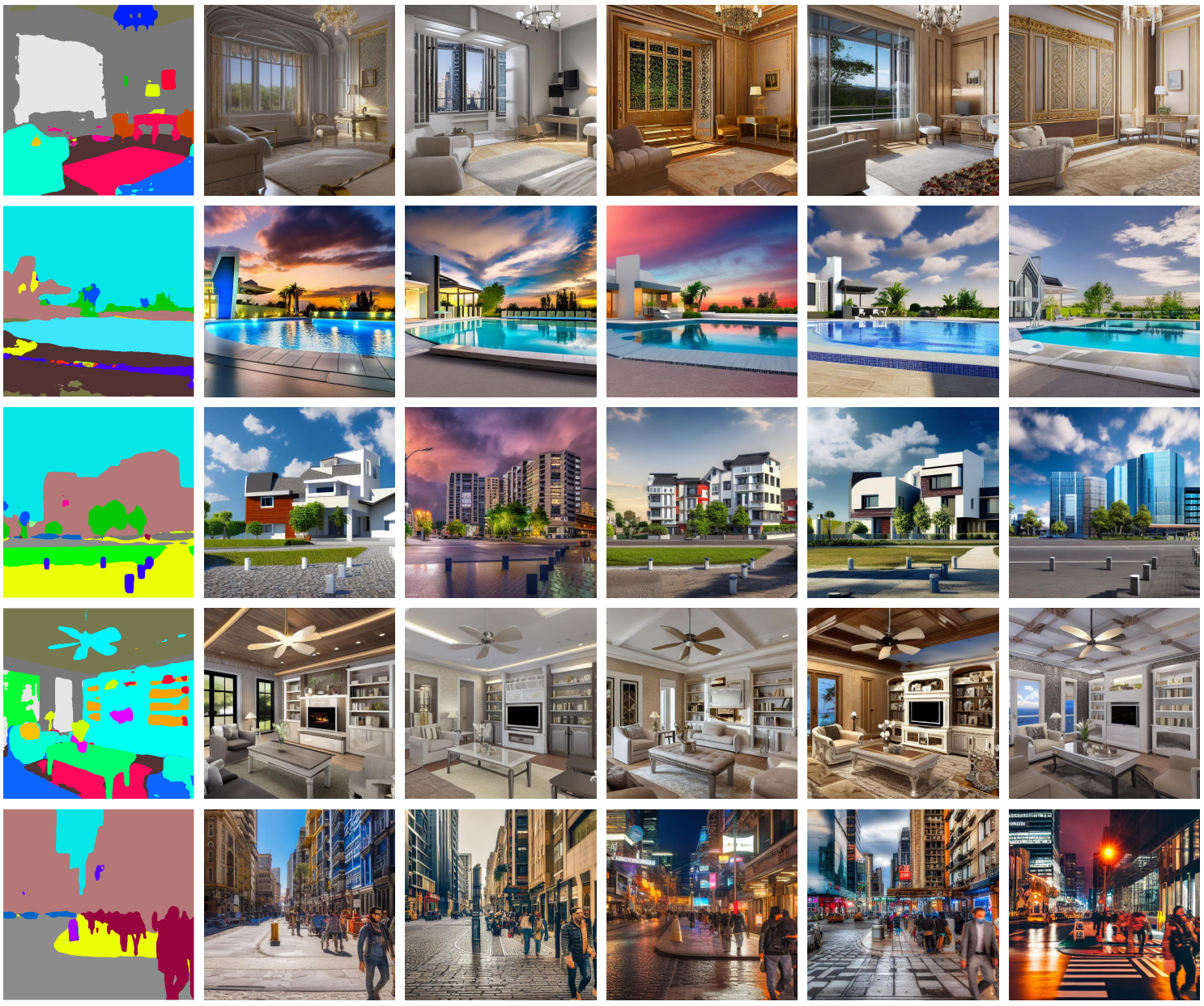


Figure 11: Controlling Stable Diffusion with ADE20K [67] segmentation map. All results are achieved with default prompt. See also the Appendix for source images for semantic segmentation map extraction.

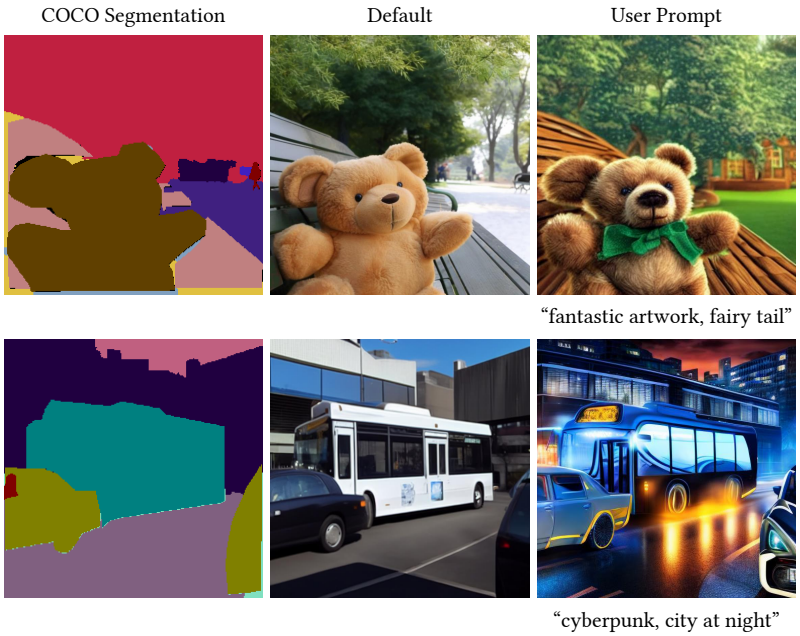


Figure 12: Controlling Stable Diffusion with COCO-Stuff [4] segmentation map.

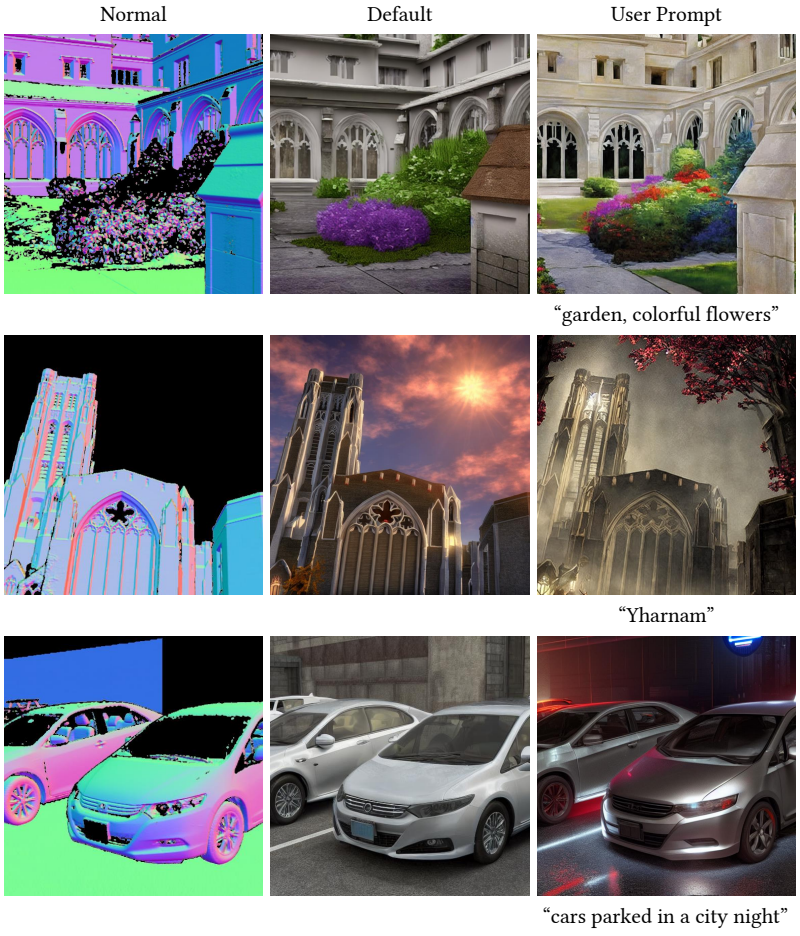
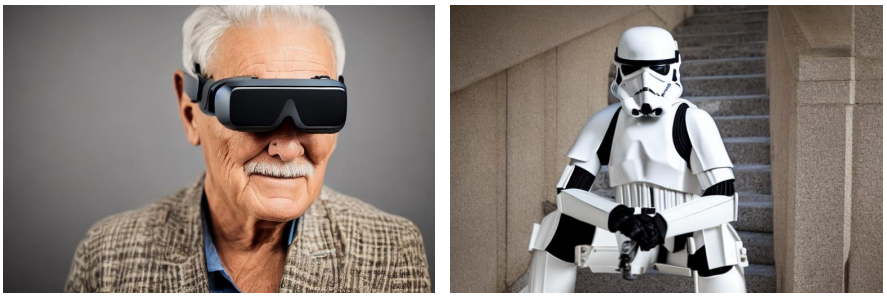


Figure 13: Controlling Stable Diffusion with DIODE [56] normal map.

Images and Midas Depth

Stable Diffusion V2 Depth-to-Image
resumed from **SD 2.0**, continued training on **Large-scale Nvidia A100 Clusters**,
more than **12M** training data, **more than 2000 GPU-hours** (estimation)



Stable Diffusion with **Depth-based ControlNet**
controlling **SD 1.5**, trained on **one single Nvidia RTX 3090TI**,
with **200K** training data, trained **less than one week**

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.

Cartoon line drawing "1girl, masterpiece, best quality, ultra-detailed, illustration"

Figure 15: Controlling Stable Diffusion (anime weights) with cartoon line drawings. The line drawings are inputs and there are no corresponding "ground truths". This model may be used in artistic creation tools.

[12] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[13] G. Gu, B. Ko, S. Go, S.-H. Lee, J. Lee, and M. Shin. Towards light-weight and real-time line segment detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[14] D. Ha, A. M. Dai, and Q. V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017.

[15] Heathen. github.com/automatic1111/stable-diffusion-webui/discussions/2670, hypernetwork style training, a tiny guide, 2022.

[16] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[17] J. Ho, A. Jain, and P. Abbeel. *Denoising diffusion probabilistic models*. NeurIPS, 2020.

[18] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans. Cascaded diffusion models for high fidelity image generation. *CoRR*, 2106:15282, 2021.

[19] X. Huang, A. Mallya, T.-C. Wang, and M.-Y. Liu. Multimodal conditional image synthesis with product-of-experts gans. *arXiv preprint arXiv:2112.05130*, 2021.

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

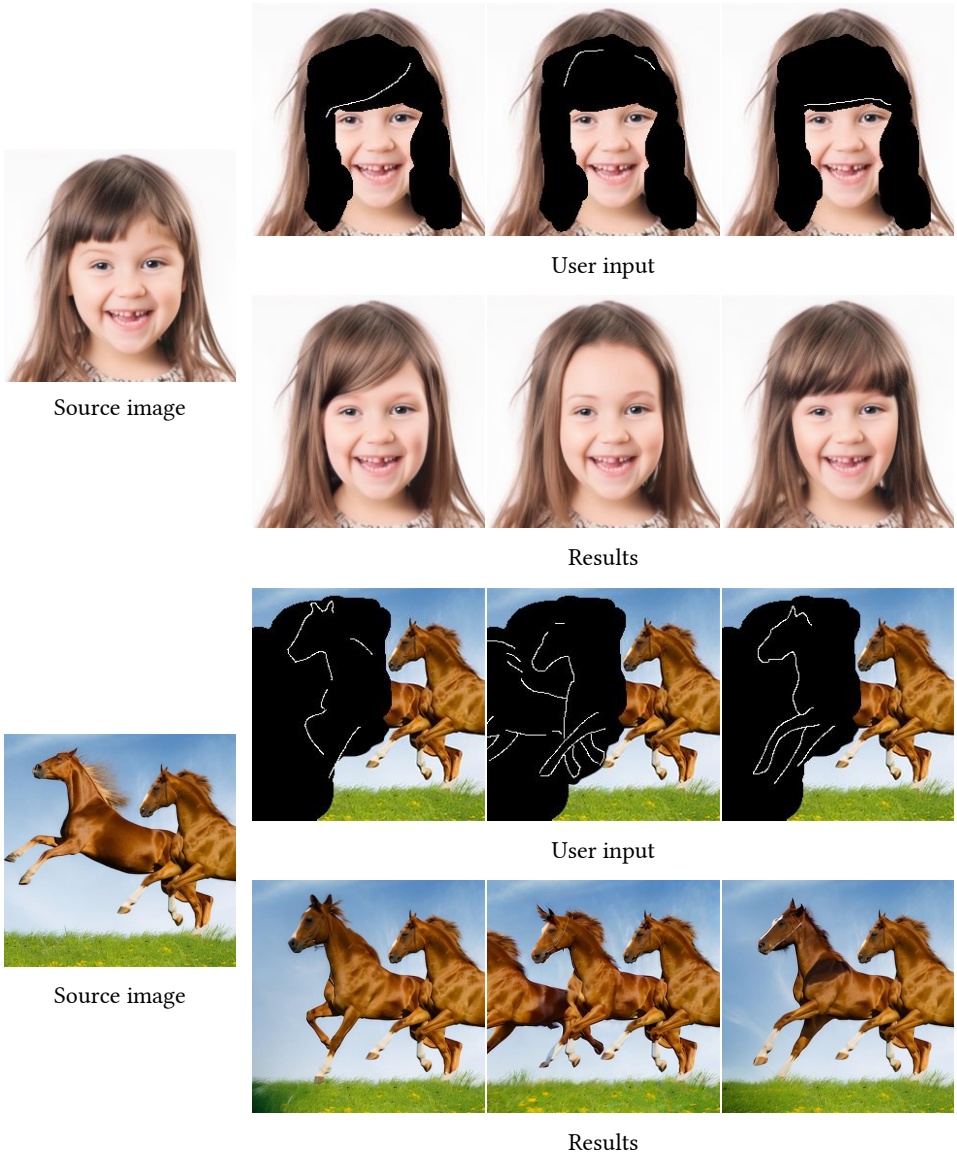
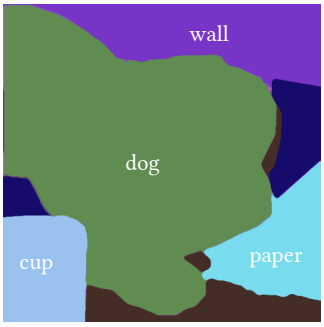
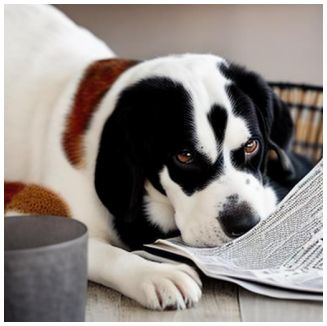


Figure 16: Masked Diffusion. By diffusing images in masked areas, the Canny-edge model can be used to support pen-based editing of image contents. Since all diffusion models naturally support masked diffusion, the other models are also likely to be used in manipulating images.



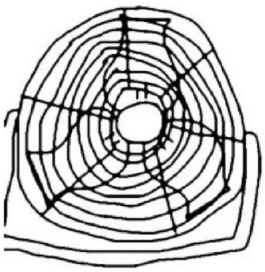
Segmentation

PITI, Wang *et.al.* 2022



Ours (default) Ours ("golden retriever")

Figure 17: Comparison to Pretraining-Image-to-Image (PITI) [59]. Note that the semantic consistency of the "wall", "paper", and "cup" is difficult to handle in this task.

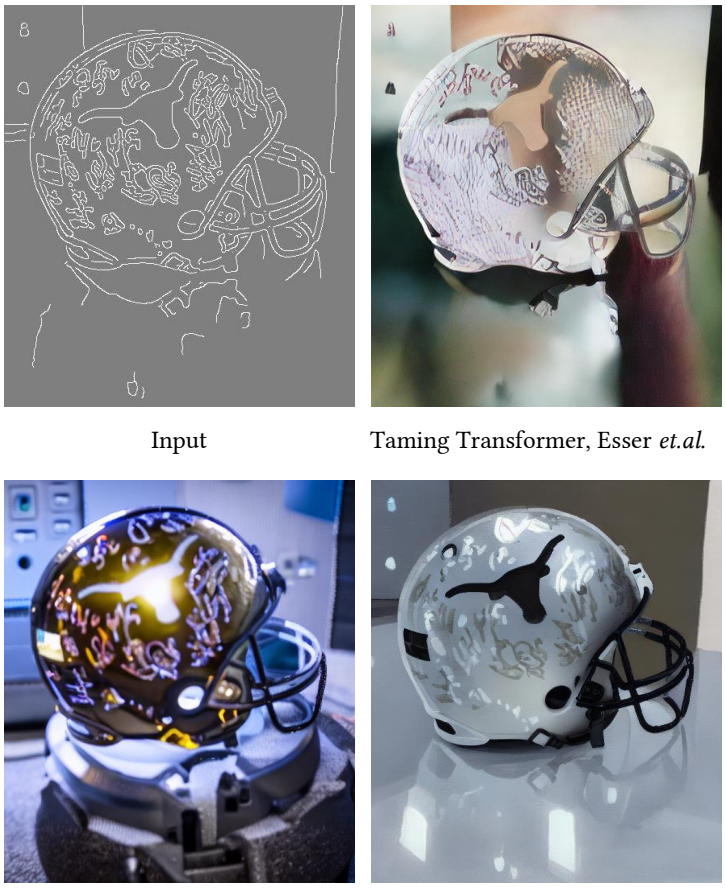


User Input

Sketch-Guided, Voynov *et.al.* 2022

Ours ("electric fan")

Figure 18: Comparison to Sketch-guided diffusion [58]. This input is one of the most challenging cases in their paper.



Input | Taming Transformer, Esser *et.al.*

Ours (default) | Ours ("white helmet on table")

Figure 19: Comparison to Taming Transformers [11]. This input is one of the most challenging cases in their paper.

[21] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017.

[22] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks, 2018.

[23] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani. Imagic: Text-based real image editing with diffusion models. *arXiv preprint arXiv:2210.09276*, 2022.

[24] G. Kim, T. Kwon, and J. C. Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022.

[25] D. P. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. 2107:00630, 2021.

[26] Z. Kong and W. Ping. On fast sampling of diffusion probabilistic models. *CoRR*, 2106, 2021.

[27] S. Kreiss, L. Bertoni, and A. Alahi. OpenPifPaf: Composite Fields for Semantic Keypoint Detection and Spatio-Temporal Association. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–14, March 2021.

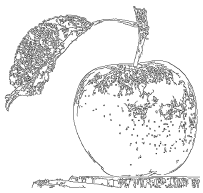
[28] Kurumuz. https://blog.novelai.net/novelai-improvements-on-stable-diffusion-e10d38db82ac, novelai improvements on stable diffusion, 2022.

Input canny map
Same CFG scale (9.0)
Same DDIM sampler
Same default prompt setting
("detailed high-quality professional image" without mentioning image contents)

without ControlNet
(using Stability's "official" method to add the channels to input layer, same as their depth-to-image structure)

SD + ControlNet

Figure 20: Ablative study. We compare the ControlNet structure with a standard method that Stable Diffusion uses as default way to add conditions to diffusion models.



Test condition

Same prompt:
“apple”
+ default “a detailed high-quality professional image”
Same CFG scale (9.0)

Learning rate 1e-5
AdamW
without using tricks like ema

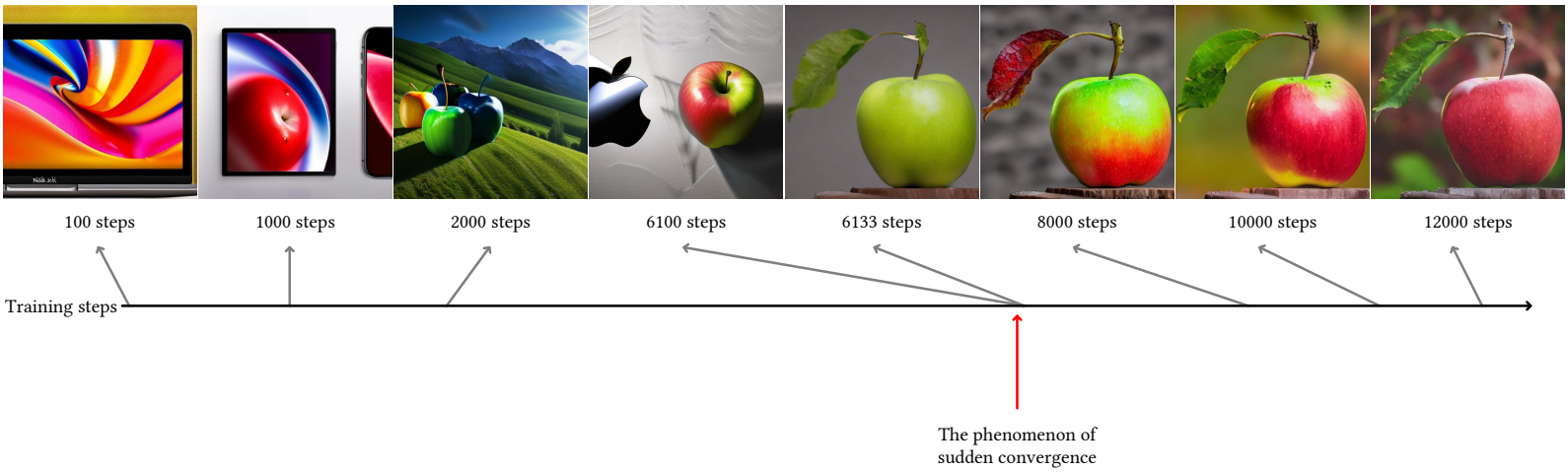


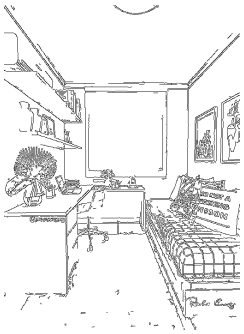
Figure 21: The sudden converge phenomenon. Because we use zero convolutions, the neural network always predict high-quality images during the entire training. At a certain point of training step, the model suddenly learns to adapt to the input conditions. We call this “sudden converge phenomenon”.

Input Canny edge | 1k training samples | 10k training samples | 50k training samples | 500k training samples | 3m training samples

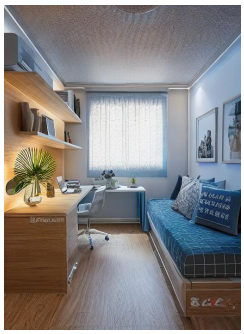
Figure 22: Training on different scale. We show the Canny-edge-based ControlNet trained on different experimental settings with various dataset size.

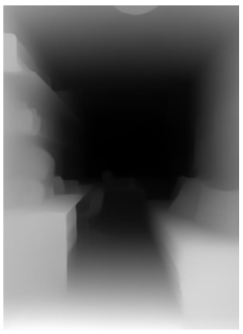
Source Image

Same prompt:
“room”
+ default “a detailed high-quality professional image”
Same CFG scale (9.0)

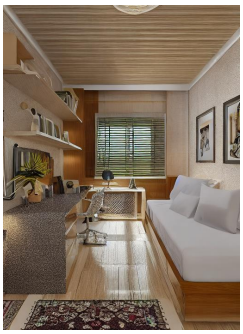


Canny Edge

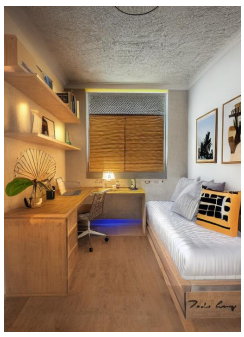


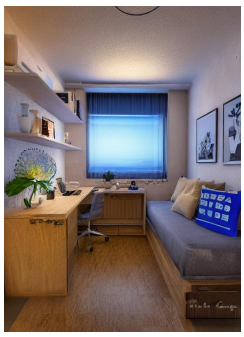


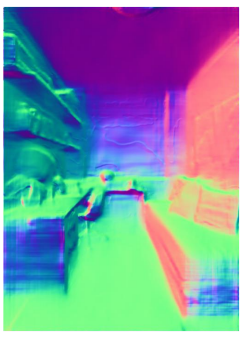
Depth (midas)



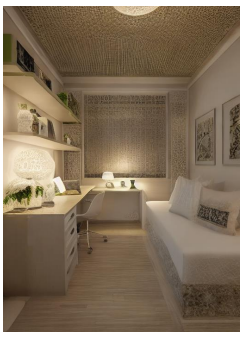
HED

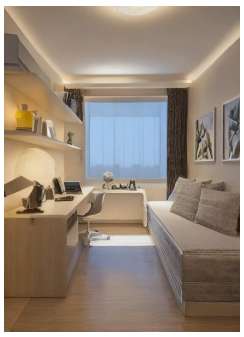


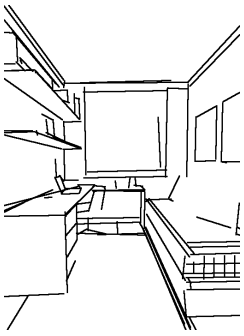




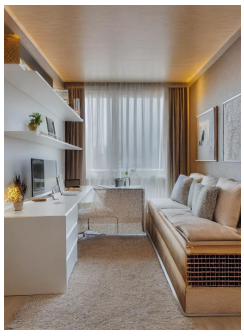
Normal (from midas)

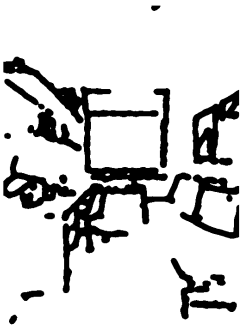






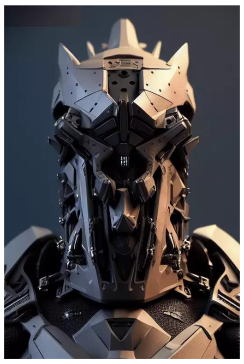
Line (M-LSD)





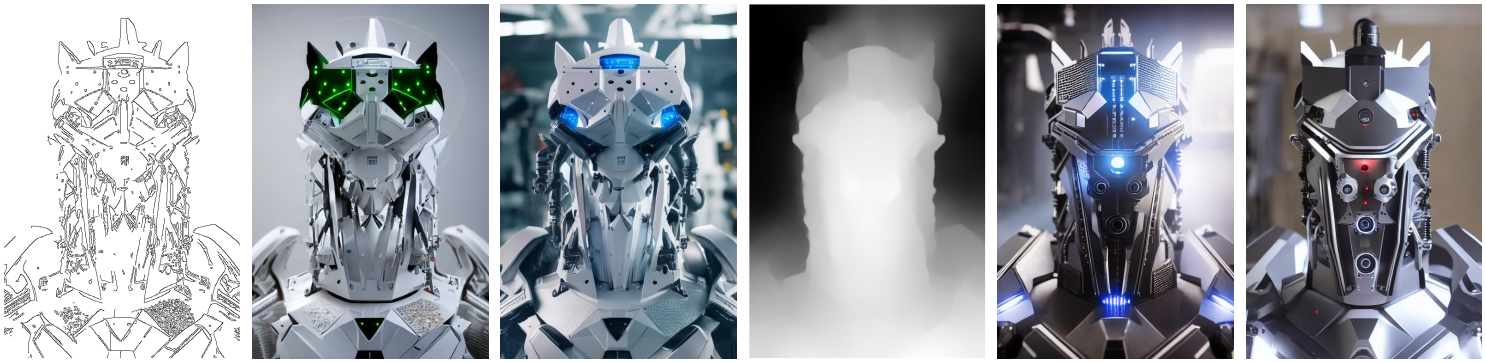
Scribbles (synthesized)

Figure 23: Comparison of six detection types and the corresponding results. The scribble map is extracted from the HED map with morphological transforms.



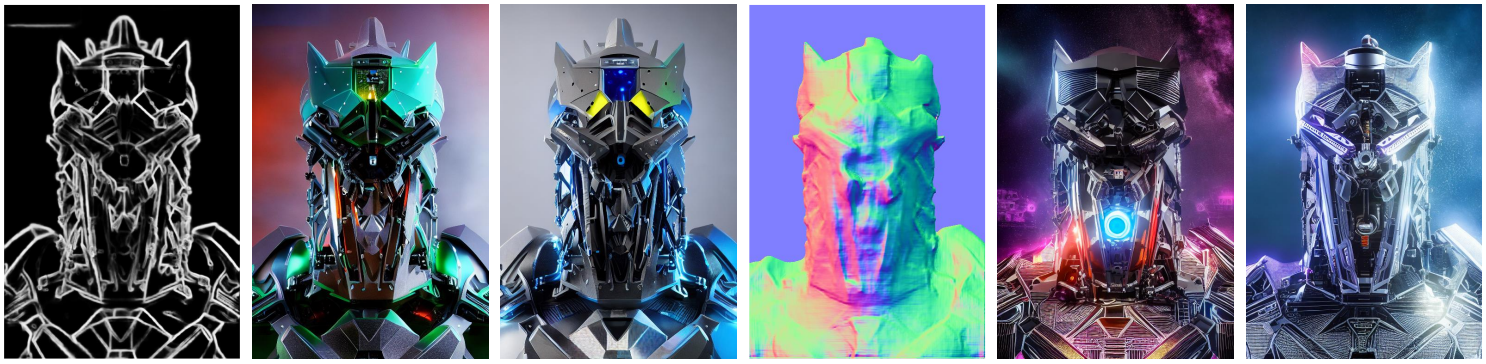
Same prompt:
“robotics”
+ default “a detailed high-quality professional image”
Same CFG scale (9.0)

Source Image



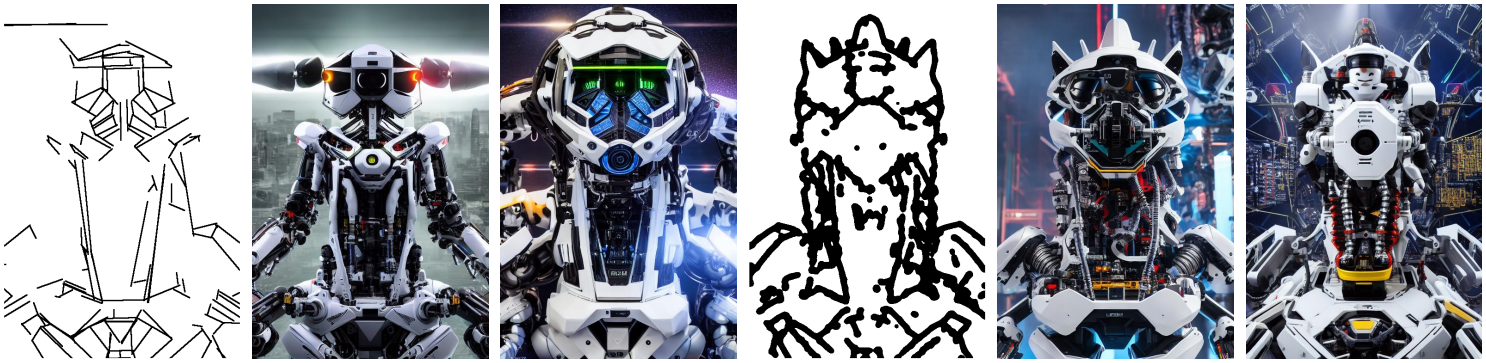
Canny Edge

Depth (midas)



HED

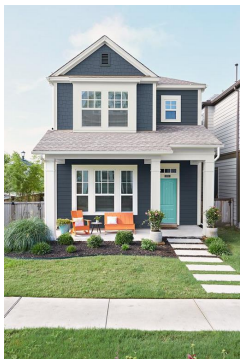
Normal (from midas)



Line (M-LSD)

Scribbles (synthesized)

Figure 24: (Continued) Comparison of six detection types and the corresponding results. The scribble map is extracted from the HED map with morphological transforms.



Source Image

Same prompt:
“house”
+ default “a detailed high-quality professional image”
Same CFG scale (9.0)

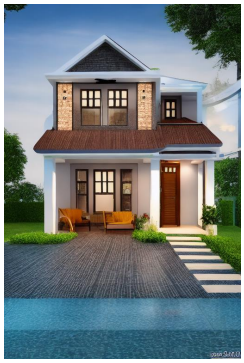
Canny Edge

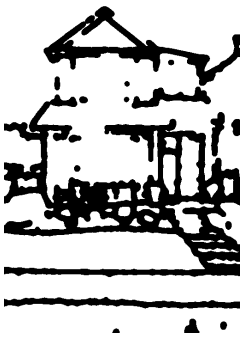
Depth (midas)

HED

Normal (from midas)

Line (M-LSD)



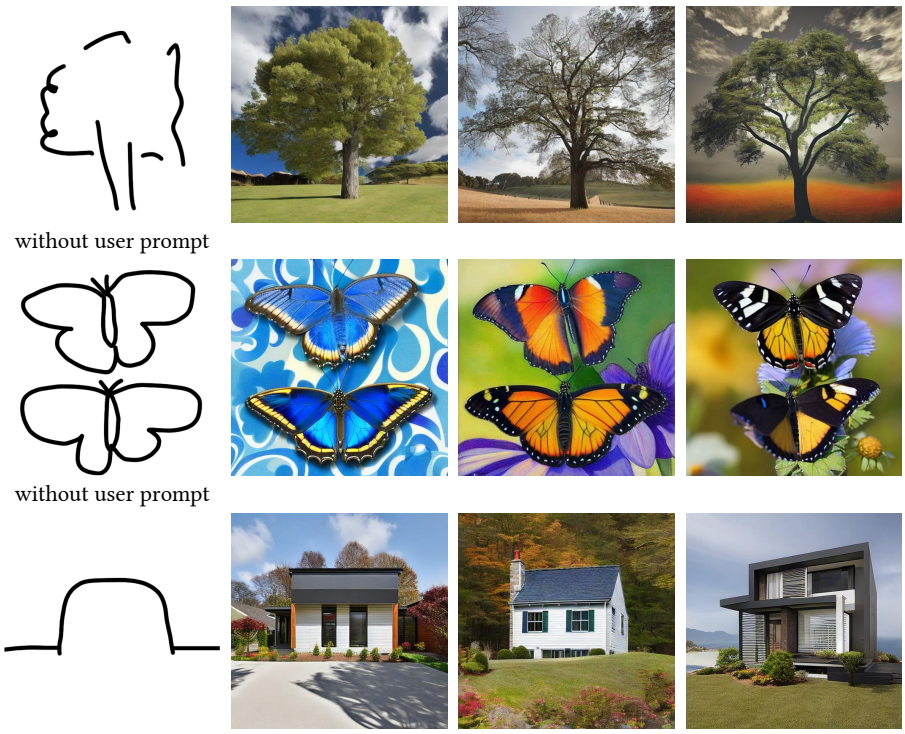


Scribbles (synthesized)

Figure 25: (Continued) Comparison of six detection types and the corresponding results. The scribble map is extracted from the HED map with morphological transforms.

Input Canny edge | Output without prompt | “a watch in black color”

Figure 26: Example of simple object. When the diffusion content is relatively simple, the model can achieve very accurate control to manipulate the content materials.

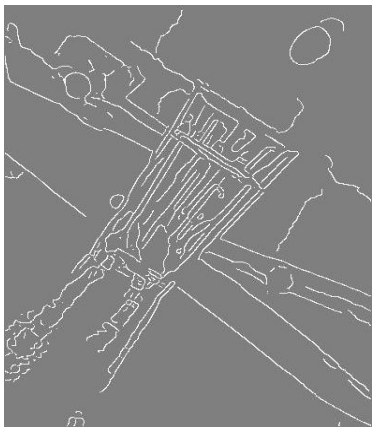


without user prompt

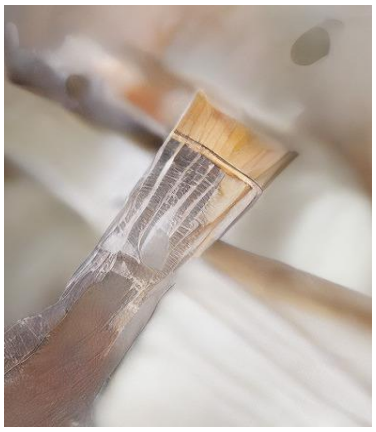
without user prompt

“house”

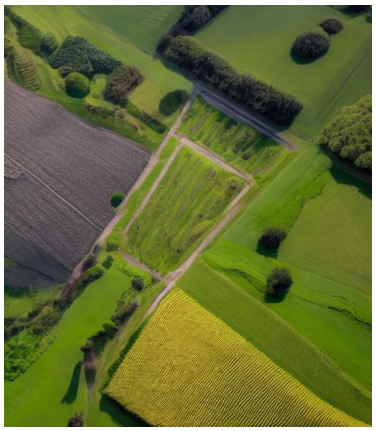
Figure 27: Coarse-level control. When users do not want their input shape to be preserved in the images, we can simply replace the last 50% diffusion iterations with standard SD without ControlNet. The resulting effect is similar to image retrieval but those images are generated.



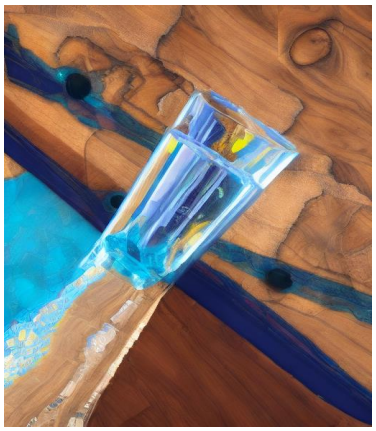
Input



Taming Transformer, Esser *et.al.*



Ours default
(Seems to be interpreted as a bird's eye view of an agricultural field)



Ours “a glass of water”
(Seems unable to eliminate the effects of mistaken recognitions)

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.

[29] S. Kutuzova, O. Krause, D. McCloskey, M. Nielsen, and C. Igel. Multimodal variational autoencoders for semi-supervised learning: In defense of product-of-experts. *arXiv preprint arXiv:2101.07240*, 2021.

[30] K. Lasinger, R. Ranftl, K. Schindler, and V. Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *CoRR*, abs/1907.01341, 2019.

[31] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[32] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[33] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila. Noise2noise: Learning image restoration without clean data, 2018.

[34] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

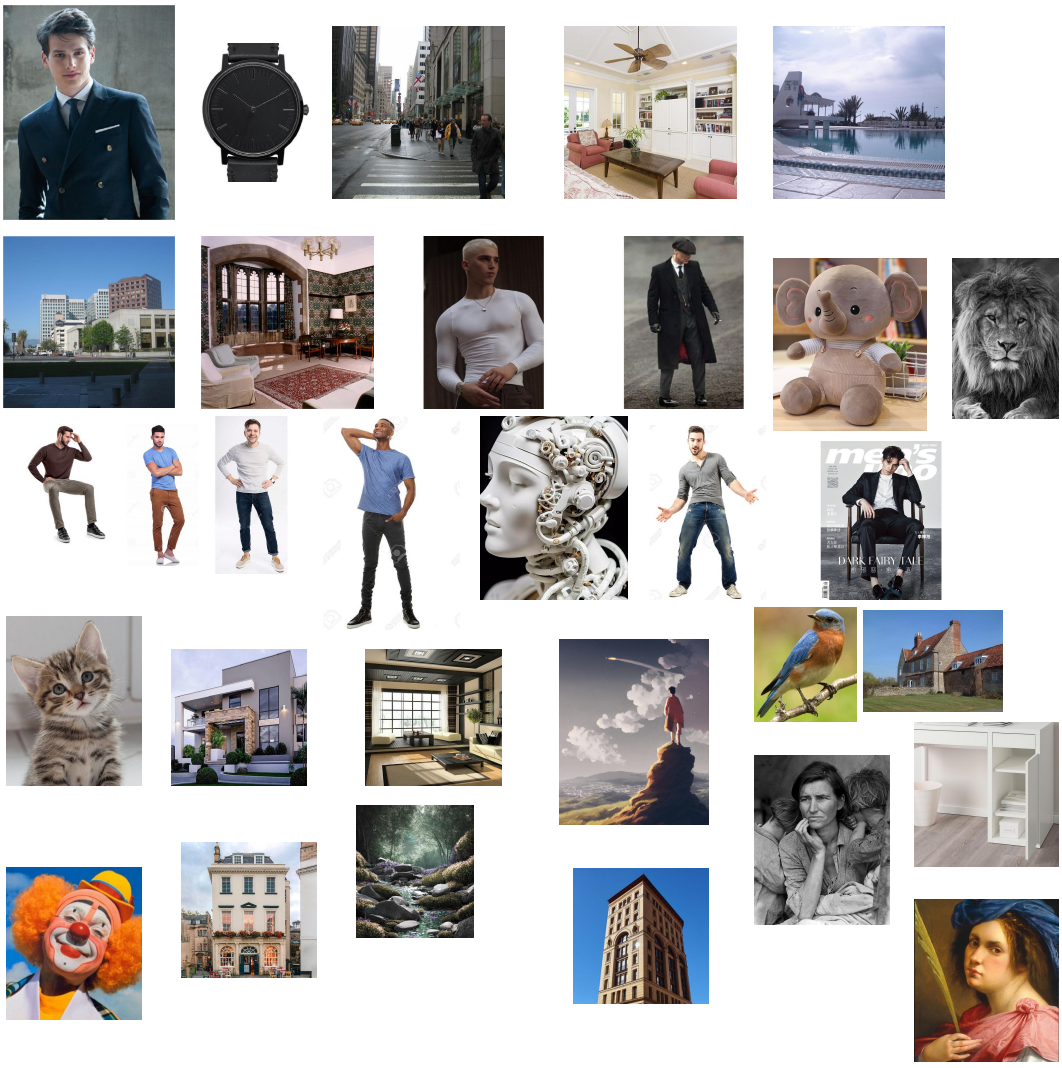


Figure 29: Appendix: all original source images for edge detection, semantic segmentation, pose extraction, *etc*. Note that some images may have copyrights.

[35] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J.-Y. Zhu, and S. Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[36] A. Mercurio. Waifu diffusion, 2022.

[37] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models, 2021.

[38] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[39] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019.

[40] G. Qian, J. Gu, J. S. Ren, C. Dong, F. Zhao, and J. Lin. Trinity of pixel enhancement: a joint solution for demosaicking, denoising and super-resolution. *arXiv preprint arXiv:1905.02538*, 1 (3):4, 2019.

[41] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021.

[42] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[43] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[44] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[45] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In MICCAI (3)*, volume 9351 of* Lecture Notes in Computer Science, pages 234–241. 2015.

[46] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022.

[47] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, Oct. 1986.

[48] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393379.

[49] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

[50] R. San-Roman, E. Nachmani, and L. Wolf. Noise estimation for generative diffusion models. *CoRR*, 2104, 2021.

[51] A. Shamsian, A. Navon, E. Fetaya, and G. Chechik. Personalized federated learning using hypernetworks. In *International Conference on Machine Learning*, pages 9489–9502. PMLR, 2021.

[52] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR*, 1503, 2015.

[53] J. Song, C. Meng, and S. Ermon. *Denoising diffusion implicit models*. In *ICLR*. OpenReview.net, 2021.

[54] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *CoRR*, 2011:13456, 2020.

[55] Stability. stable-diffusion-2-depth, https://huggingface.co/stabilityai/stable-diffusion-2-depth, 2022.

[56] I. Vasiljevic, N. Kolkin, S. Zhang, R. Luo, H. Wang, F. Z. Dai, A. F. Daniele, M. Mostajabi, S. Basart, M. R. Walter, and G. Shakhnarovich. DIODE: A Dense Indoor and Outdoor DEpth Dataset. *CoRR*, abs/1908.00463, 2019.

[57] Y. Vinker, Y. Alaluf, D. Cohen-Or, and A. Shamir. Clipascene: Scene sketching with different types and levels of abstraction, 2022.

[58] A. Voynov, K. Abernan, and D. Cohen-Or. Sketch-guided text-to-image diffusion models. 2022.

[59] T. Wang, T. Zhang, B. Zhang, H. Ouyang, D. Chen, Q. Chen, and F. Wen. Pretraining is all you need for image-to-image translation, 2022.

[60] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018.

[61] X. Xiang, D. Liu, X. Yang, Y. Zhu, and X. Shen. Anime2sketch: A sketch extractor for anime arts with deep networks. `https://github.com/Mukosame/Anime2Sketch`, 2021.

[62] S. Xie and Z. Tu. Holistically-nested edge detection. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1395–1403, 2015.

[63] P. Zhang, B. Zhang, D. Chen, L. Yuan, and F. Wen. Cross-domain correspondence learning for exemplar-based image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5143–5153, 2020.

[64] Z. Zhang, J. Ma, C. Zhou, R. Men, Z. Li, M. Ding, J. Tang, J. Zhou, and H. Yang. M6-ufc: Unifying multi-modal controls for conditional image synthesis. *arXiv preprint arXiv:2105.14211*, 2021.

[65] J. Zhao, F. Schäfer, and A. Anandkumar. Zero initialization: Initializing residual networks with only zeros and ones. *CoRR*, abs/2110.12661, 2021.

[66] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[67] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, 2017.

[68] X. Zhou, B. Zhang, T. Zhang, P. Zhang, J. Bao, D. Chen, Z. Zhang, and F. Wen. Cocosnet v2: Full-resolution correspondence learning for image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11465–11475, 2021.

[69] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. *Advances in neural information processing systems*, 30, 2017.