Volume xx (200y), Number z, pp. 1–12

Pencil Drawing Video Rendering Using Convolutional Networks

Dingkun Yan[1], Yun Sheng[2†], Xiaoyang Mao[3],

[1] The School of Computer Science and Technology, East China Normal University, China
[2]The Department of Computer Science, Liverpool John Moores University, U.K.
[3]The Department of Computer Science and Engineering, University of Yamanashi, Japan
[3]The College of Computer Science, Hangzhou Dianzi University, China

Abstract

Traditional pencil drawing rendering algorithms when applied to video may suffer from temporal inconsistency and shower-door effect due to the stochastic noise models employed. This paper attempts to resolve these problems with deep learning. Recently, many research endeavors have demonstrated that feed-forward Convolutional Neural Networks (CNNs) are capable of using a reference image to stylize a whole video sequence while removing the shower-door effect in video style transfer applications. Compared with video style transfer, pencil drawing video is more sensitive to the inconsistency of texture and requires a stronger expression of pencil hatching. Thus, in this paper we develop an approach by combining a latest Line Integral Convolution (LIC) based method, specializing in realistically simulating pencil drawing images, with a new feed-forward CNN that can eliminate the shower-door effect successfully. Taking advantage of optical flow, we adopt a feature-map-level temporal loss function and propose a new framework to avoid the temporal inconsistency between consecutive frames, enhancing the visual impression of pencil strokes and tone. Experimental comparisons with the existing feed-forward CNNs have demonstrated that our method can generate temporally more stable and visually more pleasant pencil drawing video results in a faster manner.

CCS Concepts

• ***Computing methodologies*** → *Artificial intelligence; Non-photorealistic rendering;*

1. Introduction

Pencil drawing is one of the most fundemental art styles for human beings to describe natural scenes and has drawn considerable attention from the community of Non-photorealistic Rendering (NPR). During the past years, researchers in NPR have developed various sophisticated approaches to render pencil drawing images as realistic as possible and achieved many promising results. Among these existing algorithms, Line Integral Convolution (LIC) based methods [MNI01, KSZ18, TNF99, YKM12] filtering a random noise degraded image along a given vector field have been most widely adopted because the LIC generated texture resembles pencil hatching strokes. Meanwhile, according to recent researches, Convolutional Neural Networks (CNNs) have achieved great progress in image and video style transfer. For example, Gatys *et al* [GEB16] pioneered a neural network algorithm to stylize images, but their method relied on a time-consuming optimization process to iteratively reduce the difference of style between the input image and reference image. Johnson *et al* [JAF16] proposed a feed-forward CNN to significantly accelerate the generation process to a real-time speed. For video style transfer, a natural way to generalize the image processing techniques is to directly apply the feed-forward CNN to video transfer frame by frame. However, this strategy inevitably results in temporal inconsistencies and flickering artifacts. To address these issues, Ruder *et al* [RDB16] suggest-

Figure 1: *An example of shower-door effect. The positions of the generated pencil strokes are fixed between the two frames rather than moving with the objects, such as the horses, although the intensity of generated strokes varies according to that of the input frames. Note that for illustration purposes the two images shown are not consecutive frames.*

† e-mail: y.sheng@ljmu.ac.uk

ed a temporal loss function, which uses the optical flow to evaluate the inconsistency of consecutive frames. Their method succeeded in generation of stably video sequences, but at a higher cost of time to transfer a single frame. Inspired by [JAF16, RDB16], Chen *et al* [CLY*17] proposed a recurrent neural network to achieve real-time video style transfer, but their system requires to train two extra networks to compute the optical flow and occlusion mask. Huang *et al* [HWL*17] developed a two-frame synergic training method to maintain the consistency of stylized video sequences, and to train the feed-forward CNN for video style transfer. Gao *et al* [GGZY18] then introduced a feature-map-level temporal loss function in order to render more stable texture. Nevertheless, these neural network methods are inapplicable to pencil drawing video generation. Although Johnson *et al* [JAF16] experimented their networks for pencil drawing generation, their network did not specialize in pencil drawing rendering, leading to disordered hatching results. Unlike style transfer mainly relying on color and shape transformations, pencil drawing video rendering should pay special attention to interframe hatching consistency, which is, however, harder to handle with the existing neural networks of style transfer.

On the other hand, the traditional algorithms [LXJ12, KSZ18] have professional performance in pencil drawing rendering, but suffer from the temporal inconsistency and shower-door effect in pencil drawing video rendering. The temporal inconsistency is caused by the random noise and texture used in pencil hatching synthesis, and the shower-door effect is an illusion that random variation exists on a piece of glass through which the scene is being viewed. Fig 1 shows such an example of shower-door effect, caused by the position fixing of generated strokes, making the resulting pencil drawing frames appear to be covered by a shower door. The shower-door effect can be removed only if the random variation follows the movement of objects. Therefore, a conventional way to solve the shower-door effect is to adopt temporally coherent noise capable of tracking object movement [KP11]. Moreover, Fišer *et al* [FLJ*14] developed a workflow to render temporally coherent hand-colored animation by controlling the level of temporal noise. These methods can achieve notable results, but at the price of higher computational complexity and this kind of noise is unsuitable for generating pencil hatching.

Inspired by [GEB16, JAF16, HWL*17, GGZY18] which use a CNN to capture high-level features, we attempt in this paper to incorporate traditional methods with these deep learning models to generate pencil drawing video. As these neural network algorithms have been demonstrated to be capable of removing the shower-door effect and restrain the temporal inconsistency, we consider the feed-forward CNN an effective implement to resolve these problems occuring in pencil drawing video rendering. In order to produce visually clear and dense enough pencil hatching strokes, we combine a latest LIC-based method [KSZ18] specializing in rendering visually pleasant pencil hatching with neural networks to render pencil drawing video sequences. We propose a new feed-forward CNN by taking advantage of feature-map-level temporal loss to implement temporally consistent pencil hatchings. We experimentally compare our rendering results with those generated by other methods to demonstrate that our system is capable of generating more stable pencil drawing video when using an LIC generated pencil drawing as the reference image. The main contributions of this paper are concluded as follows:

- A new feed-forward CNN is proposed to produce pencil drawing video. With the new network, more low-level features are remained while the frame information goes through the CNN. Skip connections are set to directly convert information from the encoder to the decoder in the stylizing network.
- We justify the direct application of a feed-forward CNN to pencil drawing video rendering for the first time. By combining with the feature-map-level temporal loss, our feed-forward CNN is able to generate temporally stable and visually clear pencil hatching strokes for video sequences.

The rest of this paper is structured as: Section 2 reviews the related work. Section 3 describes the details of our method, including our network architecture and loss function design. Section 4 experimentally demonstrates the effectiveness of our method. Section 5 draws a conclusion.

2. Related Work

As there is no existing work of using deep learning to render pencil drawing video, we instead review the state of the art literature on CNN-based style transfer as well as the extant 2D work on pencil drawing rendering in this section.

2.1. Neural Network Algorithms

Style transfer aims to transfer the style of an input image or video sequence in accordance with that of a reference image or video sequence. The idea of using CNN to stylize images was first developed by Gatys *et al* [GEB16], where style transfer was performed in an optimization manner by running back-propagation with a perceptual loss function defined on high-level features of pre-trained VGG-19 [SZ15]. Though this method achieved impressive stylization results, the optimization process was time-consuming. Thus Johonson *et al* [JAF16] proposed to train a feed-forward CNN with a similiar perceptual loss function defined on VGG-16 to replace this time-consuming process. Their method enabled real-time style transfer on a image basis and achieved notable results. Later Ulyanov *et al* [UVL16] demonstrated that using instance normalization could generate visually more pleasant stylization results.

A straightforward way to generalize image style transfer methods to video is to treat each video frame as an independent image. But without considering temporal consistency, severe flickering artifacts can be observed in resulting stylized video. In order to suppress flickering and enforce temporal consistency, many algorithms have been developed for different video generation tasks [KP11, LWA*12, BTS*15, RDB16, ABMO16]. For example, Ruder *et al* [RDB16] introduced a temporal loss function to constraint the interframe inconsistency. Their loss function consists of optical flow and an occlusion mask defined in [SBK10], and is iteratively optimized for each frame until the loss converges. Their method can achieve satisfactory results but its running speed is rather slow due to its iterative optimization process and on-the-fly computation of the optical flow. Inspired by Johnson's work, Huang *et al* [HWL*17] trained a feed-forward CNN with Ruder's hybrid loss

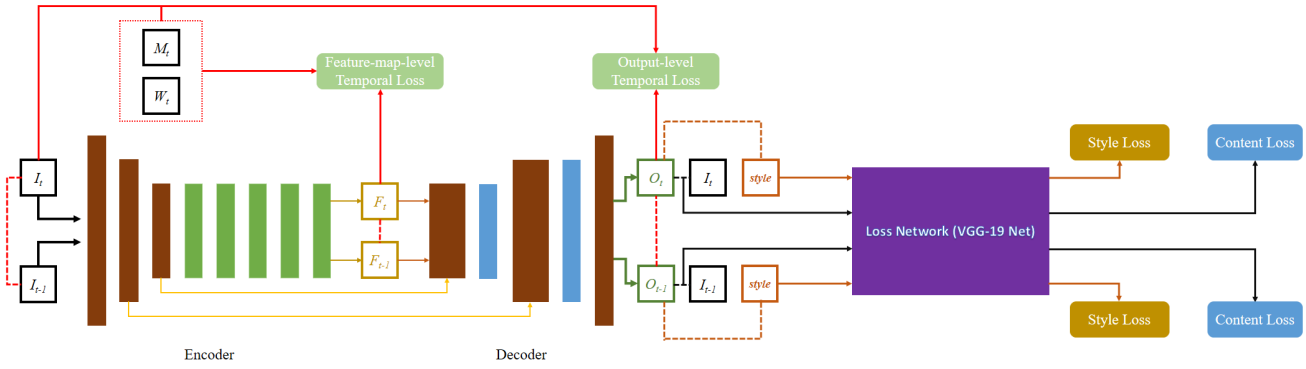


Figure 2: *An overview of our deep learning model, which consists of two networks: a stylizing network (encoder and decoder) and a loss network. In the stylizing network, convolutional blocks, residual blocks, and deconvolutional blocks are colored in brown, green, and blue, respectively. I_t, F_t, O_t, and style represent the input frame, encoded feature map, stylized output, and given reference image, respectively. W_t and M_t denote the optical flow and occlusion mask between frames $t-1$ and t. To compute the temporal loss, a two-frame synergic training mechanism [HWL*17] is adopted. The feature-map-level loss is taken into account for temporal consistency. When applied to computing the feature-map-level temporal loss, the occlusion mask M_t and optical flow W_t will be downscaled .*

function and ground truth optical flow [RDB16]. They demonstrated that temporal consistency and style transfer could be simultaneously learned by the feed-forward CNN and that the on-the-fly computation of optical flow was unnecessary when applying this network to video stylization. Gao *et al* [GGZY18] introduced a feature-map-level temporal loss function to penalize variations of the high-level features of the same objects in consecutive frames, in order to generate more stable video results.

In addition, superiority of reinforcement learning has also been justified in still painting rendering [XHS12, HHZ19, JFB*19]. The above reviewed deep learning methods only require to lower an overall loss. In contrast to them, reinforcement learning methods tend to behave more intelligently. This, unfortunately, hinders their learning methods to branch out to video style transfer. Here, a more sophisticated reward would be required taking temporal coherence into account. As a consequence such methods would need much more computation time.

2.2. Traditional Pencil Drawing Rendering Methods

Pencil drawing rendering has been a booming area in NPR for years and many sophisticated 2D methods, such as LIC-based methods [MNI01, KSZ18] as well as a method to simulate a combination of sketch and tone (CST) of real pencil drawing [LXJ12], etc, have been proposed to automatically render pencil drawing images. Note that there are also some 3D pencil drawing rendering methods [HZ00, GTDS10, SBB17] that will not be reviewed in this paper as our work is 2D-related.

LIC was first developed to visualize a vector field within an image by locally blurring textures along the vector field [CL93]. Since its output highly resembles pencil hatching strokes, LIC was then employed by Takagi *et al* [TNF99] in color pencil drawing rendering. Subsequently, Mao *et al* [MNI01] introduced random binary white noise for LIC to realistically simulate pencil hatching, followed by Yang *et al* [YKM12]. In real pencil drawing artists process their drawings with hatching graduation, in order to achieve the illusion of multi-dimensional forms. To simulate the characteristics of hatch graduation, Kong *et al* [KSZ18] introduced a hybrid noise model for LIC to realistically simulate the progression of hatching in terms of both stroke density and intensity. Moreover, apart from the LIC-based methods, Almeraj *et al* [AWI*09] proposed a system to generate aesthetically pleasant pencil lines through modeling the movement of human arm. Lu *et al* [LXJ12] designed an algorithm to automatically produce realistic pencil drawing by combining sketch and tone. Nevertheless, all of these traditional methods, when used in pencil drawing video generation, will inevitably suffer from the shower-door effect. To resolove the shower-door effect, in this paper we propose a new feed-forward CNN to produce pencil drawing video with rich stroke texture as well as temporal consistency.

3. Our Method

Our training model of pencil drawing video rendering consists of two neural networks as shown in Fig 2: A stylizing network and a loss network. The stylizing network including two modules, an encoder and a decoder, takes one frame as input and produces its corresponding pencil drawing output. The encoder aims to convert an input frame into a high-level feature map, from which the feature-map-temporal loss is derived, while the decoder is devoted to generating a pencil drawing image from the feature map. The loss network, pre-trained for the ImageNet classification task [SZ15] and fixed in our training process, extracts frame features and computes the spatial loss that will be used to penalize the difference of high-level features between the generated output and reference image. A complete training process works as: Two consecutive video frames

I_t and I_{t-1} are first input into the encoder of the stylizing network and extracted as high-level feature maps F_t and F_{t-1}, which will be used to compute the feature-map temporal loss; then the decoder uses these feature maps to generate pencil drawing outputs O_t and O_{t-1}, which will be devoted to calculation of the output-level temporal loss. When computing these temporal losses, optical flow and an occlusion mask will be used to warp F_{t-1} and O_{t-1}. The loss network continues to extract features of the generated pencil drawing image and to compute the spatial loss with these features. The temporal loss is used to evaluate the temporal coherence of generated frames and to enforce the temporal consistency of stylized video. The spatial loss, a weighted sum of content loss and style loss, assesses the style transfer quality in the spatial domain. The content loss defines the difference of high-level contents between an input frame and its corresponding stylized frame, while the style loss measures the similarity between stylistic features of the stylized frame and given reference image. In the inference stage, only the stylizing network is used for generating pencil drawing video.

Table 1: *The stylizing network architecture. Conv denotes the convolutional block (convolutional layer + instance normalization + activation); Res denotes the residual block; Deconv denotes the deconvolutional block (reflection padding + convolutional layer + instance normalization + activation)*

	Layer	Layer Size	Stride	Channel	Activation
Encoder	Conv	3	1	16	ReLU
	Conv	3	2	32	ReLU
	Conv	3	2	48	ReLU
	Res × 5				
Decoder	Conv	3	1	96	ReLU
	Deconv	3	0.5	48	ReLU
	Conv	3	1	64	ReLU
	Deconv	3	0.5	32	ReLU
	Conv	3	1	3	Tanh
Res	Conv	3	1	48	ReLU
	Conv	3	1	48	

3.1. Network Architecture

Our training pipeline consists of a stylizing network and a loss network, as shown in Fig 2. The stylizing network is a feed-forward CNN responsible for transferring the style of input video frames to the pencil drawing style given by the reference image. The architecture of the stylizing network is outlined in Table 1. Inspired by [RFB15, IZZE17, GGZY18], we construct the stylizing network with an encoder and a decoder, and convert the feature map directly from the encoder to decoder through two skip connections. The encoder includes three convolutional blocks and five residual blocks. The three convolutional blocks including one stride-1 convolution and two stride-2 convolutions aim to reduce the resolution of the feature map to one quarter of the input, and to extract high-level features with outputs of the second and third convolutional blocks delivered to the decoder. The feature map extracted by the five residual blocks are then used to compute the feature-map-level temporal loss. Three convolutional blocks and two deconvolutional blocks are alternately set to decode the feature map into a stylized pencil drawing frame. The stylized frame will be devoted to computation of the output-level temporal loss and to evaluation of the temporal consistency. We use a VGG-19 network to extract features of the stylized pencil drawing frame in order to compute the spatial loss. The efficiency of VGG-19 has been demonstrated on image content and style representations [GEB16]. Note that the loss network does not take part in the network training, which means the loss network will not be influenced by the loss function.

Different from the existing network achitectures, our stylizing network has two skip connections between mirrored layers in the encoder and decoder stacks, and is trained with both the feature-map-level and output-level temporal loss functions. These exclusive features make our network capable of generating temporally more coherent pencil drawing video with higher quality of hatching texture using a smaller number of channels. In other words, our networks can generate visually more pleasant pencil drawing video sequences in a faster manner.

3.2. Loss Functions

A fundamental constraint to enable the stylizing network to generate coherently stylized video is the hybrid loss, which simultaneously takes both temporal inconsistency and stylization performance into account. The hybrid loss $\mathcal{L}_{hybrid}$ consists of the spatial loss $\mathcal{L}_{spatial}$ and the temporal loss $\mathcal{L}_{temporal}$, and is formulated as:

$$\mathcal{L}_{hybrid}(t,t-1) = \sum_{i\in\{t,t-1\}} \mathcal{L}_{spatial}(i) + \mathcal{L}_{temporal}(t,t-1) \quad (1)$$

3.2.1. The Spatial Loss

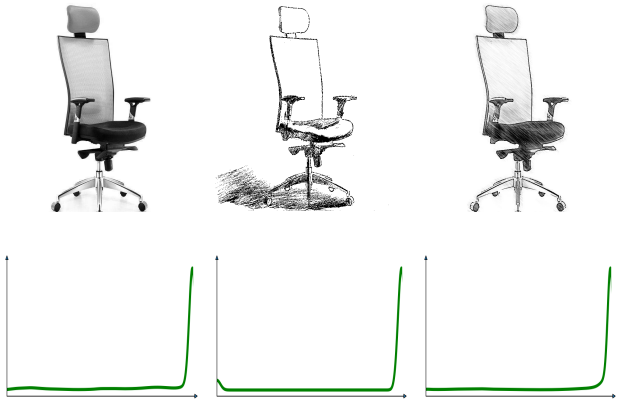
The spatial loss is computed separately for two consecutive frames $\mathbf{x}^i$ and $\mathbf{x}^{i-1}$, and their stylized results $\hat{\mathbf{x}}^i$ and $\hat{\mathbf{x}}^{i-1}$, and is tailored to evaluate the stylization performance in the spatial domain. We adopt the perceptual loss defined by Gatys *et al* [GEB16] as the spatial loss to evaluate the stylization performance, which is expressed as:

$$\begin{aligned}\mathcal{L}_{spatial}(t) =& \lambda_c \sum_l \mathcal{L}^l_{content}(\hat{\mathbf{x}}^t, \mathbf{x}^t) + \lambda_s \sum_l \mathcal{L}^l_{style}(\hat{\mathbf{x}}^t, \mathbf{s}) + \\ & \lambda_{tv}\mathcal{L}_{tv}(\hat{\mathbf{x}}^t)\end{aligned} \quad (2)$$

where l indexes a feature extraction layer in the loss network, and λ_c, λ_s and λ_{tv} are hyperparameters set to 10^5, 10^{11} and 100, respectively. The content loss $\mathcal{L}^l_{content}$ is a mean squared error at layer l between the features of the original and the stylized frames, and formulated as:

$$\mathcal{L}^l_{content}(\hat{\mathbf{x}}^t, \mathbf{x}^t) = \frac{1}{C_l H_l W_l} \|\phi_l(\mathbf{x}^t) - \phi_l(\hat{\mathbf{x}}^t)\|_2^2 \quad (3)$$

where $\phi_l(\mathbf{x}^t)$ represents the feature map of layer l in the loss network, whose dimension is $C_l \times H_l \times W_l$ with C_l, H_l, and W_l denoting the channel number, height, and weight, respectively. Johnson *et al* [JAF16] discovered that if a higher layer were used to compute the content loss, the image content and overall spatial structure would be preserved. Thus we choose *ReLU2_2* to calculate the



(a) Photograph (b) Real (c) LIC result

Figure 3: *Comparison of grayscale value distributions between a real pencil drawing and LIC generated result [KSZ18]. The first row shows the images and the second row shows their corresponding grayscale value distributions.*

content loss in order to generate pencil drawing video with visually more clear strokes. The content loss forces the stylized frame $\hat{\mathbf{x}}^t$ to be perceptually similar to the original frame in the feature level rather than in the output level, so that the rendered hatching texture can stick to moving objects when the stylizing network is applied to video generation.

In order to penalize the style deviation between the output frame and the given reference image, the perceptual style loss proposed by Gatys *et al* [GEB16] is employed. To evaluate the stylization performance we need to define the Gram matrix to capture the style information of features. Elements of the Gram matrix are given as

$$G_l^{\phi}(\hat{\mathbf{x}}^t)_{c,c'} = \frac{1}{C_l H_l W_l} \sum_{h=1}^{H_l} \sum_{w=1}^{W_l} \phi_l(\hat{\mathbf{x}}^t)_{h,w,c} \phi_l(\hat{\mathbf{x}}^t)_{h,w,c'} \tag{4}$$

where $G_l^{\phi}(\hat{\mathbf{x}}^t)_{c,c'}$ is interpreted as the (c,c')-th element in $G_l^{\phi}(\hat{\mathbf{x}}^t)$; $\phi_l(\hat{\mathbf{x}}^t)_{h,w,c}$ and $\phi_l(\hat{\mathbf{x}}^t)_{h,w,c'}$ respectively index the (h,w,c)-th and (h,w,c')-th elements in $\phi_l(\hat{\mathbf{x}}^t)$. Since $G_l^{\phi}(\hat{\mathbf{x}}^t)$ is an inner product of the feature maps of channel c and c', and has been demonstrated to be capable of expressing the activation degree between different features [GEB16,JAF16], it can capture the style information of the input image, such as texture, color, and shape. The style loss $\mathcal{L}_{style}$ can then be defined as a Frobenius norm of the difference between Gram matrices of the features extracted from the stylized output and the given reference image,

$$\mathcal{L}_{style}^l(\hat{\mathbf{x}}^t, \mathbf{s}) = \|G_l^{\phi}(\hat{\mathbf{x}}^t) - G_l^{\phi}(\mathbf{s})\|_F^2 \tag{5}$$

where $\mathbf{s}$ represents the reference image. To ensure that the style information in different scales are captured, features in a set of layers are needed. In this paper, we choose *ReLU1_2, ReLU2_2, ReLU3_2, ReLU4_2* to calculate the style loss.

Futhermore, we adopt the total variation regularization to smooth our pencil drawing video, which is formulated as:

$$\mathcal{L}_{tv}(\hat{\mathbf{x}}^t) = \sum_{i,j} \|(\hat{\mathbf{x}}_{i,j+1}^t - \hat{\mathbf{x}}_{i,j}^t)^2 + (\hat{\mathbf{x}}_{i+1,j}^t - \hat{\mathbf{x}}_{i,j}^t)^2\|^{\frac{\eta}{2}} \tag{6}$$

where $\hat{\mathbf{x}}_{i,j}^t$ denotes the (i,j)-th pixel in stylized frame $\hat{\mathbf{x}}^t$ and η is set as 1 according to [MV15].

3.2.2. The Temporal Loss

The temporal loss is tailored to constrain the temporal inconsistency during the rendering of pencil drawing video. In this paper, we adopt a multi-layer temporal loss function which utilizes the optical flow to calculate the feature difference between consecutive frames $\mathbf{x}^t$ and $\mathbf{x}^{t-1}$ on both feature-map-level and output-level. The temporal loss $\mathcal{L}_{temporal}$ is written as:

$$\begin{aligned}\mathcal{L}_{temporal}(t,t-1) =& \lambda_f \mathcal{L}_{temp,f}(\mathbf{x}^t, \mathbf{x}^{t-1}) + \\ & \lambda_o \mathcal{L}_{temp,o}(\hat{\mathbf{x}}^t, \hat{\mathbf{x}}^{t-1}, \mathbf{x}^t, \mathbf{x}^{t-1})\end{aligned} \tag{7}$$

where hyperparameters λ_f and λ_o are set to 1×10^8 and 2×10^3, respectively, while $\mathcal{L}_{temp,f}$ and $\mathcal{L}_{temp,o}$ represent the feature-map level and output-level temporal loss, respectively. The feature-map-level temporal loss $\mathcal{L}_{temp,f}$ can be expressed as:

$$\mathcal{L}_{temp,f}(\mathbf{x}^t, \mathbf{x}^{t-1}) = \frac{1}{D_p} \sum_{k=1}^{D_p} m_k^t (\phi_p(\mathbf{x}_k^t) - \omega_{t-1}^t(\phi_p(\mathbf{x}_k^{t-1}))^2 \tag{8}$$

where $\phi_p(\mathbf{x}^t)$ denotes the feature map of layer p in the stylizing network, and $D_p = H_p \times W_p$ with H_p and W_p being the height and weight of the feature map. m^t indicates the ground-truth confident coefficients between frame t and frame $t-1$ with 0 representing either in occluded regions or on motion boundries, and 1 elsewhere. ω_{t-1}^t is a function that utilizes a pre-computed and downscaled optical flow between $\mathbf{x}^t$ and $\mathbf{x}^{t-1}$ to warp the output of layer s at time $t-1$ to time t. The formula of the output-level temporal loss $\mathcal{L}_{temp,o}$ is given as:

$$\begin{aligned}&\mathcal{L}_{temp,o}(\hat{\mathbf{x}}^t, \hat{\mathbf{x}}^{t-1}, \mathbf{x}^t, \mathbf{x}^{t-1}) = \\ &\frac{1}{D_o} \sum_{k=1}^{D_o} \sum_{c \in \{r,g,b\}} m_k^t (\hat{\mathbf{x}}_k^t - \omega_{t-1}^t(\hat{\mathbf{x}}_k^{t-1}))_c - (\mathbf{x}_k^t - \omega_{t-1}^t(\mathbf{x}_k^{t-1}))_Y)^2\end{aligned} \tag{9}$$

where $D_o = H_o \times W_o, H_o$ and W_o denote the height and weight of stylized frames, respectively. c and Y symbolize respectively the RGB channels and the luminance in the XYZ color space. Traditional output-level temporal loss disregards the fluctuation of luminance [RDB16, HWL*17], resulting in severe flickering and poor visual performance in pencil drawing video rendering. To address this problem we adopt the definition of temporal loss function in Equation 7 to maintain consistency of the luminance [GGZY18].

4. Experiments

In this section, we will experimentally introduce the implementation and training details of our neural networks. To justify the use

Figure 4: *Outputs rendered by our networks but trained with different reference images. The top row shows the original inputs while the leftmost column shows the references, corresponding in turn to three typical real pencil drawings in different styles, a drawing by the CST method [LXJ12], and a pencil drawing generated by the LIC-based method [KSZ18]. Each following column shows the pencil drawing results of inputting a topmost image using variant style references.*

Table 2: *Temporal errors and FPS (Frames Per Second, standing for the number of frames transfered each second) of different methods on the Cat, Train, Tower and Farmer video sequences.*

	Ours	Ours(no connection)	Huang *et al* [HWL*17]	ReCoNet [GGZY18]	Johnson *et al* [JAF16]
Cat	0.0769670	0.0783420	0.0881978	0.0767470	0.1053774
Train	0.0877978	0.0869236	0.0910891	0.0898393	0.0989200
Tower	0.0605936	0.0575110	0.0710667	0.0558016	0.0746633
Farmer	0.1273785	0.1335144	0.1352576	0.1212932	0.1394699
FPS	94.85	107.34	106.48	34.30	48.65

Figure 5: *Avoidance of the shower-door effect. The left and right columns respectively show three frames of the rendered video sequences using our method and the LIC-based method [KSZ18]*

of the LIC-based method, we will compare the outputs of our feed-forward CNN using different pencil drawings as the reference image. To justify our proposed network in pencil drawing video rendering, we also carry out several experiments to compare our results with those generated by other neural network architectures as well as traditional methods.

4.1. Implementation Details

Our neural networks are coded with Pytorch 1.0, trained and tested on an NVIDIA GTX 1080 Ti GPU. The training dataset is made up of video frames, optical flows and occlusion masks. We download 30 video sequences (about 21,000 frames) from Videvo [Tea19], of which themes cover natural scenes, humans, and animals. 20 sequences are used as a training dataset and the rest are used to validate our method. All the frames are resized to 512×512. We use Farneback's method [Far02] to compute the optical flow and follow the method suggested in [RDB16] to compute the occlusion masks. We train our feed-forward stylizing network with a batch size of 2 for 42,000 iterations, roughly four epoches over the training dataset. Two neighbouring frames are input into the proposed feed-forward network each step, and then their stylized outputs are used to calculate the temporal loss corporately and the spatial loss separately. We utilize the Adam optimization [KB15] to control our training with $\alpha = 0.5$ and $\beta = 0.999$, and start with a learning rate of 1×10^{-3}.

4.2. Comparison of Using Different Reference Images

Directly application of real pencil drawing as the reference image to the training of our neural network is a natural thinking to rendering pencil drawing video. Nonetheless, it is insufficient for the stylizing network to produce satisfactory pencil drawing video due to the randomness introduced by real pencil drawing. Though pencil drawing is an artistic mirror of what an artist visualizes, it may be casually expressed. For example, Fig 3 shows that the real pencil drawing differs from a chair photograph not only in color but also in grayscale value distribution. Instead the LIC-based method [KSZ18], selected as the latest in its kind representing the state of the art technology in pencil drawing rendering, automatically renders a pencil drawing image in line with not only the color but also the intensity distribution of the original photograph.

In order to demonstrate that by combining the LIC-based method [KSZ18] we can obtain better pencil drawing rendering results, we carry out an experiment to compare the outputs rendered by our networks but trained with different reference images, as shown in Fig 4. It can be seen that the pencil drawing rendering results trained with the real pencil drawing images fail to generate visually clear hatching strokes. Using the reference generated by the LIC-based method to train our network can achieve pencil drawing frames with the most distinct hatching strokes. The real pencil drawings fail to achieve satisfactory results because it is too difficult for the convolutional layers to capture artistic features of the real pencil drawings. More results of pencil drawing video sequences generated by our networks can be found in our supplementary material.

4.3. Comparison to Other Methods

In order to demonstrate the effectiveness of our method in removing the shower-door effect, we compare the 61st, 73rd, and 85th frames

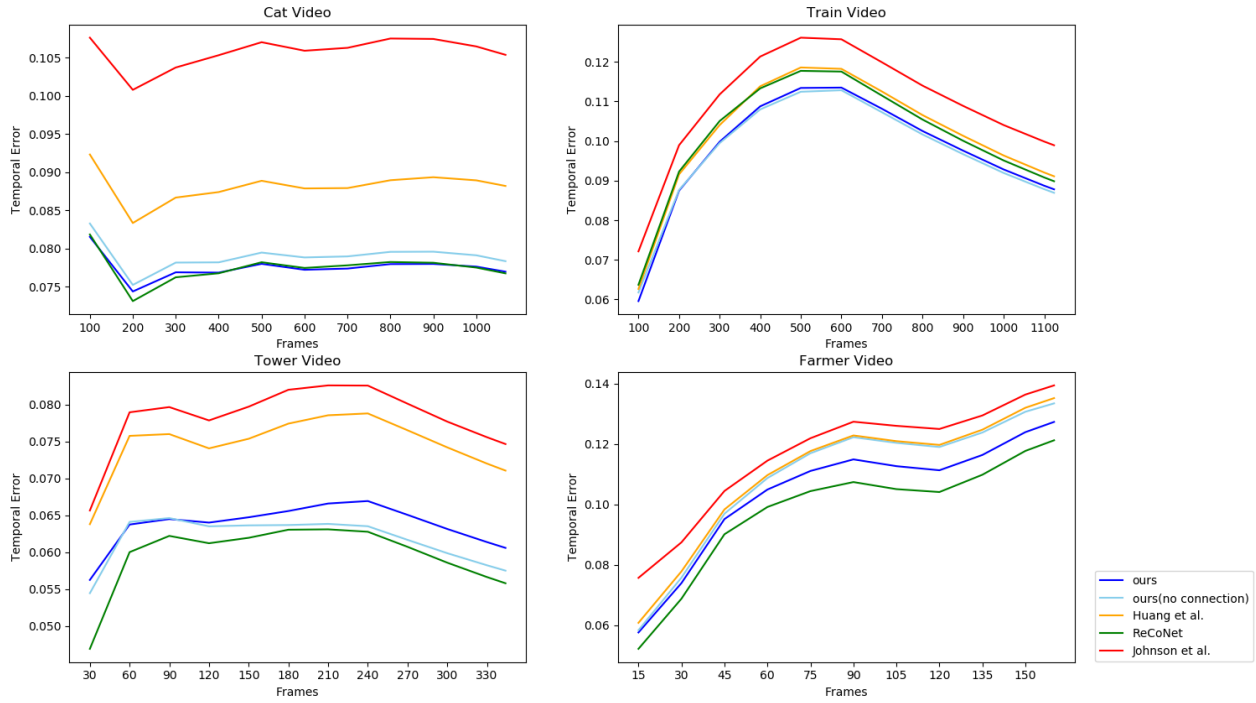


Figure 6: *Temporal error variations of using different networks on the Cat, Train, Tower, Farmer sequences. We compute the temporal errors at regular intervals on these video sequences for a better visualization of temporal consistency.*

of a pencil drawing video sequence rendered by both our method and the LIC-based method [KSZ18] in Fig 5. It can be seen that the hatching strokes rendered by our method move as the deer raises its head. In other words, hatching strokes rendered by our network can move in line with the motion of objects in the scene, leading to removal of the shower-door effect casued by the position fixing of hatching strokes rendered by the conventional method.

In addition, we use a temporal error e_{temp} defined in Equation 10 to evaluate the temporal consistency of pencil drawing video sequences,

$$e_{temp} = \sqrt{\frac{1}{(T-1)\times D}\sum_{t=2}^{T}\sum_{k=1}^{D} m_k^t(\hat{\mathbf{x}}_k^t - \omega_{t-1}^t(\hat{\mathbf{x}}_k^{t-1}))} \qquad (10)$$

where T denotes the total number of frames, and $D = H \times W$ with H and W representing the height and weight of the test frames. A lower value of e_{temp} corresponds to a rendered video sequence with a higher temporal consistency, *vice versa*. In order to demonstrate our network has better performance in keeping video coherence, we compare our temporal loss with those reported in [HWL*17, GGZY18, JAF16]. To testify its effectiveness, we also compare the temporal loss of our feed-forward CNN with and without direct information connection between the encoder and decoder in our network. Temporal errors of using different neural networks are compared in Table 2 and Fig 6. As can be seen, the temporal errors of video sequences rendered by our method are close to those of ReCoNet but lower than those by Huang *et al* and Johnson *et al*, showing our results are more stable than those of Huang *et al* and Johnson *et al*. On the other hand, though ReCoNet reaches a satisfactory temporal consistency, it has the lowest FPS value among these network architectures while ours with no connection can even beat that of Huang *et al*. We consider that the difference mainly results from the temporal loss function. Johnson *et al* designed their network for image style transfer where their loss function involving only the spatial loss did not take temporal consistency into account. Huang *et al* constructed their temporal loss function without the luminance constraint in output-level and penalization in feature-map-level, both of which are taken into consideration by ReCoNet and ours. Thus results of Huang *et al* are more stable than those of Johnson *et al* but much poorer than those of ours and ReCoNet, as shown in Fig 6 and Table 2 that the temporal errors of their results are lower than Johnson *et al* but higher than ours and ReCoNet.

We show some pencil drawing video frame samples in Fig 7 to subjectively compare the rendering performance of different networks. Since Johnson *et al*'s network is trained without the temporal loss function, meaning that their network places more emphasis on the stylization performance than anything else, their results appear to have the richest texture details and the strongest visual effect, but comes with temporal inconsistency which can be clearly seen in the background of Fig 7(b), as some bright spots on the grass as well as the highlighted deer face and rabbit ears flicker along the consecutive frames. ReCoNet reduces the depth of net-

Figure 7: *Pencil drawing rendering results generated with different network architecutres.(a) Consecutive input frames; (b) results generated by Johnson* et al*'s network; (c) results by ReCoNet; (d) results by Huang* et al*'s network; (e) ours.*

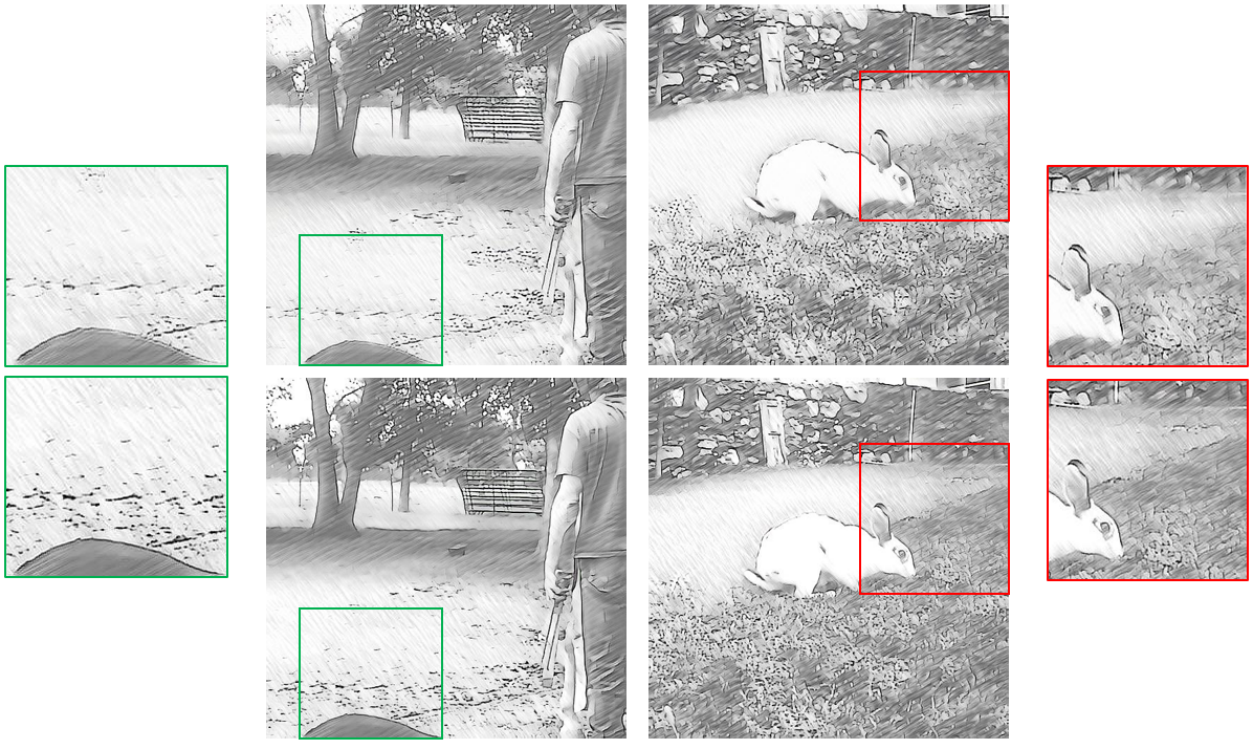
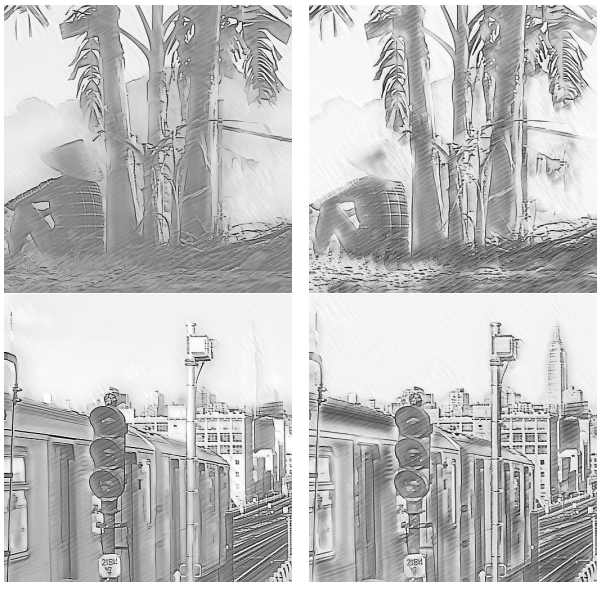


Figure 8: *A comparison of pencil drawing frames rendered by our networks. The first and second rows correspond to the results of our feed-forward CNN without and with the skip connetions between the encoder and the decoder.*

(a) 4-Res structure (b) 5-Res structure

Figure 9: *Pencil drawing rendering results of different network architectures. The two images in turn show the outputs of Res-4 and Res-5 stylizing networks. While hatching strokes in the former are visually more evident in a single frame, it will appear unpleasant due to the massive presence of strokes in video sequences.*

work to four residual blocks but increases the number of neurons compared to Johnson *et al*'s model, leading to a slower rendering speed and visually unsatisfactory pencil drawing results. Hatching strokes generated by ReCoNet appear with lack of graduality, as can be seen in the grass around the rabbit in Fig 7(c). ReCoNet also fails to generate edges as clear as Johnson *et al* and ours, as highlighted in the close-ups. Huang *et al*'s model only reduces the



(a) Default (b) with a lower value of λ_f

Figure 10: *Pencil drawing frames rendered by our network trained with different λ_f settings. λ_f is set to 1×10^8 and 1×10^7 for the left and right columns, respectively.*

Table 3: *User study results. We generate pencil drawing sequences of the Cat, Rabbit and Deer videos using each method and aggregate votes on the three questions for all the three video sequences. 'Same' means that a voter thinks there is no difference between two compared results.*

	ReCoNet	Ours	Same	Huang *et al.*	Ours	Same
Q1	30	67	23	21	89	10
Q2	28	89	10	13	105	2
Q3	30	79	11	16	100	4

number of neurons to achieve a faster rendering speed, and does not adopt both the feature-map-level temporal loss and skip connection. As a result, their network cannot generate visually clear object edges and hatching strokes, as highlighted in the close-ups where the boundary of the deer face and the rabbit ears are missing. Though the temporal error of our method is close to that of ReCoNet, hatching strokes rendered by our mehtod tremble less in video sequences than those by ReCoNet, which can neither be evaluated by the temporal error, nor be visualized in terms of images, but can be observed in our supplementary video.

Note that in our network the temporal loss function aims to penalize the temporal inconsistency of video sequences. In order to suppress the temporal inconsistency, the temporal loss attempts to make the rendered frames as bright as possible because the brighter the rendered frames, the smaller the temporal inconsistency. This explains the reason that our pencil drawing rendering results appear relatively brighter, especially in areas whose inputs are bright.

To further compare the overall performance of our method with others, we carry out a user study in this section. As the method proposed by Johnson *et al* was not dedicated to video, results of which in pencil drawing video rendering have also shown obvious weaknesses in terms of visual effect in our previous experiments, comparison with this method is excluded in this study. We choose three videos downloaded from videvo.net [Tea19] to generate pencil drawing videos and invite 40 people to answer three questions after watching. (Q1) Which result has less flickering; (Q2) which result has visually more pleasant hatching; (Q3) which result do you prefer orverall. Note that to facilitate the study we set up comparisons by pairing every result of ours with its counterpart generated by each of the competitors, and then counting the total votes. As shown in Table 3, the study demonstrates that our mehtod has superiority against the competitors.

4.4. Influence of Network Architecture

The alteration of network architecture plays an important role in pencil drawing video rendering. Since we adapt the feed-forward CNN by directly converting low-level features from the encoder to decoder through the skip connections, we take further experiments to testify the influence of skip connections on pencil hatching strokes rendering. As can be discovered from both Table 2 and Fig 6, the skip connection with two extra convolutional blocks makes the rendering process slightly slower, but makes no much difference to the temporal consistency. However, Fig 8 presents a direct comparison of pencil drawing frames, where the hatching strokes rendered by our feed-forward CNN with the skip connections present higher hatching graduality and clearer objects edges, as highlighted in the close-ups.

In order to better understand the influence of the network depth on hatching generation, we also train another feed-forward CNN with only four residual blocks, that is one block less than the one we previously used. The rendering comparison of two network architectures is shown in Fig 9. We discover that by decreasing the depth of the network, hatching strokes turn to be denser but much fuzzier, and the video results tend to have a higher temporal error.

Meanwhile, we find that the temporal loss affects the rendering performance as well. Unlike other art styles which rely on color and shape for artistic expression, pencil drawing is a style that stresses artistic expression with lines and shade, making it more sensitive to an adjustment of hyperparameters. Pencil drawing frames rendered by our network trained with different λ_f values (Equation 7) are shown in Fig 10. Though the right images show visually more clear pencil hatching strokes with more evident outlines, their temporal errors for the *Farmer* and *Train* sequences are 0.1365856 and 0.1006114, both greater than 0.1273785 and 0.877978, produced with the default λ_f value.

5. Conluding Remarks

In this paper, we propose a new feed-forward CNN to cater for pencil drawing video rendering. Our network is trained with the latest temporal loss function and constructed with two additional skip connections as well as convolutional layers. Video sequences rendered by our method are significantly more stable than those by [JAF16, HWL*17], and have lower temporal errors. Although temporal errors introduced by our method are close to those by ReCoNet [GTL*18], a significant reduction of hatching trembling can be observed in the pencil drawing video rendering results with a remarkable acceleration of rendering speed. Experiments have demonstrated that our method can render visually more clear and pleasant strokes than those existing neural network algorithms. Though these competitors were not specifically designed for pencil drawing video rendering, we hope the comparisons between our method and these methods can provide some useful information for readers. Moreover, we have also explored the influence of alteration of network architecture on pencil drawing rendering, including the variations of the skip connections, the depth of our feed-forward CNN, and the λ_f value. Note that although our method can render more stable pencil drawing video than the existing deep learning models, it cannot avoid the trembling of hatchings between neighbouring frames to some extent.

References

[ABMO16] ANDERSON A. G., BERG C. P., MOSSING D. P., OLSHAUSEN B. A.: DeepMovie: Using Optical Flow and Deep Neural Networks to Stylize Movies. *arXiv:1605.08153* (2016). 2

[AWI*09] ALMERAJ Z., WYVILL B., ISENBERG T., GOOCH A. A., GUY R.: Automatically mimicking unique hand-drawn pencil lines. *Computers & Graphics 33*, 4 (2009), 496–508. 3

[BTS*15] BONNEEL N., TOMPKIN J., SUNKAVALLI K., SUN D., PARIS S., PFISTER H.: Blind video temporal consistency. *ACM Trans. Graph. 34*, 6 (2015), 196:1–196:9. 2

[CL93] CABRAL B., LEEDOM L. C.: Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH* (1993), pp. 263–270. 3

[CLY*17] CHEN D., LIAO J., YUAN L., YU N., HUA G.: Coherent online video style transfer. In *Proceedings of IEEE International Conference on Computer Vision, ICCV* (2017), pp. 1114–1123. 2

[Far02] FARNEBÄCK G.: *Polynomial expansion for orientation and motion estimation*. PhD thesis, Linköping University, Sweden, 2002. 7

[FLJ*14] FIŠER J., LUKÁČ M., JAMRIŠKA O., ČADÍK M., GINGOLD Y., ASENTE P., SÝKORA D.: Color Me Noisy: Example-based rendering of hand-colored animations with temporal noise control. *Computer Graphics Forum 33*, 4 (2014), 1–10. 2

[GEB16] GATYS L. A., ECKER A. S., BETHGE M.: Image style transfer using convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2016), pp. 2414–2423. 1, 2, 4, 5

[GGZY18] GAO C., GU D., ZHANG F., YU Y.: ReCoNet: Real-time coherent video style transfer network. In *Proceedings of 14th Asian Conference on Computer Vision, ACCV* (2018), pp. 637–653. 2, 3, 4, 5, 7, 8

[GTDS10] GRABLI S., TURQUIN E., DURAND F., SILLION F. X.: Programmable rendering of line drawing from 3D scenes. *ACM Trans. Graph. 29*, 2 (2010), 18:1–18:20. 3

[GTL*18] GAO C., TANG M., LIANG X., SU Z., ZOU C.: Pencilart: A chromatic penciling style generation framework. *Comput. Graph. Forum 37*, 6 (2018), 395–409. 11

[HHZ19] HUANG Z., HENG W., ZHOU S.: Learning to Paint with Model-based Deep Reinforcement Learning. *arXiv:1903.04411* (2019). 3

[HWL*17] HUANG H., WANG H., LUO W., MA L., JIANG W., ZHU X., LI Z., LIU W.: Real-time neural style transfer for videos. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2017), pp. 7044–7052. 2, 3, 5, 7, 8, 11

[HZ00] HERTZMANN A., ZORIN D.: Illustrating smooth surfaces. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH* (2000), pp. 517–526. 3

[IZZE17] ISOLA P., ZHU J., ZHOU T., EFROS A. A.: Image-to-image translation with conditional adversarial networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2017), pp. 5967–5976. 4

[JAF16] JOHNSON J., ALAHI A., FEI-FEI L.: Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of 14th European Conference on Computer Vision, ECCV* (2016), pp. 694–711. 1, 2, 4, 5, 7, 8, 11

[JFB*19] JIA B., FANG C., BRANDT J., KIM B., MANOCHA D.: PaintBot: A Reinforcement Learning Approach for Natural Media Painting. *arXiv:1904.02201* (2019). 3

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *Proceedings of 3rd International Conference on Learning Representations, ICLR* (2015). 7

[KP11] KASS M., PESARE D.: Coherent noise for non-photorealistic rendering. *ACM Trans. Graph. 30*, 4 (2011), 30:1–30:6. 2

[KSZ18] KONG Q., SHENG Y., ZHANG G.: Hybrid noise for LIC-based pencil hatching simulation. In *Proceedings of the IEEE International Conference on Multimedia and Expo* (2018). 1, 2, 3, 5, 6, 7, 8

[LWA*12] LANG M., WANG O., AYDIN T. O., SMOLIC A., GROSS M. H.: Practical temporal consistency for image-based graphics applications. *ACM Trans. Graph. 31*, 4 (2012), 34:1–34:8. 2

[LXJ12] LU C., XU L., JIA J.: Combining sketch and tone for pencil drawing production. In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering, NPAR'12* (2012), pp. 65–73. 2, 3, 6

[MNI01] MAO X., NAGASAKA Y., IMAMIYA A.: Automatic generation of pencil drawing from 2D images using line integral convolution. In *Proceedings of CAD/Graphics* (2001), pp. 240–248. 1, 3

[MV15] MAHENDRAN A., VEDALDI A.: Understanding deep image representations by inverting them. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2015), pp. 5188–5196. 5

[RDB16] RUDER M., DOSOVITSKIY A., BROX T.: Artistic style transfer for videos. In *Proceedings of 38th German Conference on Pattern Recognition, GCPR* (2016), pp. 26–36. 1, 2, 3, 5, 7

[RFB15] RONNEBERGER O., FISCHER P., BROX T.: U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of Medical Image Computing and Computer-Assisted Intervention - MICCAI* (2015), pp. 234–241. 4

[SBB17] SUAREZ J., BELHADJ F., BOYER V.: Real-time 3D rendering with hatching. *The Visual Computer 33*, 10 (2017), 1319–1334. 3

[SBK10] SUNDARAM N., BROX T., KEUTZER K.: Dense point trajectories by GPU-accelerated large displacement optical flow. In *Proceedings of European Conference on Computer Vision - ECCV* (2010), pp. 438–451. 2

[SZ15] SIMONYAN K., ZISSERMAN A.: Very deep convolutional networks for large-scale image recognition. In *Proceedings of 3rd International Conference on Learning Representations, ICLR* (2015). 2, 3

[Tea19] TEAM V.: Videvo: Free stock video footage. https://www.videvo.net/, 2019. 7, 11

[TNF99] TAKAGI S., NAKAJIMA M., FUJISHIRO I.: Volumetric modeling of colored pencil drawing. In *Proceedings of Pacific Conference on Computer Graphics and Applications, PG* (1999), pp. 250–258. 1, 3

[UVL16] ULYANOV D., VEDALDI A., LEMPITSKY V.: Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv:1607.08022* (2016). 2

[XHS12] XIE N., HACHIYA H., SUGIYAMA M.: Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting. In *Proceedings of the 29th International Conference on Machine Learning, ICML* (2012). 3

[YKM12] YANG H., KWON Y., MIN K.: A stylized approach for pencil drawing from photographs. *Comput. Graph. Forum 31*, 4 (2012), 1471–1480. 1, 3