Neural Strokes: Stylized Line Drawing of 3D Shapes Supplementary Material

1. Additional comparisons and results

First, we note that our source code and dataset are available on our project website:
www.github.com/DifanLiu/NeuralStrokes

Additional comparisons with Bénard *et al*. [1]. Figure 1 shows the training artist's drawing on the top, and results from Bénard *et al*. [1] in the bottom (zoom-in for details, and compare with our results in Fig. 4, 8, 5 of our main paper). They roughly capture the overall distribution of line properties, without matching the artist's choices well. Moreover, Bénard *et al*. [1] introduces many holes and cannot handle challenging cases, such as varying stroke thickness or large deformation (the rightmost style in Figure 1).

More generalization cases. Figure 2 demonstrates challenging generalization cases: given a training drawing of a shape belonging to one category (e.g., humanoid), we synthesize a drawing for a shape from an entirely different category (e.g., mechanical object) in the same style. Our method still generalizes sufficiently in these challenging cases.

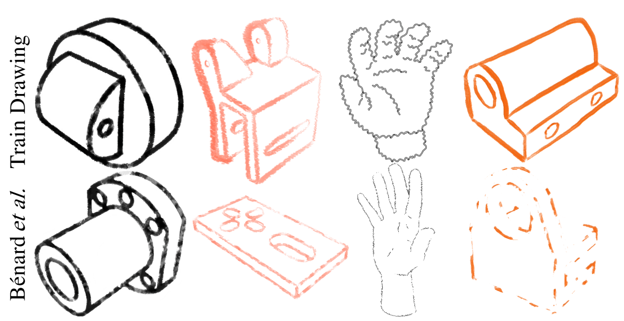

Figure 1: *Top:* artist-drawn training drawings. *Bottom:* results from Bénard *et al*. [1].

Figure 2: *Left to right:* training artist's drawing, test geometric curves, Neural Strokes.

2. Network architecture

We provide here additional details of our network architecture (see also Section 3.2 and 3.3 of our main text).

Surface geometry module. Our surface geometry module uses the architecture shown in Table 1. All convolutional layers are followed by instance normalization [3] and a ReLU nonlinearity. The module contains 4 residual blocks [2], where each residual block contains two 3×3 convolutional layers with the same number of filters for both layers.

Path geometry module. Our path geometry module uses the architecture shown in Table 2. The first two convolutional layers are followed by a ReLU nonlinearity. The last layer has 3 output channels: two for 2D displacement, and one for thickness. For thickness, we use a ReLU activation to guarantee non-negative outputs, while for the 2D real-valued displacement output, we do not use any nonlinearity.

Layer	Activation size
Input	$768 \times 768 \times 9$
Conv2D(7x7, 9→10, stride=1)	$768 \times 768 \times 10$
Conv2D(3x3, 10→20, stride=2)	$384 \times 384 \times 20$
Conv2D(3x3, 20→40, stride=2)	$192 \times 192 \times 40$
4 Residual blocks	$192 \times 192 \times 40$
Conv2D(3x3, 40→40, stride=1/2)	$384 \times 384 \times 40$
Conv2D(3x3, 40→40, stride=1/2)	$768 \times 768 \times 40$
Conv2D(1x1, 40→40, stride=1)	$768 \times 768 \times 40$

Table 1: Architecture of the surface geometry module.

Layer	Activation size
Input	$M_i \times 45$
Conv1D(3x3, 45→40, stride=1)	$M_i \times 40$
Conv1D(3x3, 40→40, stride=1)	$M_i \times 40$
Conv1D(3x3, 40→3, stride=1)	$M_i \times 3$

Table 2: Architecture of the path geometry module.

Stroke texture module. Our stroke texture module uses the architecture shown in Table 3. All convolutional layers are followed by instance normalization [3] and a ReLU nonlinearity except for the last convolutional layer. The last convolutional layer is followed by a sigmoid activation function. The module contains 6 residual blocks [2], where each residual block contains two 3×3 convolutional layers with the same number of filters for both layers.

3. Additional experiments

We experimented with using one SketchPatch model for stroke geometry prediction and another SketchPatch model for stroke texture prediction, as discussed in Section 5 of our main text ("comparison methods" paragraph). Specifically, in the first step, we train a SketchPatch model (called *SketchPatch-geometry*) on the training stroke mask $\hat{\mathbf{I}}_b$ to predict stroke geometry as a grayscale raster image. In the second step, we train another SketchPatch model (called *SketchPatch-texture*) on the training drawing $\hat{\mathbf{I}}$ to generate a stylized line drawing given the output of *SketchPatch-geometry*. The results did not improve compared to *SketchPatch* in terms of our evaluation metrics (see Table 4). Figure 3 shows example output of *SketchPatch-geometry* and *SketchPatch-texture*.

4. Perceptual evaluation

We conducted an Amazon Mechanical Turk perceptual evaluation where we showed participants (a) a stylized

Layer	Activation size
Input	$768 \times 768 \times 9$
Conv2D(7x7, 9→64, stride=1)	$768 \times 768 \times 64$
Conv2D(3x3, 64→128, stride=2)	$384 \times 384 \times 128$
Conv2D(3x3, 128→256, stride=2)	$192 \times 192 \times 256$
6 Residual blocks	$192 \times 192 \times 256$
Conv2D(3x3, 256→128, stride=1/2)	$384 \times 384 \times 128$
Conv2D(3x3, 128→64, stride=1/2)	$768 \times 768 \times 64$
Conv2D(7x7, 64→3, stride=1)	$768 \times 768 \times 3$

Table 3: Architecture of the stroke texture module.

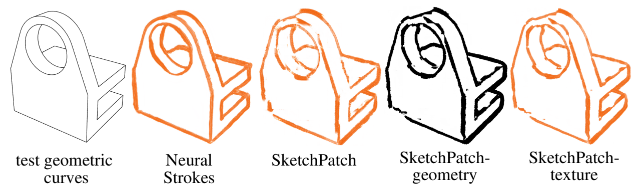

Figure 3: *Left to right:* test geometric curves, Neural Strokes, SketchPatch, SketchPatch-geometry, SketchPatch-texture result.

Method	LPIPS ↓	FID ↓
SketchPatch	0.1104	83.60
SketchPatch-texture	0.1142	86.96
Neural Strokes	**0.0956**	**62.40**

Table 4: Quantitative evaluation of SketchPatch variants.

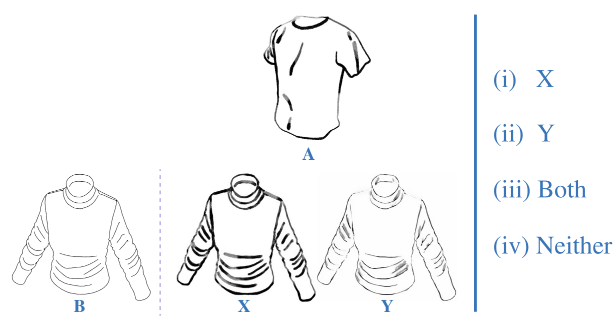

Figure 4: Layout shown to participants of our user study.

artist's drawing for a training shape (Figure 4, A) , (b) test geometric curves (Figure 4, B) , (c) a pair of stylized line drawings of the test shape placed in a randomized left/right position (Figure 4, X and Y): one line drawing was picked from our method, while the other came from *SketchPatch*, *SinCUT*, *NST*, Bénard *et al.* [1], or *Artists* (5 possible comparison cases). We asked participants to select the drawing that best mimicked the style of training drawing A. Participants could pick one of four options: drawing X, drawing Y, "neither of the drawings mimicked the style well", or "both drawings mimicked the style well". The study included the 31 styles from our dataset and each style consists of 3 test shapes. As a result, there were total 93 test cases, each involving the above-mentioned 5 comparisons (465 total comparisons).

Each questionnaire was released via the MTurk platform. It contained 15 unique questions, each asking for one comparison. Then these 15 questions were repeated in the questionnaire in a random order. In these repeated questions, the order of compared line drawings was flipped. If a worker gave more than 5 inconsistent answers for the repeated questions, then the worker was marked as "unreliable". Each participant was allowed to perform the questionnaire only once to ensure participant diversity. A total of 161 participants took part in the study. Among 161 participants, 68 workers were marked as "unreliable". For each of the 465 comparisons, we gathered votes from 3 different "reliable" users. The results are shown in Figure 6 of the main text.

References

[1] Pierre Bénard, Forrester Cole, Michael Kass, Igor Mordatch, James Hegarty, Martin Sebastian Senn, Kurt Fleischer, Davide Pesare, and Katherine Breeden. Stylizing animation by example. *ACM Trans. Graph.*, 32(4), 2013. 1, 2

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 1, 2

[3] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 1, 2