**ControlNet 1.1 Depth**

Training data: Midas depth (resolution 256/384/512) + Leres Depth (resolution 256/384/512) + Zoe Depth (resolution 256/384/512). Multiple depth map generator at multiple resolution as data augmentation.

Acceptable Preprocessors: Depth\_Midas, Depth\_Leres, Depth\_Zoe. This model is highly robust and can work on real depth map from rendering engines.

Improvement :

1. The training dataset of previous cnet 1.0 has several problems including (1) a small group of greyscale human images are duplicated thousands of times (!!), causing the previous model somewhat likely to generate grayscale human images; (2) some images has low quality, very blurry, or significant JPEG artifacts; (3) a small group of images has wrong paired prompts caused by a mistake in our data processing scripts. The new model fixed all problems of the training dataset and should be more reasonable in many cases.
2. The new depth model is a relatively unbiased model. It is not trained with some specific type of depth by some specific depth estimation method. It is not over-fitted to one preprocessor. This means this model will work better with different depth estimation, different preprocessor resolutions, or even with real depth created by 3D engines.
3. Some reasonable data augmentations are applied to training, like random left-right flipping.
4. The model is resumed from depth 1.0, and it should work well in all cases where depth 1.0 works well. If not, please open an issue with image, and we will take a look at your case. Depth 1.1 works well in many failure cases of depth 1.0

**ControlNet 1.1 Normal**

Training data: [Bae's](https://github.com/baegwangbin/surface_normal_uncertainty) normalmap estimation method.

Acceptable Preprocessors: Normal BAE. This model can accept normal maps from rendering engines as long as the normal map follows [ScanNet's](http://www.scan-net.org/) protocol. That is to say, the color of your normal map should look like [the second column of this image](https://raw.githubusercontent.com/baegwangbin/surface_normal_uncertainty/main/figs/readme_scannet.png).

Note that this method is much more reasonable than the normal-from-midas method in ControlNet 1.0. The previous method will be abandoned.

Improvement:

1. The normal-from-midas method in Normal 1.0 is neither reasonable nor physically correct. That method does not work very well in many images. The normal 1.0 model cannot interpret real normal maps created by rendering engines.
2. This Normal 1.1 is much more reasonable because the preprocessor is trained to estimate normal maps with a relatively correct protocol (NYU-V2's visualization method). This means the Normal 1.1 can interpret real normal maps from rendering engines as long as the colors are correct (blue is front, red is left, green is top).
3. In our test, this model is robust and can achieve similar performance to the depth model. In previous CNET 1.0, the Normal 1.0 is not very frequently used. But this Normal 2.0 is much improved and has potential to be used much more frequently.

**ControlNet 1.1 MLSD**

Training data: M-LSD Lines.

Acceptable Preprocessors: MLSD.

We fixed several problems in previous training datasets. The model is resumed from ControlNet 1.0 and trained with 200 GPU hours of A100 80G.

**Improvements in MLSD 1.1:**

1. The training dataset of previous cnet 1.0 has several problems including (1) a small group of greyscale human images are duplicated thousands of times (!!), causing the previous model somewhat likely to generate grayscale human images; (2) some images has low quality, very blurry, or significant JPEG artifacts; (3) a small group of images has wrong paired prompts caused by a mistake in our data processing scripts. The new model fixed all problems of the training dataset and should be more reasonable in many cases.
2. We enlarged the training dataset by adding 300K more images by using MLSD to find images with more than 16 straight lines in it.
3. Some reasonable data augmentations are applied to training, like random left-right flipping.
4. Resumed from MLSD 1.0 with continued training with 200 GPU hours of A100 80G.

**ControlNet 1.1 Scribble**

Training data: Synthesized scribbles.

Acceptable Preprocessors: Synthesized scribbles (Scribble\_HED, Scribble\_PIDI, etc.) or hand-drawn scribbles.

We fixed several problems in previous training datasets. The model is resumed from ControlNet 1.0 and trained with 200 GPU hours of A100 80G.

Improvements in Scribble 1.1:

1. The training dataset of previous cnet 1.0 has several problems including (1) a small group of greyscale human images are duplicated thousands of times (!!), causing the previous model somewhat likely to generate grayscale human images; (2) some images has low quality, very blurry, or significant JPEG artifacts; (3) a small group of images has wrong paired prompts caused by a mistake in our data processing scripts. The new model fixed all problems of the training dataset and should be more reasonable in many cases.
2. We find out that users sometimes like to draw very thick scribbles. Because of that, we used more aggressive random morphological transforms to synthesize scribbles. This model should work well even when the scribbles are relatively thick (the maximum width of training data is 24-pixel-width scribble in a 512 canvas, but it seems to work well even for a bit wider scribbles; the minimum width is 1 pixel).
3. Resumed from Scribble 1.0, continued with 200 GPU hours of A100 80G.

**ControlNet 1.1 Soft Edge**

Training data: SoftEdge\_PIDI, SoftEdge\_PIDI\_safe, SoftEdge\_HED, SoftEdge\_HED\_safe.

Acceptable Preprocessors: SoftEdge\_PIDI, SoftEdge\_PIDI\_safe, SoftEdge\_HED, SoftEdge\_HED\_safe.

This model is significantly improved compared to previous model. All users should update as soon as possible.

New in ControlNet 1.1: now we added a new type of soft edge called "SoftEdge\_safe". This is motivated by the fact that HED or PIDI tends to hide a corrupted greyscale version of the original image inside the soft estimation, and such hidden patterns can distract ControlNet, leading to bad results. The solution is to use a pre-processing to quantize the edge maps into several levels so that the hidden patterns can be completely removed. The implementation is [in the 78-th line of annotator/util.py](https://github.com/lllyasviel/ControlNet-v1-1-nightly/blob/4c9560ebe7679daac53a0599a11b9b7cd984ac55/annotator/util.py#L78).

The perforamce can be roughly noted as:

Robustness: SoftEdge\_PIDI\_safe > SoftEdge\_HED\_safe >> SoftEdge\_PIDI > SoftEdge\_HED

Maximum result quality: SoftEdge\_HED > SoftEdge\_PIDI > SoftEdge\_HED\_safe > SoftEdge\_PIDI\_safe

Improvements in Soft Edge 1.1:

1. Soft Edge 1.1 was called HED 1.0 in previous ControlNet.
2. The training dataset of previous cnet 1.0 has several problems including (1) a small group of greyscale human images are duplicated thousands of times (!!), causing the previous model somewhat likely to generate grayscale human images; (2) some images has low quality, very blurry, or significant JPEG artifacts; (3) a small group of images has wrong paired prompts caused by a mistake in our data processing scripts. The new model fixed all problems of the training dataset and should be more reasonable in many cases.
3. The Soft Edge 1.1 is significantly (in nealy 100% cases) better than HED 1.0. This is mainly because HED or PIDI estimator tend to hide a corrupted greyscale version of original image inside the soft edge map and the previous model HED 1.0 is over-fitted to restore that hidden corrupted image rather than perform boundary-aware diffusion. The training of Soft Edge 1.1 used 75% "safe" filtering to remove such hidden corrupted greyscale images insider control maps. This makes the Soft Edge 1.1 very robust. In out test, Soft Edge 1.1 is as usable as the depth model and has potential to be more frequently used.

**ControlNet 1.1 Segmentation**

Training data: COCO + ADE20K.

Acceptable Preprocessors: Seg\_OFADE20K (Oneformer ADE20K), Seg\_OFCOCO (Oneformer COCO), Seg\_UFADE20K (Uniformer ADE20K), or manually created masks.

Now the model can receive both type of ADE20K or COCO annotations. We find that recognizing the segmentation protocol is trivial for the ControlNet encoder and training the model of multiple segmentation protocols lead to better performance.

**Improvements in Segmentation 1.1:**

1. COCO protocol is supported. The previous Segmentation 1.0 supports about 150 colors, but Segmentation 1.1 supports another 182 colors from coco.
2. Resumed from Segmentation 1.0. All previous inputs should still work.

**ControlNet 1.1 Openpose**

The model is trained and can accept the following combinations:

* Openpose body
* Openpose hand
* Openpose face
* Openpose body + Openpose hand
* Openpose body + Openpose face
* Openpose hand + Openpose face
* Openpose body + Openpose hand + Openpose face

However, providing all those combinations is too complicated. We recommend to provide the users with only two choices:

* "Openpose" = Openpose body
* "Openpose Full" = Openpose body + Openpose hand + Openpose face

**Improvements in Openpose 1.1**:

The improvement of this model is mainly based on our improved implementation of OpenPose. We carefully reviewed the difference between the pytorch OpenPose and CMU's c++ openpose. Now the processor should be more accurate, especially for hands. The improvement of processor leads to the improvement of Openpose 1.1.

More inputs are supported (hand and face).

The training dataset of previous cnet 1.0 has several problems including (1) a small group of greyscale human images are duplicated thousands of times (!!), causing the previous model somewhat likely to generate grayscale human images; (2) some images has low quality, very blurry, or significant JPEG artifacts; (3) a small group of images has wrong paired prompts caused by a mistake in our data processing scripts. The new model fixed all problems of the training dataset and should be more reasonable in many cases.

**ControlNet 1.1 Inpaint**

Some notices:

1. This inpainting ControlNet is trained with 50% random masks and 50% random optical flow occlusion masks. This means the model can not only support the inpainting application but also work on video optical flow warping. Perhaps we will provide some example in the future (depending on our workloads).
2. We updated the gradio (2023/5/11) so that the standalone gradio codes in main ControlNet repo also do not change unmasked areas. Automatic 1111 users are not influenced.

**ControlNet 1.1 Tile**

Update 2023 April 25: The previously unfinished tile model is finished now. The new name is "control\_v11f1e\_sd15\_tile". The "f1e" means 1st bug fix ("f1"), experimental ("e"). The previous "control\_v11u\_sd15\_tile" is removed. Please update if your model name is "v11u".

The model can be used in many ways. Overall, the model has two behaviors:

* Ignore the details in an image and generate new details.
* Ignore global prompts if local tile semantics and prompts mismatch, and guide diffusion with local context.

Because the model can generate new details and ignore existing image details, we can use this model to remove bad details and add refined details. For example, remove blurring caused by image resizing.

If your image already have good details, you can still use this model to replace image details. Note that Stable Diffusion's I2I can achieve similar effects but this model make it much easier for you to maintain the overall structure and only change details even with denoising strength 1.0 .