Real-Time Arbitrary Video Style Transfer

Xingyu Liu[1,2], Zongxing Ji[1,2], Piao Huang[2], Tongwei Ren[1,2,*]
[1] Shenzhen Research Institute of Nanjing University, Shenzhen, China
[2] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
{liuxingyu,jizx,151250059}@smail.nju.edu.cn,rentw@nju.edu.cn

ABSTRACT

Video style transfer aims to synthesize a stylized video that has similar content structure with a content video and is rendered in the style of a style image. The existing video style transfer methods cannot simultaneously realize high efficiency, arbitrary style and temporal consistency. In this paper, we propose the first real-time arbitrary video style transfer method with only one model. Specifically, we utilize a three-network architecture consisting of a prediction network, a stylization network and a loss network. Prediction network is used for extracting style parameters from a given style image; Stylization network is for generating the corresponding stylized video; Loss network is for training prediction network and stylization network, whose loss function includes content loss, style loss and temporal consistency loss. We conduct three experiments and a user study to test the effectiveness of our method. The experimental results show that our method outperforms the state-of-the-arts.

CCS CONCEPTS

• **Artificial intelligence** → **Reconstruction**; *Appearance and texture representations.*

KEYWORDS

Video style transfer, arbitrary style transfer, real-time, temporal consistency

ACM Reference Format:
Xingyu Liu[1,2], Zongxing Ji[1,2], Piao Huang[2], Tongwei Ren[1,2,*]. 2021. Real-Time Arbitrary Video Style Transfer. In *ACM Multimedia Asia (MMAsia '20), March 7–9, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3444685.3446301

1 INTRODUCTION

Video style transfer aims to synthesize a *stylized video* that has similar content structure with a *content video* and is rendered in the style of a *style image* [25]. Similar to many video processing tasks, exploration of video style transfer methods is extended from the research on image style transfer. Recently, image style transfer methods based on neural networks [4, 10, 33] have demonstrated exciting performance. However, directly applying these methods to video style transfer may lead to many problems, such as instability and almost imperceptible stylization effect, thereby leaving special challenges to be tackled.

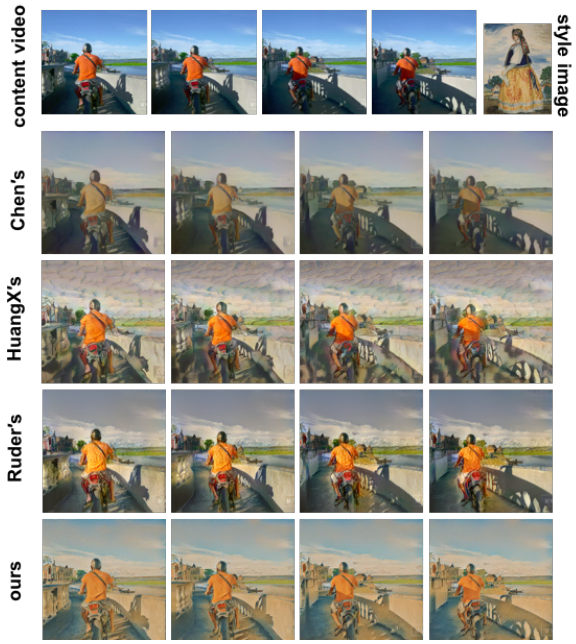

Figure 1: Comparison among different video style transfer methods. The style image in the first row specifies the reference style for processing the content video.

High efficiency, arbitrary style and temporal consistency are the three key requirements of video style transfer. Specifically, high efficiency means that the final stylized video is expected to be generated in real time; arbitrary style highlights that any style image can be applied without retraining the model; temporal consistency suggests that consistent stylization effect is required to be created on adjacent video frames.

To the best of our knowledge, no existing video style transfer methods can simultaneously satisfy all the aforementioned requirements. Some methods [1, 25] can achieve arbitrary style transfer for videos without introducing obvious jitters, but at low efficiency. Other methods [7, 11, 15, 21, 27, 28, 31] can process any given style images at desired speed, but cannot solve the problem of instability. Although some previous methods [5, 12, 14, 26] succeed in reducing the time cost for producing stable stylized videos, they can only capture a limited number of style images, *i.e.*, they need

* Corresponding author.

MMAsia '20, March 7–9, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8308-0/21/03…$15.00
https://doi.org/10.1145/3444685.3446301

to train new models when additional style images are involved. As shown in Figure 1, the reference style is almost imperceptible on the stylized video generated by Chen *et al.* [7]; HuangX *et al.* [15] fails to produce stable stylization effect as represented by the sky in the background; Both Ruder *et al.* [25] and ours can keep temporal consistency of the stylized videos.

To address the limitations of the existing video style transfer methods, we propose a novel real-time arbitrary video style transfer method with a three-network architecture, which can efficiently produce temporally consistent stylized videos using arbitrary style images. Figure 2 shows an overview of our method. Prediction network is trained to predict the parameters of the style image for conditional instance normalization; stylization network is a feed-forward network that can efficiently transfer content video frames to stylized video frames; loss network is used for training prediction network and stylization network, which integrates a VGG-19 network [29] pretrained by ImageNet and a temporal consistency loss. In particular, drawing inspiration from conditional instance normalization for arbitrary transformation [9, 11, 15] and feed-forward network for temporal consistency [14, 25], we explore a way of adding conditional instance normalization to a feed-forward network to transfer video frames without the assistance of optical flows in the stylization process.

We construct a video style transfer dataset with 350 content videos downloaded from the Internet and 10,000 style images on the basis of the Kaggle dataset [18], on which we compare our method with other typical methods under the criteria of efficiency and arbitrariness. We also randomly select three content videos with ground truth optical flows in the Sintel dataset [2] to evaluate the effectiveness of our method in terms of temporal consistency. Moreover, we conduct a user study with 2,400 test cases equally assigned to 30 volunteers. The experimental results show that our method is superior to the state-of-the-arts.

In summary, our contributions are threefold: (1) We propose a three-network architecture for real-time arbitrary video style transfer. (2) We realize arbitrary stylization for videos with a single model, which is proved to be very effective. (3) We address the problem of instability and discontinuity with temporal coherence preserved among consecutive video frames.

2 RELATED WORK

2.1 Image style transfer

Image style transfer aims to render a content image in the style of a style image [16]. Gatys *et al.* [10] formulate image style transfer as an optimization problem that encourages the stylized image to show neural activations of the content image and feature correlations of the style image within a single neural network. They use Gram matrices to represent the style of the input image. Li *et al.* [23] provide a novel interpretation of neural style transfer, and explain why Gram matrices could represent style. Yin [33] extends Gatys's model to transfer the style from a high resolution image to a low resolution content image. Chen and Hsu [8] propose a masking out strategy and high-order statistics for content-aware style transfer. Champandard [4] proposes a content-aware generative model that utilizes the semantic map that is either manually annotated or automatically generated to give meaningful control over the operation of style transfer. Castillo *et al.* [3] present a targeted style transfer method for stylizing an object specified either by the user or a semantic segmentation algorithm with non-stylized background.

In regard to efficiency, Johnson *et al.* [17] train feed-forward networks assisted with a loss network to solve the same optimization problem as [10]. Ulyanov *et al.* [30] propose texture networks to synthesize textures by a single forward pass. Li *et al.* [20] use adversarial generative networks to improve the efficiency of texture synthesis. In order to avoid compromising generality while improving efficiency, Li *et al.* [22] propose to encode multiple textures in a single generative feed-forward network. Chen *et al.* [6] propose StyleBank to explicitly represent various styles by multiple convolution filters. Dumoulin *et al.* [9] firstly propose conditional instance normalization to realize a single and scalable deep network, which can capture N styles. Zhang *et al.* [34] propose a Multi-style Generative Network (MSGNet) to achieve real-time performance as well as model flexibility.

2.2 Video style transfer

Efficient video style transfer. Ruder *et al.* [26] train a feed-forward network with the prior image, *i.e.*, the warped previous stylized frame, to create stylized videos with a much lower run time per frame. Huang *et al.* [14] also train a feed-forward convolutional neural network using a smaller number of channels to accelerate the inference speed. Gupta *et al.* [12] propose a recurrent convolutional network to produce video frames without explicitly computing optical flow, which is feasible to run in real-time. It should be noted that models of these methods cannot support arbitrary style transfer. When new style images are needed for stylization, these methods need to train new models.

Arbitrary video style transfer. As the follow-up study of [9], Ghiasi *et al.* [11] train a style prediction network to predict affine transformation parameters in [9]. HuangX *et al.* [15] propose adaptive instance normalization for real-time arbitrary style transfer by simply aligning the channel-wise mean and variance of the content feature maps to those of the style feature maps. Chen *et al.* [7] propose a patch-based swap operation for constructing the target activations in feature space, and then decode with an inverse network to generate stylized frames in image space. Shen *et al.* [27] propose a style decorator module that can be easily embedded into an image reconstruction network to render multi-scale stylization for any given style image in one feed-forward pass. Although these methods meet the requirement of efficiency and flexibility, problems such as instability and imperceptible stylization still exist in the final stylized videos.

Temporally consistent video style transfer. In order to enhance the coherence of the stylized videos, Ruder *et al.* [25] extend Gatys's model [10] by taking optical flows into account and initialize each stylized frame with the warped previous one. They also introduce a temporal constraint to further strengthen the continuity of moving objects. On the basis of Gatys's model, Anderson *et al.* [1] apply optical flows to initialize the style transfer optimization for each stylized frame. However, both methods are time-consuming in consequence of initialization and gradient decent procedure to stylize each frame. Chen *et al.* [5] propose the

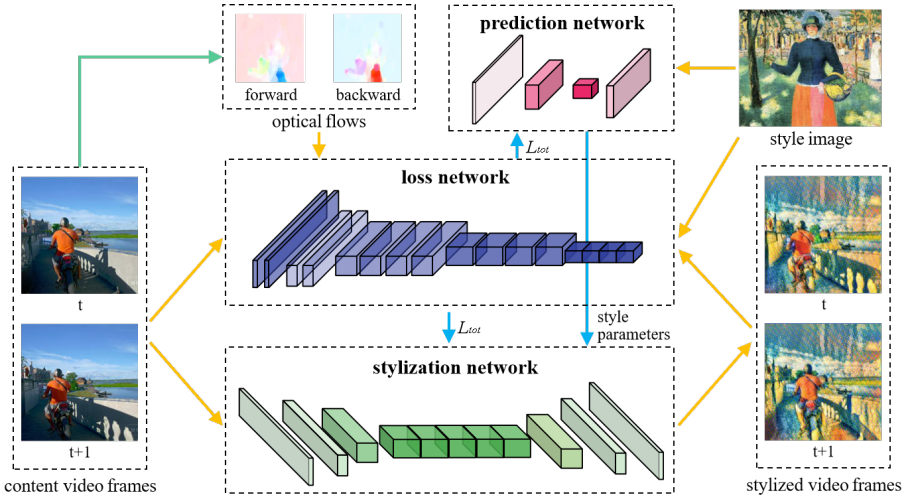

Figure 2: An overview of our proposed method. The arrows denote the dataflow in the training process. Specifically, the yellow arrows denote the inputs and outputs of the three networks; the blue arrows denote the parameters transferred among the networks; the green arrow denotes the estimation process of optical flows.

first end-to-end model for video style transfer, which propogates short-term consistency to ensure the continuity and stability of the stylized videos, but at the cost of generality.

3 METHOD

3.1 Overview

We propose a novel video style transfer method for real-time arbitrary video stylization. The whole framework is shown in Figure 2. Prediction network extracts style parameters from a given style image. Stylization network obtains style parameters from prediction network and stylizes each frame of a content video to generate the corresponding stylized video. Loss network is used for training prediction network and stylization network, whose loss function includes content loss, style loss and temporal consistency loss. Two adjacent content video frames are required in the training process, but only one content video frame is required as the input of the stylization network in the stylization process.

Training process. We use a style image and two adjacent frames from a content video as inputs for each training. We first feed the style image into prediction network to extract style parameters. We then feed the two adjacent content frames and the extracted style parameters into stylization network to generate two adjacent stylized frames. Next, we estimate both forward and backward optical flows between the two adjacent content frames, which are used to warp the previous and the subsequent stylized frames, respectively. Finally, we compare two stylized frames with the style image and the corresponding content frames to calculate style loss and content loss. Temporal consistency loss is calculated by comparing each stylized frame with the warped previous and subsequent stylized frames. The three losses are weighted and summed to calculate the total loss, which is then minimized for the optimization of model parameters of both prediction network and stylization network.

Stylization process. With adequate training, prediction network acquires the ability to effectively extract style parameters, and stylization network realizes temporal consistency constraints. Therefore, when a content video is stylized, only one content frame is required each time to be fed into stylization network, which allows parallel processing of all the content frames for efficiency.

3.2 Prediction Network

Arbitrary video style transfer is based on the premise that style parameters can be extracted from a randomly given style image in prediction network. Inspired by [11], we use a four-layer prediction network for parameter extraction. To improve training efficiency, we resize the resolution of the style image to 256×256. On the basis of Inception-v3 architecture, we then obtain a feature vector whose channel size and channel number are 17×17 and 768, respectively. Next, we calculate the mean of each channel and the channel size of the new feature vector is compressed to 1×1. Finally, the new feature vector passes through two fully connected layers to obtain a style parameter vector $\vec{S} = \{\vec{\gamma}_s, \vec{\beta}_s\}$. Specifically, the first fully connected layer is used to reduce the channel number to 100, while the second fully connected layer is used to increase the channel number to 2758.

3.3 Stylization Network

According to [9], conditional instance normalization can satisfy the requirements of arbitrariness and efficiency, which simplifies the procedure of stylizing each content frame as:

$$\hat{z} = \gamma_s\left(\frac{z-\mu}{\sigma}\right) + \beta_s, \tag{1}$$

where z denotes an activation unit and $\hat{z}$ is the tuned activation unit; μ and σ represent the mean and standard deviation of z, respectively; γ_s and β_s are style parameters extracted from any given style image in prediction network.

Table 1: Stylization network configuration. The convolutional layer parameters are denoted as "conv <kernel size>-<number of channels>".

Component	Layer	Stride	Activation
convolutional block	conv 9-32	1	relu
convolutional block	conv 3-64	2	relu
convolutional block	conv 3-128	2	relu
residual block ($\times 5$)	conv 3-128	1	relu
	conv 3-128	1	linear
upsampling block	conv 3-64	1	relu
upsampling block	conv 3-32	1	relu
convolutional block	conv 9-3	1	sigmoid

However, using Equation (1) alone cannot ensure temporal consistency of the stylized video. Hence, we utilize a 16-layer stylization network to constrain the mapping from the pixels in content frames to the activation units. Table 1 lists the configuration of our stylization network.

Inspired by [17], we set the first three components of stylization network as convolutional blocks. Each convolutional block includes a convolutional layer, conditional instance normalization and activation. The next five components are residual blocks [13] and each residual block contains two convolutional layers. The following two components are upsampling blocks that use nearest-neighbor interpolation for upsampling. The last component is also a convolutional block.

Because reducing the number of channels may cause the degradation of stylization quality, we do not follow the strategy of channel number reduction in [14]. Moreover, after each convolution we replace the batch normalization with conditional instance normalization to apply the style parameters extracted from prediction network to stylization network.

3.4 Loss Network

To optimize the model parameters in both prediction network and stylization network, we use a loss network with two-frame synergic training mechanism [14]. Figure 3 shows the procedure of loss network. We refer to $\boldsymbol{f^t}$ as the tth frame in a content video; $\boldsymbol{a}$ represents a style image; $\boldsymbol{x^t}$ denotes the corresponding stylized frame of $\boldsymbol{f^t}$. We feed two adjacent content frames $\boldsymbol{f^t}$ and $\boldsymbol{f^{t+1}}$, two adjacent stylized frames $\boldsymbol{x^t}$ and $\boldsymbol{x^{t+1}}$, together with a style image $\boldsymbol{a}$ into the aforementioned VGG-19 network [29], which extracts the feature maps F^{lt}, A^{l} and X^{lt} with the dimensionality of $M^l \times N^l$ from $\boldsymbol{f^t}$, $\boldsymbol{a}$ and $\boldsymbol{x^t}$ in layer l .

Similar to [24], the loss function in loss network consists of content loss, style loss and temporal consistency loss, which is calculated as follows:

$$\mathcal{L}_{tot} = \alpha \mathcal{L}_{con} + \beta \mathcal{L}_{sty} + \gamma \mathcal{L}_{tem}, \tag{2}$$

where $\mathcal{L}_{tot}$, $\mathcal{L}_{con}$, $\mathcal{L}_{sty}$ and $\mathcal{L}_{tem}$ denote total loss, content loss, style loss and temporal consistency loss, respectively; α, β and γ are weight parameters to control the influences of different losses exerted on stylization effect.

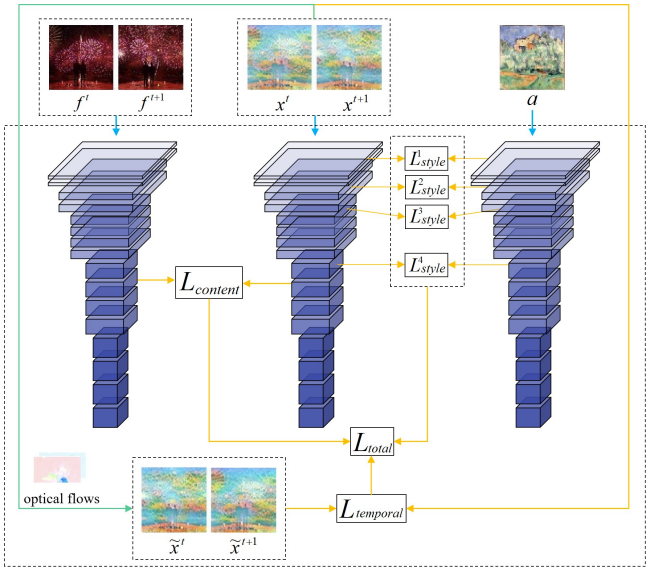

Figure 3: The procedure of loss network. The yellow arrows denote the inputs of loss calculation; the blue arrows denote the inputs of VGG-19; the green arrow denotes the warping process of stylized frames with optical flows.

Content loss. Content loss is used to evaluate the appearance similarity between content frames and the corresponding stylized frames. We select relu 4_2 layer from the VGG-19 network as the content layer and calculate content loss as follows:

$$\mathcal{L}_{con} = \sum_{k \in \{t, t+1\}} \sum_{i=1}^{M^l} \sum_{j=1}^{N^l} (\mathrm{X}_{ij}^{lk} - \mathrm{F}_{ij}^{lk})^2. \tag{3}$$

Style loss. Style loss is used to evaluate the appearance similarity between the style image and the stylized frames. We select relu 1_1, relu 2_1, relue 3_1,relu 4_1 layers from the VGG-19 network for style loss calculation:

$$\mathcal{L}_{sty} = \sum_{k \in \{t, t+1\}} \sum_{l=1}^{L} \frac{\lambda^l}{(M^l N^l)^2} \sum_{i=1}^{M^l} \sum_{j=1}^{N^l} \left(\mathcal{G}_{ij}(\mathrm{X}^{lt}) - \mathcal{G}_{ij}(\mathrm{A}^{l})\right)^2, \tag{4}$$

where $\mathcal{G}_{ij}(\cdot)$ denotes the (i, j) position of the Gram matrix, which represents the feature correlation based on inner product, *e.g.*, $\mathcal{G}_{ij}(\mathrm{X}^l) = \sum_{k=1}^{N^l} \mathrm{X}^l_{ik} \mathrm{X}^l_{jk}$; λ^l is a weight threshold with the default value of 1; L is the number of layers, which equals 4.

Temporal consistency loss. Temporal consistency loss is used for evaluating the coherence between two adjacent stylized frames. In the training process, we represent the pixel correspondence between adjacent video frames with optical flows, which is estimated by Deepflow [32]. The time cost of optical flow estimation does not affect the efficiency of our video style transfer method because it is only required in the training process.

Similar to [25], We detect the motion boundaries of the current content frame to obtain the disoccluded regions, *i.e.*, blurring areas produced by moving objects, and compare the stylized result with

Table 2: The time costs of different methods for stylizing each video frame in seconds under three resolutions.

Method	Resolution			Style	Temporal Consistency
	256	512	1024		
Chen [7]	0.12	1.50	-	∞	no
Ghiasi [11]	**0.01**	**0.03**	**0.09**	∞	no
Huang [14]	**0.01**	**0.03**	**0.09**	1	yes
HuangX [15]	0.03	0.10	0.38	∞	no
Johnson [17]	0.01	0.05	0.17	1	no
Li [21]	0.62	1.14	2.95	∞	no
Ruder [25]	14.91	56.26	524.50	∞	yes
Ulyanov [30]	0.02	0.05	0.15	1	no
Ours	**0.01**	**0.03**	**0.10**	∞	yes

the warped previous or subsequent stylized frame in rest regions:

$$\mathcal{L}_{tem} = \frac{1}{|\mathcal{H}^t|} \sum_{p_{ij}^t \in \mathcal{H}^t} (\boldsymbol{x}_{ij}^t - \widetilde{\boldsymbol{x}}_{ij}^{t+1})^2 + \frac{1}{|\mathcal{H}^{t+1}|} \sum_{p_{ij}^{t+1} \in \mathcal{H}^{t+1}} (\boldsymbol{x}_{ij}^{t+1} - \widetilde{\boldsymbol{x}}_{ij}^{t})^2, \tag{5}$$

where $\mathcal{H}^t$ denotes the set of pixels that belong to the rest regions of $\boldsymbol{f}^t$; p_{ij}^t denotes the pixel belonging to $\mathcal{H}^t$ in (i, j); $\boldsymbol{x}_{ij}^t$ denotes the stylized result of p_{ij}^t; $\widetilde{\boldsymbol{x}}_{ij}^t$ denotes the pixel in the frame warped from $\boldsymbol{x}^t$ with forward optical flow, while $\widetilde{\boldsymbol{x}}_{ij}^{t+1}$ denotes the pixel in the frame warped from $\boldsymbol{x}^{t+1}$ with backward optical flow; $|\cdot|$ denotes the cardinality of a set.

Different from the loss function in [24], our method employs a two-frame synergic training mechanism with the correspondences between two adjacent frames considered, thereby achieving temporal consistency.

4 EXPERIMENT

We collected 350 content videos from the Internet, including 266 real content videos (76%) and 84 cartoon videos (24%), with the lengths varying from 2*s* to 20*s*. We also clustered the 80,000 style images in the training set of Kaggle [18] into 10,000 categories according to their color histograms, and selected the image with the minimal distance to the cluster center in each category as our style images. In this way, we constructed a dataset with 350 content videos and 10,000 style images for video style transfer.

4.1 Experiment settings

In the training process, all the video frames as well as the style images were resized to the resolution of 256×256, and the optical flows were estimated for each video in advance for improving training speed. We set α, β and γ in Equation (2) as 1, 1e-3 and 1e3 respectively to balance the influences of three losses, and used Adam [19] for stochastic gradient descent to update the model parameters. The learning rate was set to 10e-4. We trained our stylization network with approximately 8 million iterations. In each training, two adjacent content frames and one style image were randomly selected as the inputs. It took about 240 hours for training our stylization network and prediction network. We conducted all the experiments on a computer with i7 3.5GHz CPU, 32GB memory and 1080Ti GPU.

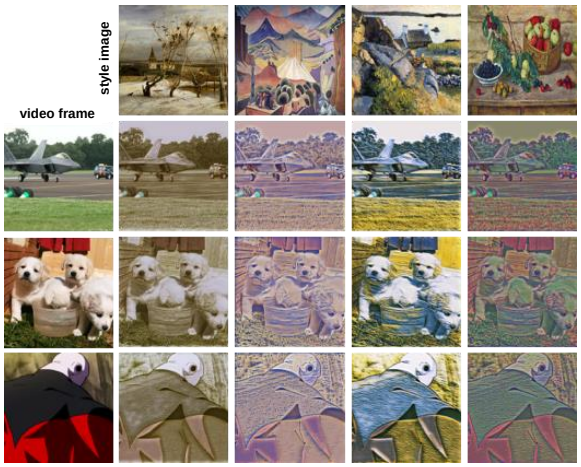

Figure 4: Examples of stylized video frames. Different stylizaion effects are generated with four style images.

4.2 Efficiency study

We compared the efficiency of our method with eight existing typical methods, including Chen [7], Ghiasi [11], Huang [14], HuangX [15], Johnson [17], Li [21], Ruder [25] and Ulyanov [30], on ten randomly selected videos and calculated the average time cost for stylizing each frame. The results are shown in Table 2. It should be specified that the time cost of Chen [7] under the resolution of 1024×1024 was not included because its requirement of GPU memory exceeds that of our computer.

We can see that: 1) In terms of time cost, Ruder [25] is inferior to other methods by a wide margin under any resolution. 2) Our method can achieve real-time video style transfer below the resolution of 512×512. It is ascribed to our stylization network, which allows parallel stylizing of all the content frames with a single forward pass.

4.3 Arbitrariness study

Our method uses 10,000 style images in the training process. After adequate training, our model can generate stylized videos with any given style images. We conducted an experiment to test the generality of our method with the style images randomly selected from the test set of Kaggle [18]. Figure 4 shows the results of our experiment. We can see that the stylized results well represent the characteristics of the style images while preserving the original content structure.

4.4 Temporal consistency study

We compared our method with three representative video style transfer methods, *i.e.*, Ruder [25], Johnson [17] and HuangX [15], using three content videos randomly selected from the Sintel dataset [2], which provides ground truth optical flows. We used *Still Life. Apples in a sieve* by Pyotr Konchalovsky as the style image and adopted the temporal consistency error E_{tem} defined by Huang [14] as Equation (6), which is the average pixel-wise

Table 3: Temporal consistency errors of different methods. Alley 1, Bamboo 1 and Market 6 are the three corresponding stylized videos.

Method	Alley 1	Bamboo 1	Market 6
Johnson [17]	0.09	0.10	0.15
Ruder [25]	0.03	0.04	0.09
HuangX [15]	0.08	0.09	0.12
Ours	0.06	0.07	0.10

Table 4: User study results of the stylized videos generated by four methods. *content retention*, *style identifiability*, *stability & continuity* and *like* are the evaluation criteria summarized from the four questions for volunteers.

Method	content retention	style identifiability	stability & continuity	like
Chen [7]	**2358**	210	719	286
HuangX [15]	1539	808	109	598
Ruder [25]	1820	1492	**2293**	1539
Ours	1984	**1538**	2167	**1755**

Euclidean color difference between consecutive frames:

$$E_{tem} = \sqrt{\frac{\sum_{t=1}^{T-1} \sum_{p_{ij}^{t+1} \in \mathcal{H}^{t+1}} (\boldsymbol{x}_{ij}^{t+1} - \widetilde{\boldsymbol{x}}_{ij}^{t})^2}{(T-1) \cdot |\mathcal{H}^{t+1}|}}, \tag{6}$$

where T is the total number of frames. It is worth noting that E_{tem} is different from that in Equation (5) because Sintel [2] only offers forward optical flows.

From Table 3, we can see that in each stylized video, our E_{tem} value is next only to that of Ruder [25]. However, as metioned in the efficiency study, Ruder [25] consumes more time to stylize each video frame than ours.

4.5 User study

We also carried out a user study for quantitative contrast between our method and three state-of-the-art video style transfer methods, *i.e.*, Chen [7], HuangX [15] and Ruder [25]. All the stylized videos were generated using 120 content videos and 20 style images randomly selected from the aforementioned video style transfer dataset, composing 2,400 test cases in total. Specifically, each test case consists of a style image, a content video and four randomly distributed stylized videos. To ensure the effectiveness of our comparisons, these five videos were played simultaneously.

Thirty volunteers aged from 18 to 45 (male : female = 1 : 1) were invited to participate in the user study. The 2,400 test cases were equally assigned to the volunteers. During the exhibition of the videos, each volunteer was asked to evaluate each stylized video by answering four questions: 1) Can you obtain sufficient content information from the stylized video as compared to the content video (*content retention*)? 2) Can you recognize the reference style from the stylized video (*style identifiability*)? 3) Do you think the stylized video is free from jitters (*stability & continuity*)? 4) Do you like the stylized video (*like*)?

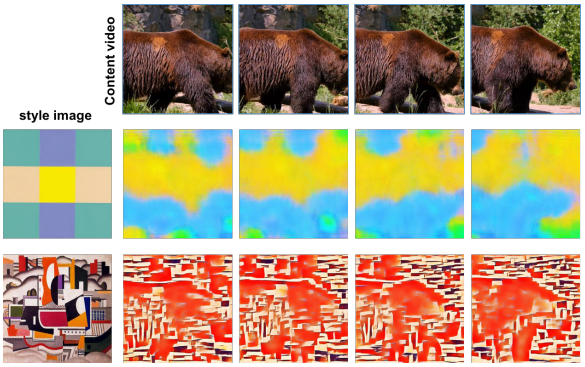

Figure 5: Examples of strong rendering in stylized frames produced by two inappropriate style images. The images in the first row are four consecutive frames extracted from the content video that displays a bear walking in a forest.

Table 4 shows the user study results. We can see that: 1) Both Chen [7] and our models perform very well under the criterion of *content retention*. 2) Although some reference styles cannot be easily rendered on specific content videos, our method still excels in *style identifiability*. 3) Ruder [25] and ours are the only two methods that avoid discontinuity, surpassing HuangX [15] by a large margin. 4) Judged by the number of *likes*, our method excels all other methods. Interestingly, *like* is slightly higher than *style identifiability* for our method. Although the reference styles are not recognizable in some of our stylized videos, volunteers still have a favorable attitude toward these videos.

Using inappropriate style images whose structural characteristics are largely different from those of content video frames may cause strong style rendering. As shown in Figure 5, when a style image with simple color blocks or abstract patterns is used for stylization, the original content information is imperceptible in the final stylized video, *i.e.*, it is hard to distinguish the bear from the background. Hence, content retention and style identifiability need to be well balanced in the future work.

5 CONCLUSION

In this paper, we proposed a novel real-time arbitrary video style transfer method using a three-network architecture, which consists of a prediction network, a stylization network and a loss network. As far as we know, it is the first method that can simultaneously satisfy the three key requirements of video style transfer, *i.e.*, high efficiency, arbitrary style and temporal consistency. We validated the effectiveness of our method by conducting three experiments and a user study. The experimental results show that our method is superior to the state-of-the-art video style transfer methods.

6 ACKNOWLEDGEMENT

This work is supported by National Science Foundation of China (62072232), Natural Science Foundation of Jiangsu Province (BK20191248), Science, Technology and Innovation Commission of Shenzhen Municipality (JCYJ20180307151516166), and Collaborative Innovation Center of Novel Software Technology and Industrialization.

REFERENCES

[1] Alexander G. Anderson, Cory P. Berg, Daniel P. Mossing, and Bruno A. Olshausen. 2016. DeepMovie: Using Optical Flow and Deep Neural Networks to Stylize Movies. *arXiv:1605.08153* (2016).

[2] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. 2012. A Naturalistic Open Source Movie for Optical Flow Evaluation. In *European Conference on Computer Vision*.

[3] Carlos Castillo, Soham De, Xintong Han, Bharat Singh, Abhay Kumar Yadav, and Tom Goldstein. 2017. Son of Zorn's Lemma: Targeted Style Transfer Using Instance-aware Semantic Segmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*.

[4] Alex Champandard. 2016. Semantic Style Transfer and Turning Two-Bit Doodles into Fine Artworks. *arXiv:1603.01768* (2016).

[5] Dongdong Chen, Jing Liao, Lu Yuan, Nenghai Yu, and Gang Hua. 2017. Coherent Online Video Style Transfer. In *IEEE International Conference on Computer Vision*.

[6] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. 2017. StyleBank: An Explicit Representation for Neural Image Style Transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[7] Tian Chen and Mark Schmidt. 2016. Fast Patch-based Style Transfer of Arbitrary Style. In *Annual Conference on Neural Information Processing Systems Workshop on Constructive Machine Learning*.

[8] Yi-Lei Chen and Chiou-Ting Hsu. 2016. Towards Deep Style Transfer: A Content-Aware Perspective. In *British Machine Vision Conference*.

[9] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. 2017. A Learned Representation For Artistic Style. In *International Conference on Learning Representations*.

[10] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2016. Image Style Transfer Using Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[11] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. 2017. Exploring the structure of a real-time, arbitrary neural artistic stylization network. In *British Machine Vision Conference*.

[12] Agrim Gupta, Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2017. Characterizing and Improving Stability in Neural Style Transfer. In *IEEE International Conference on Computer Vision*.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[14] Haozhi Huang, Hao Wang, Wenhan Luo, Lin Ma, and Wei Liu. 2017. Real-Time Neural Style Transfer for Videos. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[15] Xun Huang and Serge Belongie. 2017. Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[16] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. 2020. Neural Style Transfer: A Review. *IEEE Transactions on Visualization and Computer Graphics* 26, 11 (2020), 3365–3385.

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *European Conference on Computer Vision*.

[18] Kaggle. 2018. painter-by-numbers. https://www.kaggle.com/c/painter-by-numbers.

[19] Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.

[20] Chuan Li and Michael Wand. 2016. Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks. In *European Conference on Computer Vision*.

[21] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. 2017. Universal Style Transfer via Feature Transforms. In *Annual Conference on Neural Information Processing Systems*.

[22] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, and Ming Hsuan Yang. 2017. Diversified Texture Synthesis with Feed-Forward Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[23] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. 2017. Demystifying Neural Style Transfer. In *International Joint Conference on Artificial Intelligence*.

[24] Xingyu Liu, Jingfan Guo, Tongwei Ren, Yahong Han, and Gangshan Wu. 2018. HeterStyle: A Heterogeneous Video Style Transfer Application. In *ACM Multimedia Conference Demo*.

[25] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. 2016. Artistic Style Transfer for Videos. In *German Conference on Pattern Recognition*.

[26] Manuel Ruder, Alexey Dosovitskiy, and Thomas Brox. 2018. Artistic Style Transfer for Videos and Spherical Images. *International Journal of Computer Vision* 126, 11 (2018), 1199–1219.

[27] Falong Shen, Shuicheng Yan, and Gang Zeng. 2017. Meta Networks for Neural Style Transfer. *arXiv:1709.04111* (2017).

[28] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. 2018. Avatar-Net: Multi-scale Zero-shot Style Transfer by Feature Decoration. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[29] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations*.

[30] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. 2016. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. In *International Conference on Machine Learning*.

[31] Hao Wang, Xiaodan Liang, Hao Zhang, Dit Yan Yeung, and Eric P. Xing. 2017. ZM-Net: Real-time Zero-shot Image Manipulation Network. *arXiv:1703.07255* (2017).

[32] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. 2014. DeepFlow: Large Displacement Optical Flow with Deep Matching. In *IEEE International Conference on Computer Vision*.

[33] Rujie Yin. 2016. Content Aware Neural Style Transfer. *arXiv:1601.04568* (2016).

[34] Hang Zhang and Kristin Dana. 2017. Multi-style Generative Network for Real-time Transfer. In *European Conference on Computer Vision*.