Deep Style Transfer for Line Drawings

Xueting Liu, Wenliang Wu, Huisi Wu*, Zhenkun Wen
College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, 518060, China
hswu@szu.edu.cn

Abstract

Line drawings are frequently used to illustrate ideas and concepts in digital documents and presentations. To compose a line drawing, it is common for users to retrieve multiple line drawings from the Internet and combine them as one image. However, different line drawings may have different line styles and are visually inconsistent when put together. In order that the line drawings can have consistent looks, in this paper, we make the first attempt to perform style transfer for line drawings. The key of our design lies in the fact that centerline plays a very important role in preserving line topology and extracting style features. With this finding, we propose to formulate the style transfer problem as a centerline stylization problem and solve it via a novel style-guided image-to-image translation network. Results and statistics show that our method significantly outperforms the existing methods both visually and quantitatively.

Introduction

Line drawings are frequently used to illustrate ideas and concepts in digital documents and presentations. Though there exist a large number of line drawings on the Internet, it is still quite common for users to combine multiple line drawings into one image based on their needs. However, different line drawings may have different line styles both globally and locally. The global line style includes the range of the line-width and how the line-width changes spatially, e.g. many line drawings use thicker lines for outlines and thinner lines for inner structures. The local line style refers to how the line-width changes within each line based on local properties, such as curvature, distance to end points, etc. For example, in Fig. 1(a), the wolf, the bird, and the background have clearly different line styles. (Garces et al. 2014) proposed a style similarity metric which helps to find clip arts with similar styles. But the proposed metric only considers the average line-width. The search space is also limited and uncontrollable to users. In order that users can pick any line drawing they like, it is desired that all user-picked line drawings can be transferred to the same style, as shown in Fig. 1(b).

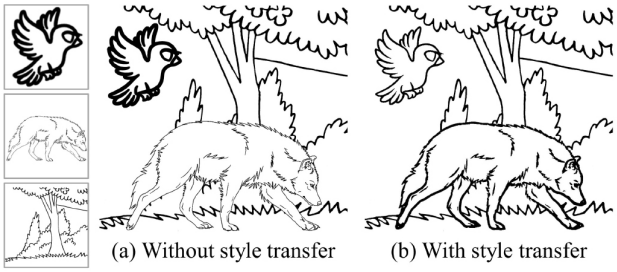


Figure 1: The styles of multiple line drawings may be different in terms of global and local line-width. (a) Without style transfer, the styles of the line drawings are visual inconsistent when composed together. (b) With style transfer, the styles of the line drawings become visually consistent.

To transfer the style of an input image based on a reference image, the existing style transfer methods, either exemplar-based (Hertzmann et al. 2001; Shih et al. 2013, 2014; Frigo et al. 2016) or learning-based (Gatys, Ecker, and Bethge 2016; Johnson, Alahi, and Fei-Fei 2016; Li et al. 2017; Huang and Belongie 2017), require the output image to have similar content with input and similar style with reference by constraining on content similarity and style similarity respectively. However, the similarity constraint is loose, so the output image usually fails to preserve the content (topology of lines) of the input (Fig. 2(c)&(d)). Another potential solution is to first disentangle the content from the input and the style from the reference respectively, and then combine the disentangled content and style to generate the output image. While various style-content disentanglement methods have been proposed (Huang et al. 2018; Gonzalez-Garcia, van de Weijer, and Bengio 2018; Yang et al. 2019; Yu et al. 2019; Lee et al. 2019), they generally require paired data, i.e. images of the same content but different styles, for training, which is difficult to acquire for line drawings. The style-based image generation method (Abdal, Qin, and Wonka 2019) may also be adopted to synthesize the output based on an input image and a style vector. But this method require the output images to have similar semantics, e.g. human faces, cats, dogs, which is not applicable in our case where line drawings may contain arbitrary content (Fig. 2(e)).

There are three major challenges in the line style transfer

*Corresponding author. Email: hswu@szu.edu.cn.

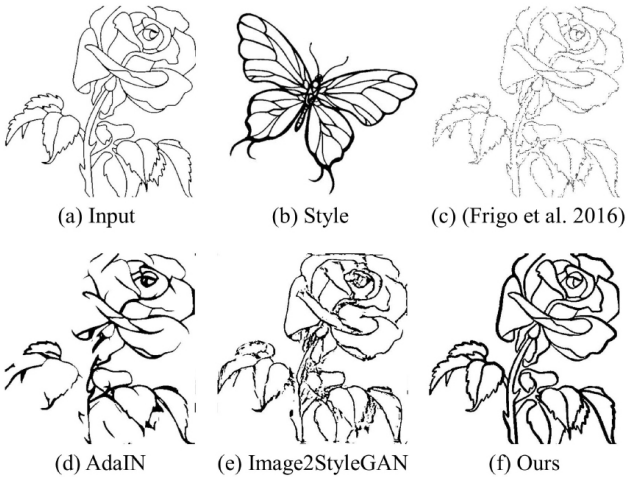


Figure 2: Comparisons with existing methods. Given (a) as input and (b) as style/reference, the existing style transfer methods (c)&(d) fail to preserve the content of the input. The style-based image generation method (e) also fails to generate a satisfying result in both content and style. In comparison, our result (d) preserves content of input and mimics the style of reference.

application. Firstly, the content of the line drawing is precise, slightly modifying the content may demolish the smoothness and/or connectivity of the lines. Secondly, the style of the line drawing is both local and global, we need to capture both the local style and the global style as described above. Thirdly, the style of each line drawing is quite unique, so supervised methods that need a large number of paired data are generally inapplicable. To preserve the topology of the lines during style transfer, we propose to first extract the centerline (a proximate of the line topology) of the input line drawing, and then formulate the style transfer problem as a centerline stylization problem. With this novel formulation, we are able to preserve the content of the input with high fidelity, and at the same time avoid the problem of lacking paired data for line drawings with same content but different styles. Moreover, by identifying the centerline as content, we are able to better disentangle the style from the content of the style image (Fig. 2(f)).

Our system takes an input image and a style image as input, and outputs an image which has the same topology as the input image and a similar style as the style image. We first extract the centerline of the input image and feed it to a centerline stylization network. The centerline stylization network encodes the centerline image into image features and fuses with the style code extracted from the style image. Then the style-fused image features are decoded to an output line drawing. We further employ a discriminator to make the output line drawing look more realistic. With such network designs, we are able to formulate this problem as an image-to-image translation where the input is a centerline image and the output is the original line drawing. To train this network, we further prepare a large number of line drawings and their corresponding centerline images as the training dataset.

Extensive experiments are performed on line drawings of different content and styles. Convincing results are obtained. Our contributions can be summarized as follows:

- We identify the importance of centerline in the line style transfer problem for the purposes of preserving line topology and extracting style features.
- By identifying the centerline as content, we novelly formulate the line style transfer problem as a centerline stylization problem and solve it via a learning-based method.

Related Works

Image Style Transfer Image style transfer can be divided into two categories, traditional style transfer and learning-based style transfer. The traditional style transfer methods generally synthesize the output images by constructing the content of the input image based on the exemplars from the style image based on paired data (Hertzmann et al. 2001) or unpaired data (Shih et al. 2013, 2014; Frigo et al. 2016). While the traditional exemplar-based methods may generate good results when the local characteristics of two styles are very different from each other, they generally fail to identify the style difference between two line drawings since different line styles still exhibit similar local characteristics (i.e. white background and black lines).

Recently, the development of convolutional neural networks (CNNs) has significantly improved the results of style transfer. To translate between two specific domains of styles, supervised image-to-image translation methods have been proposed, such as photo-to-art (Kotovenko et al. 2019) and photo-to-sketch (Yi et al. 2019). For unspecific styles, methods have been proposed to ensure content and style similarities via a content loss and a style loss (Gatys, Ecker, and Bethge 2016; Johnson, Alahi, and Fei-Fei 2016), but these methods usually suffer from limited styles. To be applicable to more flexible styles, (Huang and Belongie 2017) proposed to combine content and style via an adaptive instance normalization layer (AdaIN), and made significant improvement in style transfer. Later, methods have been proposed to further improve the visual quality (Li et al. 2017; Sanakoyeu et al. 2018; Zhang et al. 2019; Song et al. 2019; Wang et al. 2020) or computational efficiency (Li et al. 2018; Gao et al. 2020). However, all the above networks adopted a content loss to ensure the similarity of content between the input and the output, which is unable to preserve the line topology with high fidelity. In comparison, we directly take the centerline of the input as content formulate the style transfer problem as a centerline stylization problem, which can well preserve the line topology with high fidelity.

Text Style Transfer Style transfer is also widely explored in text/fonts (Yang et al. 2017, 2019; Azadi et al. 2018). (Yang et al. 2017) proposed a traditional statistics-based method to transfer the style of the input text image based on stylized patches. But this method requires patch searching and is extremely time-consuming. (Yang et al. 2019) proposed a CNN-based style transfer network, TET-GAN, which first decomposes the input and reference text into g-

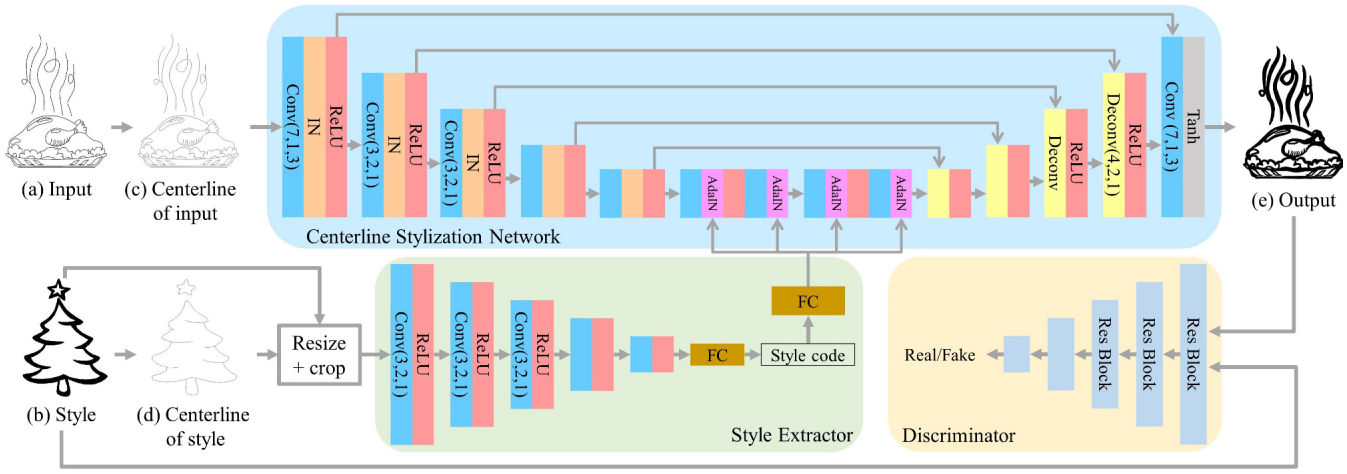


Figure 3: System overview. Our system formulates style transfer as an image-to-image translation where the centerline of the input is fed to a centerline stylization network and fused with the style code extracted from the style image via a style extractor. The fused features will be decoded to an output line image which is further justified by a discriminator.

lyph feature and effect feature, and then recombines the glyph and the effect to form a new font. (Azadi et al. 2018) further proposed a multi-content generative adversarial network (GAN) for few-shot style transfer. (Li et al. 2020) proposed a text style transfer method that transfers the style based on k-shot adaptive instance normalization. However, all the above methods require paired data or domain-specific data for training, which is difficult to acquire for line drawings. Directly adopting the trained models to our line drawing application is unsatisfying both qualitatively and quantitatively since text (especially computer font) and line drawings are still quite different in style. Interestingly, we find that hand-written text exhibit similar style features with line drawings, so we can directly apply our method to transfer the styles of hand-written text.

Style-Content Disentanglement To perform style transfer, the style-content disentanglement methods can also be adopted, which propose to disentangle the style and content for an input image by extracting the shared feature from an image pair with the same content but different styles (Liu, Breuel, and Kautz 2017; Huang et al. 2018; Gonzalez-Garcia, van de Weijer, and Bengio 2018; Lee et al. 2018). (Yu et al. 2019) extended the above methods in manipulating different parts of the latent representations for multi-modal and multi-domain translations. However, in order to successfully disentangle style and content, a large number of paired or domain-specific data are needed, which is generally not available for line drawings. Although that we may manage to create some paired data for training, the result is still unsatisfying due to the incapability to fully disentangle content and style (will be presented in the result section).

Method

The key of our method is to identify the centerline of the input line drawing as the content of the image so that we can formulate the style transfer problem as an image-to-image translation problem with style guidance.

Network Design

Our system consists of a centerline stylization network, a style extractor, and a discriminator, as illustrated in Fig. 3. The input of our system are two images, an input image (Fig. 3(a)) and a style image (Fig. 3(b)). The centerline stylization network takes the centerline of the input line drawing (Fig. 3(c)) as input and stylizes this centerline image based on a style code. Here, the centerline of the input image is obtained using the built-in skeleton extraction algorithm in Matlab (Haralick and Shapiro 1992)). The style extractor takes the style image (Fig. 3(b)) and its centerline counterpart (Fig. 3(d)) as input and outputs a style code, which is further used in the centerline stylization network. The discriminator is adopted to make the output image look more realistic.

Centerline Stylization Network The centerline stylization network takes a centerline image as input and stylizes this centerline image based on a style code. Our centerline stylization network is designed based on U-net (Ronneberger, Fischer, and Brox 2015), which has an encoding subnetwork, a style-content fusion subnetwork, and a decoding subnetwork. The encoding subnetwork extracts high-level image features from the input centerline image using five downscaling blocks. The style-content fusion subnetwork combines the extracted image features with the style code via two residual AdaIN blocks (Huang and Belongie 2017). Finally, the decoding subnetwork decodes the style-fused image features into a stylized line drawing using five upscaling blocks. Note that the network is fully convolutional, so it can be used for input of any resolution. Here, we adopt U-net for the encoding and decoding subnetworks instead of VGG (Simonyan and Zisserman 2015), as in (Huang and Belongie 2017), mainly for two reasons. Firstly, U-net enables a larger receptive field, which can better capture the

global style of the input drawing. Secondly, U-net employs the skipping layers which allow better gradient propagation for higher-resolution layers so as to generate accurate and robust output image for high-frequency components.

Style Extractor The style extractor takes a 512×512 style image and its centerline counterpart as input and outputs a 1×512 style code. It consists of five downscaling blocks and one fully-connected layer. Note that instance normalization is not employed in the downscaling blocks because mean and variance of the style code need to be used in the AdaIN layers. While the network is simple, we emphasize two key designs of our style extractor. Firstly, to extract the style features, we propose to feed not only the style image, but also its centerline counterpart to the network so that the style extractor can better disentangle the style from the content knowing what the content is. Secondly, the resolution of the input style image is fixed in order to acquire a fixed-length style code. Given a style image of any resolution, we need to first resample a 512×512 image to feed to the network. To do so, there are two commonly adopted operators, resizing and cropping. However, neither resizing nor cropping can well capture the style of the style image. The cropping operator can only capture the local style, while the resizing operator can only capture the global style. To capture both global and local styles, we propose to adopt both the resizing and cropping operators, and feed both the resized image and the cropped patch to the network.

Discriminator We further employ a patch-based discriminator (Isola et al. 2017) to make the output image look more realistic. Our discriminator consists of five downscaling residual blocks. During training, real-world line drawings are fed to the discriminator as positive cases, while the output images are fed to the discriminator as negative cases.

Loss Function The loss function $\mathcal{L}$ of our network consists of two loss terms, the reconstruction loss $\mathcal{L}_{rec}$ and the adversarial loss $\mathcal{L}_{adv}$:

$$\mathcal{L} = \mathcal{L}_{rec} + \mathcal{L}_{adv} \tag{1}$$

The reconstruction loss ensures that the output image is similar with the ground-truth image. To measure the dissimilarity between two line drawings, we adopt both the perceptual loss and pixel-wise MSE loss as

$$\mathcal{L}_{rec} = \sum \|\varphi(\hat{y}) - \varphi(y)\|_2^2 + \lambda_1 \|\hat{y} - y\|_2^2 \tag{2}$$

Here, $\hat{y}$ is the output line drawing, y is the ground-truth, $\|\cdot\|_2$ is the L_2 norm operator, and $\varphi(\cdot)$ is the perceptual feature extraction as defined in (Johnson, Alahi, and Fei-Fei 2016). λ_1 is the weighting factor and set to 5 in all our experiments.

The adversarial loss ensures that the output line drawing looks like a real line drawing. We adopt the adversarial loss proposed by (Miyato et al. 2018) with gradient penalty regularization (Mescheder, Geiger, and Nowozin 2018) to improve stability, and is defined as

$$\begin{aligned}\mathcal{L}_{adv} =& E_y[\min(0, -D(y) - 1)] + E_{\hat{y}}[\min(0, D(\hat{y}) - 1)] \\ &+ \lambda_2 E_y[\|\nabla D(y)\|^2]\end{aligned} \tag{3}$$

where D is the discriminator, $\min(\cdot)$ is the minimum operator, and $E[\cdot]$ is the expectation operator. λ_2 is a weighting factor and set to 10 in all our experiments.

Training

To train our networks, we collect 4500 line drawings of various content and styles from the Internet. We randomly select 4,300 from them as the training dataset, and the rest 200 are used as the test set. During network training, we use the same image for both input and style. Note that, unlike the existing methods which cannot be trained with the same input and style, we are able to use the same image for input and style because the centerline is extracted as content. We use Adam optimizer (Kingma and Ba 2015) to train our networks. All networks are jointly trained. The learning rate is initially set to $1e^{-4}$ and gradually decreases to $2e^{-6}$. The optimization converges in about 80 epochs.

Results and Discussions

We compare our method with the state-of-the-art style transfer, image generation, and style-content disentanglement methods. For style transfer methods, we compare with one traditional method, Frigo (Frigo et al. 2016), and three CNN-based methods, Gatys (Gatys, Ecker, and Bethge 2016), WCT (Li et al. 2017), and AdaIN (Huang and Belongie 2017). For image generation, we compare with Image2StyleGAN (abbreviated as I2SGAN in Fig. 4 and Table 1) (Abdal, Qin, and Wonka 2019) trained on our line drawing dataset. For style-content disentanglement methods, we compare with CCD (Gonzalez-Garcia, van de Weijer, and Bengio 2018), TET (Yang et al. 2019), and DMIT (Yu et al. 2019). Since the style-content disentanglement methods require paired data for training, we train these networks by adopting line drawings and their centerline counterparts as the paired data, similar with our method.

Visual Comparisons

We first visually compare our results and the results generated by the existing methods. Fig. 4 shows five examples where the input and style images are of different content and line styles. The traditional style transfer method Frigo fails to generate smooth lines since line topology is not enforced with the exemplar-based approach (Fig. 4(c)). The CNN-based style transfer methods performs better than the traditional method in synthesizing smoother lines, but they still generate noisy output since the content loss cannot ensure the output to have the same line topology with input (Fig. 4(d)-(f)). The image generation method fails to synthesize visually pleasant line drawings since the content of line drawings is arbitrary and hard to summarize (Fig. 4(g)). The style-content disentanglement methods have the best performance where the content of the input are mostly well preserved. But they fail to well disentangle the styles and content from the style images with the implicit style-content disentanglement approach (Fig. 4(h)-(j)). In sharp comparison, our method not only preserves the content of the input image, but successfully transfers the line styles of the style images to the output images (Fig. 4(k)).

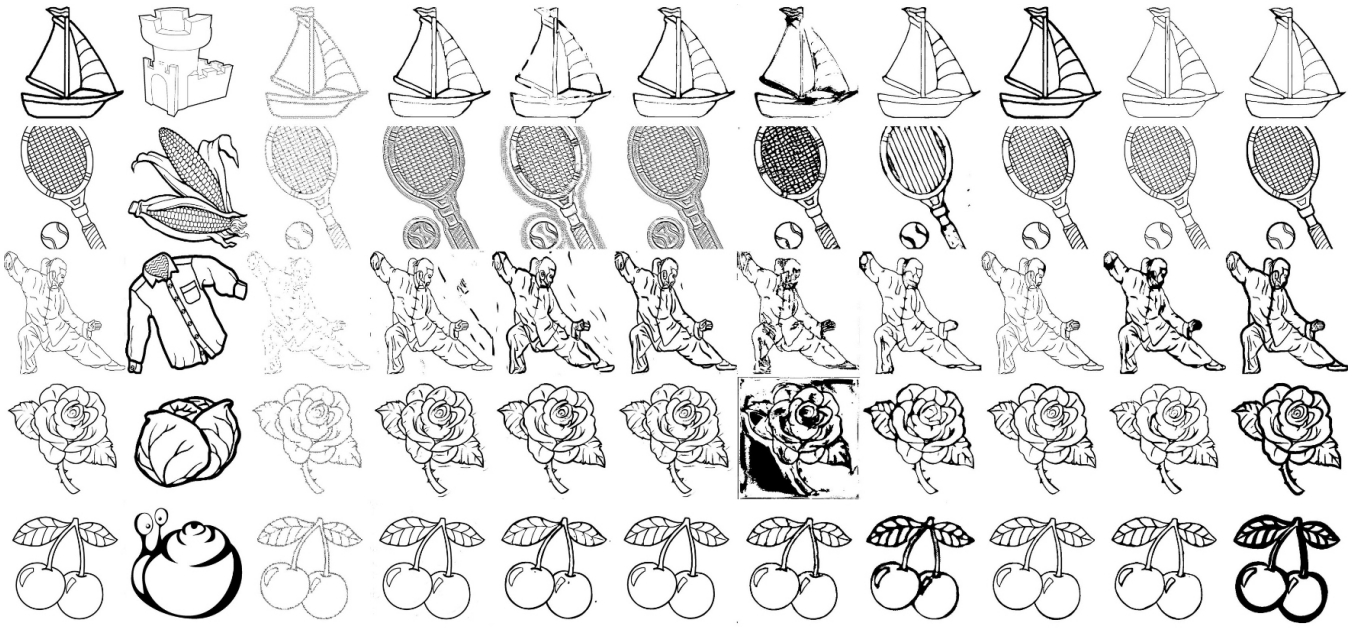


Figure 4: Comparisons of results generated by our method and all competitors.

Method	Self-reconstruction			Half-reconstruction		
	IoU	PSNR	SSIM	IoU	PSNR	SSIM
Gatys	0.41	12.41	0.65	0.39	11.97	0.64
WCT	0.28	10.20	0.65	0.45	12.14	0.75
AdaIN	0.47	12.33	0.76	0.26	10.29	0.66
I2SGAN	0.65	16.29	0.78	0.23	9.37	0.50
CDD	0.57	13.76	0.79	0.58	18.09	0.79
TET	0.66	15.33	0.84	0.66	15.29	0.84
DMIT	0.55	13.70	0.80	0.55	13.37	0.79
Ours	**0.77**	**16.96**	**0.87**	**0.77**	**16.07**	**0.85**

Table 1: Statistics of quantitative evaluation.

Quantitative Evaluation

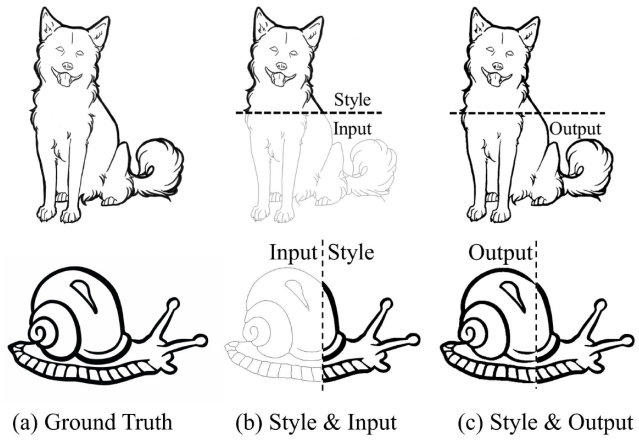
We further compare our method with the existing methods via two quantitative experiments. The first experiment is a self-reconstruction experiment. For each testing line drawing, we take itself as style and its centerline counterpart as input, with the objective to reconstruct the original line drawing. We measure the similarity between the reconstructed line drawing and the input line drawing via three commonly used metrics: Intersection over Union (IOU), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index (SSIM) (Wang et al. 2004). Here, the IoU metric is extremely useful in our case since line drawings are naturally black-and-white images where the overlapping of black pixels indicates the similarity of two line drawings. The statistics are shown under "Self-reconstruction" in Table 1. Our method outperforms all competitors in all metrics.

The second experiment is a half-image reconstruction experiment. For each testing line drawing, we first divide it into two halves (either top-and-bottom or left-and-right based on image ratio), as shown in Fig. 5. Then we take half of the image as style and the centerline of the other half as input, with the objective to reconstruct the original half image for the centerline half. Through this way, the style image and the input image have completely different contents, but their styles are naturally the same because they originally belong to the same line drawing. We also adopt IOU, PSNR, and SSIM to measure the similarity between output half image and ground-truth half image. The statistics are shown under "Half-reconstruction" in Table 1. Our method also outperforms all competitors in all metrics. One interesting thing is that, the performance of our method in "Half-reconstruction" is only slightly lower than our performance in 'Self-reconstruction". This shows that our method successfully extracts the style features from the style images and applies to the input content.

Ablation Study

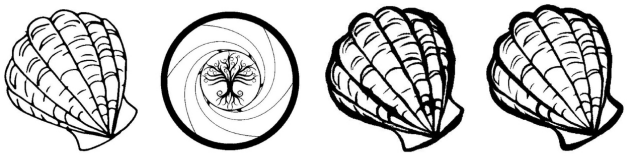
To validate the effectiveness of our network design, we conduct ablation studies for each key design of our system both visually and quantitatively. The quantitative experiment is the same with the self-reconstruction experiment above.

Ablation on Centerline Stylization Network While our centerline stylization network is based on U-net, the networks of the existing style transfer methods are usually based on VGG. So we first study the performance of U-net and VGG respectively, while keeping all the rest unchanged. A visual example is shown in Fig. 6 where the style image adopts thicker lines for outline and thinner lines for in-



(a) Ground Truth (b) Style & Input (c) Style & Output

Figure 5: Half-reconstruction experiment. Given a line drawing in (a), we take half of it as style and the centerline of the other half as input in (b), and output a half image. The output half is combined with the style half in (c).



(a) Input (b) Style (c) VGG-based (d) Ours

Figure 6: Comparisons on adopting different network structures, i.e. VGG and U-net (ours), for the centerline stylization network.

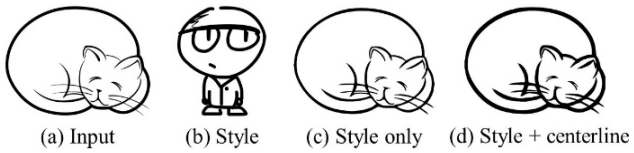
ner structures. While the VGG network fails to capture this global style, our network successfully transfers this style to the output image. The quantitative experiment also shows that our U-net structure outperforms the VGG structure in all three metrics since U-net can better capture global styles and high-frequency image details, as presented in Table 2.

Ablation on Input of Style Extractor Then we measure the effectiveness of our input design for the style extractor. Fig. 7 presents a visual comparison of feeding and not feeding the centerline to the style extractor. Without the centerline input, the style extractor fails to capture the line-width of the style image. In sharp comparison, by feeding both the style image and its centerline counterpart to the style extractor, the line-width of the style image is well preserved. The quantitative experiment also shows that the performance of not feeding the centerline is consistently lower than the performance of feeding the centerline, as shown in Table 2.

Fig. 8 further presents the visual differences of adopting different operators to resample the style image. With the cropping operator, the local line-width is captured, but the global style is not preserved. With the resizing operator, the global style is captured, but the local line-width are not preserved. With both resized and cropped input, both the global style and local line-width of the style image are well

Ablation Experiments	IoU	PSNR	SSIM
Ablation study on centerline stylization network:			
VGG-based network	0.72	16.07	0.83
Ablation study on input of style extractor:			
Without centerline	0.75	16.13	0.87
With centerline (resize only)	0.76	16.63	0.86
With centerline (crop only)	0.76	16.72	0.87
With centerline (resize + 3 crops)	**0.77**	**16.96**	**0.87**
Ablation study on discriminator:			
Without discriminator	0.76	16.78	0.87
Ablation study on centerline extraction:			
Thinning (Zhang and Suen 1984)	0.76	16.78	0.87
Ours	**0.77**	**16.96**	**0.87**

Table 2: Statistics of ablation study.



(a) Input (b) Style (c) Style only (d) Style + centerline

Figure 7: Comparisons on feeding and not feeding the centerline image to the style extractor.

preserved. While one may suggest feeding more cropped patches since one patch might be biased, we find that there's no significant improvement with more cropped patches. The quantitative experiment also shows that either feeding the resized input or the cropped input is lower than feeding both, and feeding more cropped patches does not further improve the performance.

Ablation on Discriminator We also study the effectiveness of the discriminator. Quantitatively, the system without discriminator shows slightly lower performance than the one with the discriminator, as shown in Table 2.

Ablation on Centerline Extraction Since the centerline contributes a significant part to our system, we also study how our system performs with different centerline extraction methods. We adopt another classic thinning method (Zhang and Suen 1984) to extract the centerline while keeping the rest unchanged. Statistics show that our system will not be much affected with different centerline extraction methods. The skeleton extraction method used in our paper performs slightly better than the thinning method. We think that this may because that the skeleton extraction method extracts more detailed structures, which provide better hints when synthesizing the final output.

Timing Statistics

We tested the traditional style transfer method Frigo on a PC with Intel Core i7-9700 3.0GHz CPU and 16GB memory,

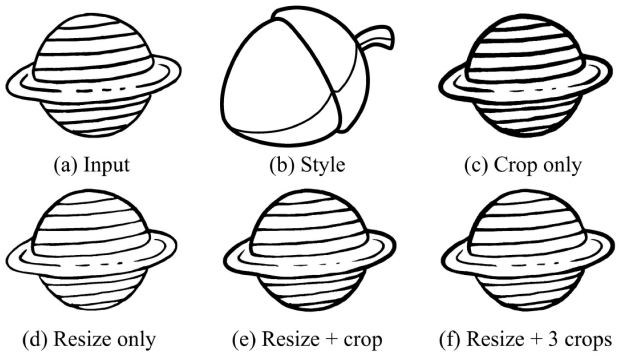


Figure 8: Comparisons on different ways of feeding the style image to the style extractor.

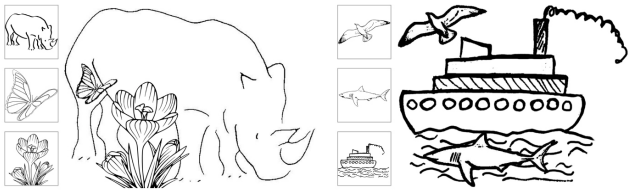


Figure 9: Application on combining multiple line drawings. In the left image, the reference style is the flower. In the right image, the reference style is the background scene.

and all the other methods on an RTX 2080Ti GPU. AdaIN, our method, and DMIT have the best performance, taking 0.6, 0.8, 1.1 seconds respectively to process a 512×512 image. WCT, TET, CCD, and Frigo perform slightly slower, taking $3.5 - 6$ seconds per image. Gatys and Image2StyleGAN are the slowest, taking 1 and 15 minutes respectively for a 512×512 image.

Applications

Combination of Multiple Line Drawings To combine multiple line drawings into one image, the users need to place the line drawings at their wanted positions and scales, and select one of them as the reference style. Then our system can automatically transfer the styles of the other line drawings to the reference style, as shown in Fig. 9.

Text Style Transfer Since hand-written text exhibit similar style features with line drawings, we can directly apply our method to transfer the styles of hand-written text, as shown in the first row of Fig. 10. Note that we directly apply our trained model without additional training. We believe that training on text images may achieve better results.

Clip Art Style Transfer While the styles of clip arts are expressed by lines and color shading, the styles of the lines have key impact on the style of the clip arts. We further apply our line style transfer methods on the lines of the clip arts by first extracting the lines, then transferring the styles of the lines, and finally recombining the lines with the color shading, as shown in the second row of Fig. 10.

Figure 10: Application on font and clip art style transfer. (c) is generated by taking (a) as input and (b) as style. (d) is generated by taking (b) as input and (a) as style.

Limitations

Our method still has some limitations. Firstly, though our method is applicable to input and style images of any resolution, the visual quality of the generated image becomes lower when the sizes of the style is significantly different from size of the input. Secondly, the collected line drawings in our training dataset may suffer from various quality issues, e.g. low resolution, JPEG compression, containing signatures with completely different styles, etc. We believe our results could be improved with a higher-quality training dataset. Furthermore, our method takes the centerline of the line drawing as content and assumes that the content remains unchanged. If the line drawing contains shading strokes or sketchy strokes, the centerline image of this line drawing will also contain the centerlines of the shading strokes or sketchy strokes, so the style of shading or sketchiness will not be captured during the style transfer. Besides, the style of content, e.g. the shape of the skeleton of hand-writing character, will not be captured as well since content remains unchanged with our system.

Conclusions

In this paper, we proposed a novel style transfer network tailored for line drawings. The centerline of the line drawing is taken as the content, so we are able to formulate this challenging problem as a centerline stylization problem and solve it via a novel style-guided image-to-image network. Experiments showed that our method significantly outperformed existing methods both qualitatively and quantitatively. While our current method only captures the style of line-width dynamics, there is potential to extend our work to transfer the style of more complex features, such as shading and sketchiness. A potential solution is to first simply the line drawing with the existing sketch simplification methods and take the simplified line drawing as content, and then capture the shading and the sketchiness of the line drawing as style. We will explore towards this direction in the future.

Acknowledgements

This work was supported in part by grants from the National Natural Science Foundation of China under Grant 61973221 and Grant 62002232, the Natural Science Foundation of Guangdong Province of China under Grant 2018A030313381 and Grant 2019A1515011165, the Key Lab of Shenzhen Research Foundation of China under Grant 201707311550233, the COVID-19 Prevention Project of Guangdong Province of China under Grant 2020KZDZX1174, and the Major Project of the New Generation of Artificial Intelligence of China under Grant 2018AAA0102900.

References

Abdal, R.; Qin, Y.; and Wonka, P. 2019. Image2StyleGAN: How to Embed Images Into the StyleGAN Latent Space? In *Proceedings of the IEEE International Conference on Computer Vision*, 4431–4440.

Azadi, S.; Fisher, M.; Kim, V. G.; Wang, Z.; Shechtman, E.; and Darrell, T. 2018. Multi-Content GAN for Few-Shot Font Style Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7564–7573.

Frigo, O.; Sabater, N.; Delon, J.; and Hellier, P. 2016. Split and Match: Example-Based Adaptive Patch Sampling for Unsupervised Style Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 553–561.

Gao, W.; Li, Y.; Yin, Y.; and Yang, M. 2020. Fast Video Multi-Style Transfer. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 3211–3219.

Garces, E.; Agarwala, A.; Gutierrez, D.; and Hertzmann, A. 2014. A similarity measure for illustration style. *ACM Trans. Graph.* 33(4): 93:1–93:9.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image Style Transfer Using Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2414–2423.

Gonzalez-Garcia, A.; van de Weijer, J.; and Bengio, Y. 2018. Image-to-image translation for cross-domain disentanglement. In *Advances in Neural Information Processing Systems*, 1294–1305.

Haralick, R. M.; and Shapiro, L. G. 1992. *Computer and Robot Vision*, volume I. USA: Addison-Wesley Longman Publishing Co., Inc., 1st edition. ISBN 0201569434.

Hertzmann, A.; Jacobs, C. E.; Oliver, N.; Curless, B.; and Salesin, D. 2001. Image analogies. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, 327–340. ACM.

Huang, X.; and Belongie, S. J. 2017. Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, 1510–1519.

Huang, X.; Liu, M.; Belongie, S. J.; and Kautz, J. 2018. Multimodal Unsupervised Image-to-Image Translation. In *Proceedings of the European Conference on Computer Vision*, volume 11207, 179–196.

Isola, P.; Zhu, J.; Zhou, T.; and Efros, A. A. 2017. Image-to-Image Translation with Conditional Adversarial Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5967–5976.

Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *Proceedings of the European Conference on Computer Vision*, volume 9906, 694–711.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *Proceedings of the 3rd International Conference on Learning Representations*.

Kotovenko, D.; Sanakoyeu, A.; Lang, S.; and Ommer, B. 2019. Content and Style Disentanglement for Artistic Style Transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, 4421–4430.

Lee, H.; Tseng, H.; Huang, J.; Singh, M.; and Yang, M. 2018. Diverse Image-to-Image Translation via Disentangled Representations. In *Proceedings of the European Conference on Computer Vision*, volume 11205, 36–52.

Lee, H.; Tseng, H.; Mao, Q.; Huang, J.; Lu, Y.; Singh, M.; and Yang, M. 2019. DRIT++: Diverse Image-to-Image Translation via Disentangled Representations. *CoRR* abs/1905.01270.

Li, W.; He, Y.; Qi, Y.; Li, Z.; and Tang, Y. 2020. FET-GAN: Font and Effect Transfer via K-shot Adaptive Instance Normalization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 1717–1724.

Li, X.; Liu, S.; Kautz, J.; and Yang, M. 2018. Learning Linear Transformations for Fast Arbitrary Style Transfer. *CoRR* abs/1808.04537.

Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; and Yang, M. 2017. Universal Style Transfer via Feature Transforms. In *Advances in Neural Information Processing Systems*, 386–396.

Liu, M.; Breuel, T.; and Kautz, J. 2017. Unsupervised Image-to-Image Translation Networks. In *Advances in Neural Information Processing Systems*, 700–708.

Mescheder, L. M.; Geiger, A.; and Nowozin, S. 2018. Which Training Methods for GANs do actually Converge? In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, 3478–3487.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral Normalization for Generative Adversarial Networks. In *Proceedings of the 6th International Conference on Learning Representations*.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, volume 9351, 234–241.

Sanakoyeu, A.; Kotovenko, D.; Lang, S.; and Ommer, B. 2018. A Style-Aware Content Loss for Real-Time HD Style Transfer. In *Proceedings of the European Conference on Computer Vision*, volume 11212, 715–731.

Shih, Y.; Paris, S.; Barnes, C.; Freeman, W. T.; and Durand, F. 2014. Style transfer for headshot portraits. *ACM Trans. Graph.* 33(4): 148:1–148:14.

Shih, Y.; Paris, S.; Durand, F.; and Freeman, W. T. 2013. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Trans. Graph.* 32(6): 200:1–200:11.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proceedings of the 3rd International Conference on Learning Representations*.

Song, C.; Wu, Z.; Zhou, Y.; Gong, M.; and Huang, H. 2019. ETNet: Error Transition Network for Arbitrary Style Transfer. In *Advances in Neural Information Processing Systems*, 668–677.

Wang, H.; Li, Y.; Wang, Y.; Hu, H.; and Yang, M. 2020. Collaborative Distillation for Ultra-Resolution Universal Style Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1857–1866.

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* 13(4): 600–612.

Yang, S.; Liu, J.; Lian, Z.; and Guo, Z. 2017. Awesome Typography: Statistics-Based Text Effects Transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2886–2895.

Yang, S.; Liu, J.; Wang, W.; and Guo, Z. 2019. TET-GAN: Text Effects Transfer via Stylization and Destylization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1238–1245.

Yi, R.; Liu, Y.; Lai, Y.; and Rosin, P. L. 2019. APDrawingGAN: Generating Artistic Portrait Drawings From Face Photos With Hierarchical GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 10743–10752.

Yu, X.; Chen, Y.; Liu, S.; Li, T. H.; and Li, G. 2019. Multi-mapping Image-to-Image Translation via Learning Disentanglement. In *Advances in Neural Information Processing Systems*, 2990–2999.

Zhang, T. Y.; and Suen, C. Y. 1984. A Fast Parallel Algorithm for Thinning Digital Patterns. *Commun. ACM* 27(3): 236–239.

Zhang, Y.; Fang, C.; Wang, Y.; Wang, Z.; Lin, Z.; Fu, Y.; and Yang, J. 2019. Multimodal Style Transfer via Graph Cuts. In *Proceedings of the IEEE International Conference on Computer Vision*, 5942–5950.