Coherent Online Video Style Transfer

Dongdong Chen[1], Jing Liao[2], Lu Yuan[2], Nenghai Yu[1,*], Gang Hua[2]

[1]University of Science and Technology of China [2]Microsoft Research, Beijing, China

cd722522@mail.ustc.edu.cn, {jliao, luyuan, ganghua}@microsoft.com, ynh@ustc.edu.cn

Abstract

Training a feed-forward network for the fast neural style transfer of images has proven successful, but the naive extension of processing videos frame by frame is prone to producing flickering results. We propose the first end-to-end network for online video style transfer, which generates temporally coherent stylized video sequences in near real-time. Two key ideas include an efficient network by incorporating short-term coherence, and propagating short-term coherence to long-term, which ensures consistency over a longer period of time. Our network can incorporate different image stylization networks and clearly outperforms the per-frame baseline both qualitatively and quantitatively. Moreover, it can achieve visually comparable coherence to optimization-based video style transfer, but is three orders of magnitude faster.

1. Introduction

Inspired by the success of work from Gatys et al. [18] on neural style transfer, there has been a number of recent works [38, 29, 9, 19] addressing the problem of style transfer using deep neural networks. In their approaches, style transfer is formulated as an optimization problem, i.e. searching for a new image presenting similar neural activations as the content image and similar feature correlations as the style image. Notwithstanding their impressive results, these methods are very slow in runtime. To mitigate this issue, many recent works[25, 40, 30, 10, 12, 31] train feed-forward networks to speed up the transfer process. Such techniques have been successfully applied to a number of popular apps such as Prisma, Pikazo, and DeepArt.

Extending neural style transfer from image to video may produce new and impressive effects, whose appeal is especially strong in short video sharing, live-view effects, and entertainment video. The approaches discussed above, when naively extended to process each frame independently, often lead to flickering and false discontinuities. This is because the solution of the style transfer task is not stable. For optimization-based methods (*e.g.*, [18]), the instability stems from the random initialization and local minima of the loss function. For feed-forward methods (*e.g.*, [25]), small perturbations in the content images, *e.g.*, lighting, noises and motions may cause large variations in stylized results (Figure 1). Consequently, it is essential to explore temporal consistency in videos for stable outputs.

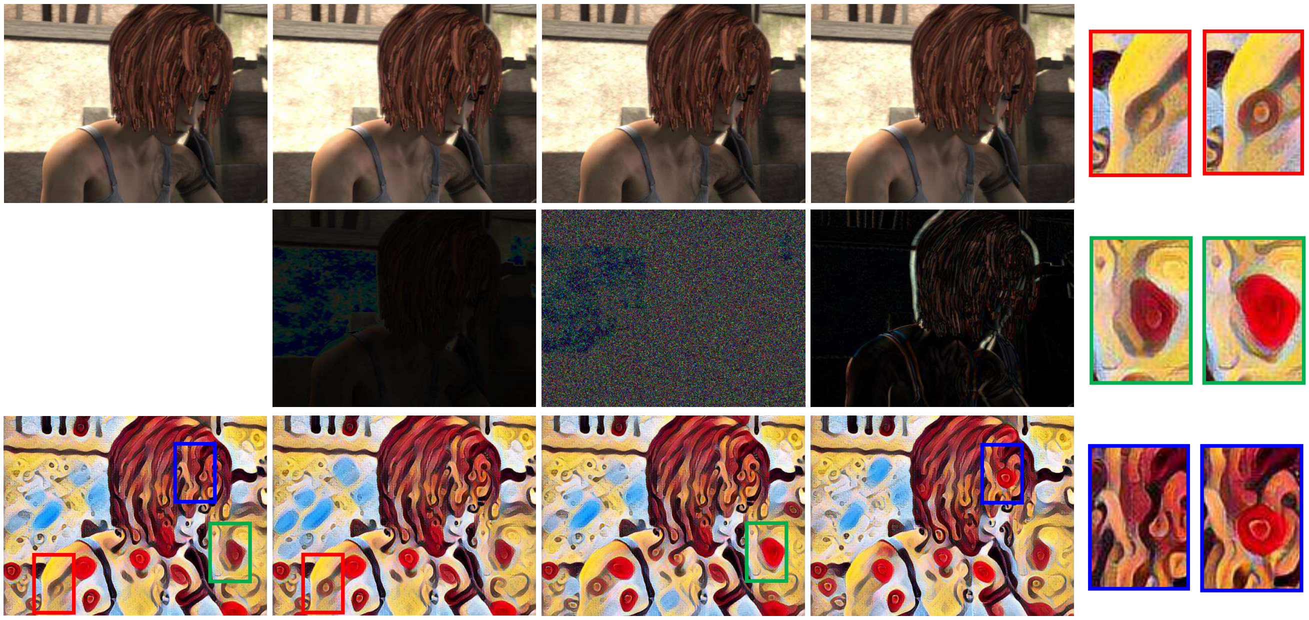
Anderson et al. [1] and Ruder et al. [37] address the problem of flickers in the optimization-based method by introducing optical flow to constrain both the initialization and the loss function. Although very impressive and smoothing stylized video sequences are obtained, their runtime is quite slow (usually several minutes per frame), making it less practical in real-world applications.

In this paper, we present the first feed-forward network leveraging temporal information for video style transfer, which follows a recurrent formulation. By incorporating a flow sub-network and a mask sub-network into a certain intermediate layer of a pre-trained stylization network (*e.g.*, [25, 10]), our method is able to produce consistent and stable stylized video sequences in near real-time.

The flow sub-network, motivated by [45], estimates dense feature correspondences between consecutive frames. It helps all consistent points along the motion trajectory align in the feature domain. The mask sub-network identifies the occlusion or motion discontinuity regions. It helps adaptively blend feature maps from previous frames and the current frame to avoid ghosting artifacts. The entire architecture is trained end-to-end with a new loss function, jointly considering stylization and temporal coherence.

To obtain stable video stylization results, long-term consistency should be considered and was enforced by additional loss terms with distant frames [37]. Instead, our feed-forward network adopts a new Recurrent Neural Network (RNN) architecture [15]. In contrast to traditional RNNs, we only consider short-term consistency (*e.g.*, two frames)

*Corresponding Author



(a) (b) (c) (d) (e)

Figure 1. The image stylization network (*e.g.*, [25]), will amplify some unnoticeable changes in inputs. The top row shows the four inputs: (a) the original one, (b) 5% lighter than (a), (c) Gaussian noises ($\mu = 0, \sigma = 1e-4$) added to (a); and (d) the next frame of (a) with subtle motions. The middle rows show the absolute difference between (a) and other three inputs. For better visualization, these differences are boosted by $3\times$. The bottom row shows the corresponding stylization results. (e) shows close-up views of some flickering regions.

in the training stage, which can effectively circumvent the vanishing and exploding gradient problems of RNNs [2]. In the inference stage, the short-term consistency can approximate long-term consistency by propagation. By this way, once one point can be traced along motion trajectories, its stylization result will keep consistent until the track ends.

In summary, our video style transfer network is unique in the following aspects:

- Our network is the first network leveraging temporal information that is trained end-to-end for video style transfer, which successfully generates stable results.
- Our feed-forward network is thousands of times faster compared to optimization-based style transfer in videos [1, 37], reaching 15 fps on modern GPUs.
- Our method enables online processing, and is cheap in both learning and inference, since we achieve good approximation of long-term temporal coherence by propagating short-term ones.
- Our network is general, and successfully applied to several existing image stylization networks, including per-style-per-net [25] and multiple-style-per-net [10].

2. Related Work

2.1. Style Transfer for Images and Videos

Traditional image stylization works mainly focus on texture synthesis based on low-level features, which use non-parametric sampling of pixels or patches in given source texture images [14, 22, 13] or stroke databases [32, 21]. Their extension to video mostly uses optical flow to constrain the temporal coherence of sampling [5, 20, 33]. A comprehensive survey can be found in [27].

Recently, with the development of deep learning, using neural networks for stylization has become an active topic. Gatys et al. [18] first propose using pre-trained Deep Convolutional Neural Networks (CNN) for image stylization. It generates more impressive results compared to traditional methods because CNN provides more semantic representations of styles. To further improve transfer quality, different complementary schemes have been proposed, including face constraints [38], Markov Random Field (MRF) prior [29], user guidance [9] and controls [19]. Unfortunately, these methods based on an iterative optimization are computationally expensive in run-time, which imposes a big limitation in real applications. To make the run-time more efficient, some work directly learn a feed-forward generative network for a specific style [25, 40, 30] or multiple styles [10, 12, 31] which are hundreds of times faster than optimization-based methods.

Another direction of neural style transfer [18] is to extend it to videos. Naive solutions that independently process each frame tend to produce flickers and false discontinuities. To preserve temporal consistency, Alexander et al. [1] use optical flow to constrain optimization initialization, and incorporate flow explicitly into the loss function.

To further reduce ghosting artifacts at the boundaries and occluded regions, Ruder et al. [37] introduce masks to filter out the flow with low confidences in the loss function. This allows them to generate consistent and stable stylized video sequences, even in cases with large motion and strong occlusions. Notwithstanding their demonstrated success in video style transfer, it is very slow due to iterative optimization. Feed-forward networks [25, 40, 30, 10, 12, 31] have proven efficient in image style transfer. However, we are not aware of any work that trains a feed-forward network that explicitly takes temporal coherence into consideration in video style transfer.

2.2. Temporal Coherence in Video Filter

Video style transfer can be viewed as applying one kind of artistic filter on videos. How to preserve the temporal coherence is essential and has been considered in previous video filtering work. One popular solution is to temporally smooth filter parameters. For instance, Bonneel et al. [3] and Wang et al. [41] transfer the color grade of one video to another by temporally filtering the color transfer functions.

Another solution is to extend 2D filter to 3D. Paris et al. [34] extend the Gaussian kernel in bilateral filtering and mean-shift clustering to the temporal domain for video applications. Lang et al. [28] also extend the notion of smoothing to the temporal domain by exploiting optical flow and revisit optimization-based techniques such as motion estimation and colorization. These temporal smoothing and 3D extension methods are specific to their applications, and cannot generalize to other applications, such as stylization.

A more general solution considering temporal coherence is to incorporate a post-processing step which is blind to filters. Dong et al. [11] segment each frame into several regions and spatiotemporally adjust the enhancement (produced by unknown image filters) of regions of different frames; Bonneel et al. [4] filter videos along motion paths using a temporal edge-preserving filter. Unfortunately, these post-processing methods fracture texture patterns, or introduce ghosting artifacts when applied to the stylization results due to high demand of optical flow.

As for stylization, previous methods (including traditional ones [5, 20, 33, 44] and neural ones [1, 37]) rely on optical flow to track motions and keep coherence in color and texture patterns along motion trajectories. Nevertheless, how to add flow constraints to feed-forward stylization networks has not been investigated before.

2.3. Flow Estimation

Optical flow is known as an essential component in many video tasks. It has been studied for decades and numerous approaches have been proposed [23, 6, 42, 7, 43, 36]). These methods are all hand-crafted, and are therefore difficult to integrate and jointly train in our end-to-end network.

Recently, deep learning has been explored to solve optical flow. FlowNet [17] is the first deep CNN designed to directly estimate optical flow and achieve good results. Its successors have focused either on accelerating the flow estimation [35] or achieving better quality [24]. Zhu et al. [45] recently integrate the FlowNet [17] with image recognition networks and train the network end-to-end for fast video recognition. Our work is inspired by their idea of applying FlowNet to existing networks. However, stylization, unlike recognition, requires some new factors to be considered in the design of network, such as the loss function, and feature composition.

3. Method

3.1. Motivation

When applying stylization networks (*e.g.*, [25]) for consecutive frames independently, subtle changes in appearance (*e.g.*, lighting, noise, motion) often result in strong flickering, as shown in Figure 1. By contrast, in still-image style transfer, such small changes in the content image, especially on flat regions, may be necessary to generate spatially rich and varied stylized patterns, making the result more impressive. Thus, how to simultaneously preserve such rich textures and the temporal consistency in videos is worthy of more careful study.

For simplicity, we start by exploring temporal coherence between two frames. Our intuition is to warp the stylized result from the previous frame to the current one, and adaptively fuse both together. In other words, we want traceable points/regions to remain unchanged, while untraceable points/regions use new results at the current frame. Such an intuitive strategy not only makes stylized results along the motion paths as stable as possible, but also avoids ghosting artifacts for occlusions or motion discontinuities. The intuitive idea is shown in Figure 2.

In fact, the strategy outlined above consists of two subproblems: propagation and composition. Since propagation relies on good and robust motion estimation, instead of optical flow, we are more inclined to estimate flow on deep features [45], which may neglect noise and small appearance variations. To further avoid seam artifacts, composition is also considered in the feature domain.

In our system (shown in Figure 3), we adopt a recurrent neural network (RNN) architecture to obtain long-term consistency. By propagation in the RNN structure, the above temporal coherence between two frames can be further extended to long-term consistency. By doing so, we expect all traceable points to be propagated as far as possible in the entire video. Once the points are occluded or the tracking is lost, the composite features will keep values independently computed at the current frame.

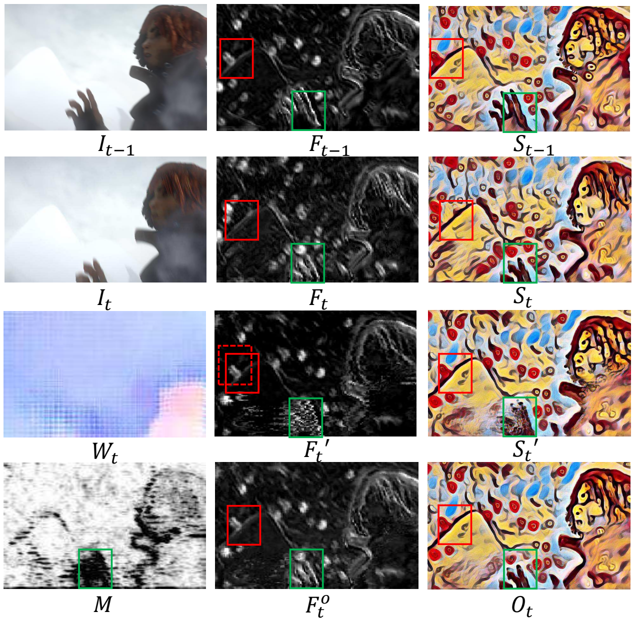


Figure 2. Visualization of two-frame temporal consistency. Two inputs I_{t-1}, I_t pass the stylization network [25] to obtain feature maps F_{t-1}, F_t, and stylized results S_{t-1}, S_t. We may notice discontinuities between S_{t-1} and S_t in the *red* and *green* rectangles. The third row shows the warped feature map F_t' and stylized result S_t' through flow W_t. We can see texture patterns (*red* rectangles) are successfully traced from $t-1$ to t, but ghosting occurs at the occluded regions (*green* rectangles) of S_t'. The occlusion mask is shown as M. In these false regions, F_t' (also S_t') is replaced with F_t (also S_t) to get the composite features F_t^o and result O_t.

3.2. Network Architecture

In this section, we explain the details of our proposed end-to-end network for video style transfer. Given the input video sequence $\{I_t|t = 1...n\}$, the task is to obtain the stylized video sequence $\{O_t|t = 1...n\}$. The overall system pipeline is shown in Figure 3. At the first frame I_1, it uses an existing stylization network (*e.g.*, [25]) denoted as Net_0 to produce the stylized result. Meanwhile, it also generates the encoded features F_1 as the input of our proposed network Net_1 at the second frame I_2. This process is iterated over the entire video sequence. Starting from the second frame I_2, we use Net_1 rather than Net_0 for style transfer.

Such a propagation structure follows a RNN structure. Since training traditional RNNs often suffers vanishing and exploding gradient problems [2], we instead only consider two-frame temporal consistency, presented in the network structure Net_1 (shown in Figure 4). It consists of three main components: the style sub-network, the flow sub-network, and the mask sub-network. At each time t, we always feed two input frames I_{t-1}, I_t to train the above sub-networks. The inference can achieve long-term consistency by propagating two-frame consistency. Specifically, at each time t, the composite feature maps F_t^o reuses F_{t-1}^o and combines

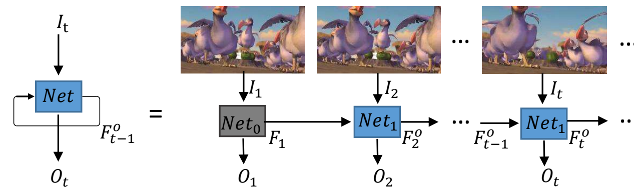


Figure 3. The overall system follows a RNN structure.

new information from I_t. To sum up, our RNN structure has two key differences compare to traditional RNNs. First, the hidden state F_t^o (composite feature maps) is defined by an explicit motion equation. Second, our training only considers two frames instead of more frames to reduce the difficulty of training.

Style Sub-network. We adopt the pre-trained image style transfer network of Johnson et al. [25] as our default style sub-network, since it is often adopted as the basic network structure for many follow-up works (*e.g.*, [12, 10]). This kind of network looks like auto-encoder architecture, with some strided convolution layers as the encoder and fractionally strided convolution layers as the decoder, respectively. Such architectures allow us to insert the flow sub-network and the mask sub-network between the encoder and the decoder. In Section 4.4, we provide a detailed analysis on which layer is better for the integration of our sub-networks.

Flow Sub-network. As a part of temporal coherence, the flow sub-network is designed to estimate the correspondences between two consecutive frames I_{t-1} and I_t, and then warp the convolutional features. We adopt *FlowNet* (the "Simple" version) [17] as our flow sub-network by default. It is pre-trained on the synthetic *Flying Chairs* dataset [17] for optical flow, and should be fine-tuned to produce feature flow suitable for our task.

The process is similar to [45], which uses it for video recognition. Two consecutive frames I_{t-1}, I_t are first encoded into feature maps F_{t-1}, F_t respectively by the encoder. W_t is the feature flow generated by the flow sub-network and bilinearly resized to the same spatial resolution as F_{t-1}. As the values of W_t are in general fractional, we warp F_{t-1} to F_t' via bilinear interpolation:

$$F_t' = \mathcal{W}_{t-1}^t(F_{t-1}) \tag{1}$$

where $\mathcal{W}_{t-1}^t(\cdot)$ denotes the function that warps features from $t-1$ to t using the estimated flow field W_t, namely $F_t'(p) = F_{t-1}(p + W_t(p))$, where p denotes spatial location in feature map and flow.

Mask Sub-network. Given the warped feature F_t' and the original feature F_t, the mask sub-network is employed to regress the composition mask M, which is then adopted to compose both features F_t' and F_t. The value of M varies

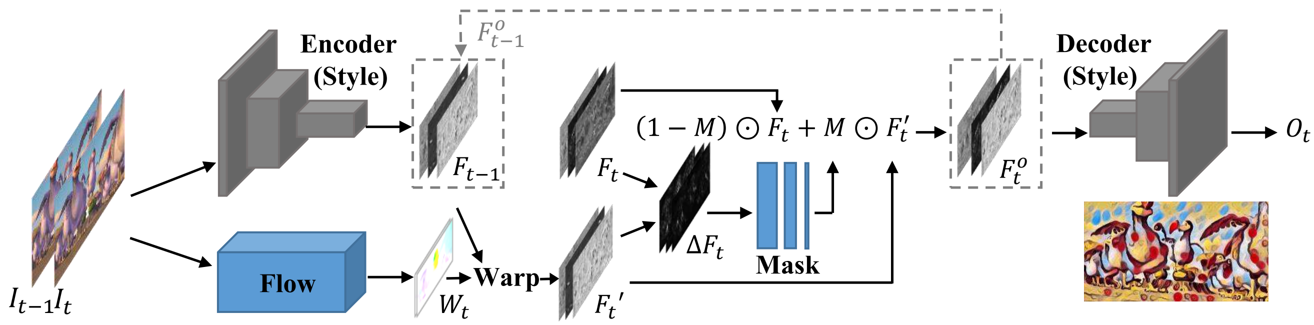


Figure 4. Our network architecture consists of three m components: the pretrained style sub-network, which is split into an encoder and a decoder; the flow sub-network to predict intermediate feature flow; and the mask sub-network to regress the composition mask.

from 0 to 1. For traceable points/regions by the flow (*e.g.*, static background), the value in the mask M tends to be 1. It suggests that the warped feature F'_t should be reused so as to keep coherence. On the contrary, at occlusion or false flow points/regions, the value in the mask M is 0, which suggests that F_t should be adopted. The mask sub-network architecture consists of three convolutional layers with stride one. Its input is the absolute difference of two feature maps

$$\Delta F_t = |F_t - F'_t|, \tag{2}$$

and the output is a single channel mask M, which means all feature channels share the same mask in the later composition. Here, we obtain the composite features F^o_t by linear combination of F_t and F'_t:

$$F^o_t = (1 - M) \odot F_t + M \odot F'_t \tag{3}$$

where $\odot$ represents element-wise multiplication.

Summary of $\mathbf{Net_1}$. Figure 4 summarizes our network Net_1 designed for two frames. Given two input frames I_{t-1}, I_t, they are fed into the encoder of a fixed style sub-network, generating convolutional feature maps F_{t-1}, F_t. This first step is different in inference, where F_{t-1} will not be computed from I_{t-1}, and instead borrowed from the obtained composite features F^o_{t-1} at $t-1$. It is illustrated by the dotted lines in Figure 4. On the other branch, both frames I_{t-1}, I_t are fed into the flow sub-network to compute feature flow W_t, which warps the features F_{t-1} (F^o_{t-1} used in inference instead) to F'_t. Next, the difference ΔF_t between F_t and F'_t is fed into the mask sub-network, generating the mask M. New features F^o_t are achieved by linear combination of F_t and F'_t weighted by mask M. Finally, F^o_t is fed into the decoder of the style sub-network, generating stylized result O_t at frame t. For inference, F^o_t is also the input for the next frame $t+1$. Since both flow and mask sub-networks only learn relative flow W_t and mask M_t between any two frames, it is not necessary for our training to incorporate historic information (*e.g.*, F^o_{t-1}) as well as the inference,,making our training quite simple.

3.3. The Loss Function

To train both the flow and mask sub-networks, we define the loss function by enforcing three terms: the coherence term $\mathcal{L}_{cohe}$, the occlusion term $\mathcal{L}_{occ}$, and the flow term $\mathcal{L}_{flow}$. The coherence term $\mathcal{L}_{cohe}$ penalizes the inconsistencies between stylized results of two consecutive frames.

$$\mathcal{L}_{cohe}(O_t, S_{t-1}) = M^g \odot ||O_t - \mathcal{W}^t_{t-1}(S_{t-1})||^2, \tag{4}$$

where S_{t-1} is the stylized result produced independently at $t-1$. The warping function $\mathcal{W}^t_{t-1}(\cdot)$ uses the ground-truth flow W^g_t. M^g is the ground-truth mask, where 1 represents consistent points/regions and 0 represents untraceable ones. It encourages the stylized result O_t to be consistent with S_{t-1} in traceable points/regions.

On the contrary, in the untraceable regions (e.g. occlusions), the occlusion term $\mathcal{L}_{occ}$ enforces O_t to be close to the independently stylized result S_t at frame I_t:

$$\mathcal{L}_{occ}(O_t, S_t) = (1 - M^g) \odot ||O_t - S_t||^2. \tag{5}$$

We add an additional term to constrain the feature flow:

$$\mathcal{L}_{flow} = ||W_t - W^g_t \downarrow ||^2. \tag{6}$$

Here we use the down-scaled version of the ground-truth optical flow $W^g_t \downarrow$, which is re-scaled to the same size of W_t, to serve as the guidance for feature flow estimation.

In summary, our loss function to train flow and mask sub-networks is the weighted average of three terms.

$$\mathcal{L} = \alpha \mathcal{L}_{cohe} + \beta \mathcal{L}_{occ} + \lambda \mathcal{L}_{flow}, \tag{7}$$

where $\alpha = 1e5$, $\beta = 2e4$ and $\lambda = 20$ by default.

Note that our loss function discards the content and style loss for training the original style network, because the pre-trained style sub-network is fixed during training. We believe that S_t (or S_{t-1}) itself can provide sufficient style supervision. One extra benefit is that we can directly leverage other trained still-image style models and apply it to videos directly. In this sense, our proposed framework is general.

4. Experiments

4.1. Dataset Set-up

Our task requires a big video dataset with varied types of motions and ground-truth optical flow. However, existing datasets are quite small, *e.g.*, the synthetic *MPI Sintel* dataset [8] (which only has 1,064 frames in total). Instead, we collect ten short videos (eight animated *Ice Age* movies, and two real videos from YouTube), around $28,000$ frames together as our training dataset.

To obtain approximated ground-truth flow W^g between every two consecutive frames in these videos, we use DeepFlow2 [43] to compute the bidirectional optical flow and use the backward flow as the ground-truth.

For the ground-truth of the composition mask M^g, we adopt the methods used in [37, 39] to detect occlusions and motion boundaries. We mask out two types of pixels, being set to 0 in M^g: 1) the occlusion pixels achieved by cross-checking the forward and backward flows; 2) the pixels at motion boundaries with large gradients of flow, which are often less accurate and may result in ghosting artifacts in composition. All other pixels in M^g are set to 1.

We use the *MPI Sintel* [8] as the test dataset, which is widely adopted for optical flow evaluation. It contains 23 short videos and is labeled with ground-truth flow and an occlusion mask. The dataset covers various types of real scenarios, such as large motions and motion blurs.

4.2. Implementation details

In our experiments, we adopt two types of pre-trained style networks (per-style-per-net [25][1], multiple-style-per-net [10][2]) as our fixed style sub-network. We train the flow sub-network and mask sub-network on the video dataset described in Section 4.1. All videos have an image resolution of 640×360. The network is trained with a batch size of 1 (frame pair) for 100k iterations. The Adam optimization method [26] is adopted with an initial learning rate of $1e-4$ and decayed by 0.8 at every $5k$ iterations.

4.3. Quantitative and Qualitative Comparison

For video style transfer, runtime and temporal consistency are two key criteria. Runtime uses the frame rate of inference. The temporal consistency is measured by

$$e_{stab}(O_t, O_{t-1}) = M^g \odot ||O_t - \mathcal{W}_{t-1}^t(O_{t-1})||^2, \quad (8)$$

where the stability error $e_{stab}(O_t, O_{t-1})$ measures the coherence loss (in Equation (4)) between two results O_t and O_{t-1}. Here, we only evaluate the stability of results on traceable regions. Lower stability error indicates more stable result. For the entire video, we use the average error instead.

[1]In our experiment, we adopt the released model of [25], but the channel number of all convolution layers is half of [25].

[2]We slightly modify the StyleBank model [10], whose encoder and decoder sub-networks adopt the same structures as [25], but the stylebank layer is inserted after the third residual block.

Methods	stability error e_{stab}				runtime
	Muse	*Candy*	*Scream*	*Udnie*	(fps)
Johnson et al. [25]	0.0199	0.0240	0.0048	0.0108	38.17
[25]+Ours	**0.0121**	**0.0105**	**0.0034**	**0.0076**	**15.07**
[25]+Ours††	0.0135	0.0120	0.0036	0.0079	15.07
Dong et al. [10]	0.0159	0.0181	0.0035	0.0059	20.6
[10] + Ours	**0.0126**	**0.0131**	**0.0030**	**0.0048**	**7.35**
Manuel et al. [37]	0.0063	0.0067	0.0019	0.0035	0.0089

Table 1. Comparison of different methods on stability error and runtime (GPU Titan X). Compared to the per-frame processing baseline [25] or [10], our method can obtain much lower stability loss while only $2.5 \sim 2.8\times$ slower. Compared to fixed flow sub-network (indicated by ††), our fine-tuned flow sub-network achieves better coherence. [35] is more stable but much slower.

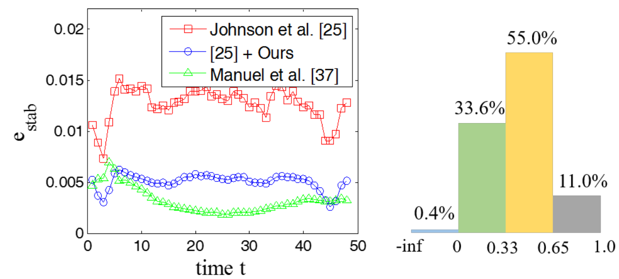


Figure 5. Left: Curve of e_{stab} over time of one example video in *Sintel* dataset. Right: The distribution of e_{stab} ratio between our method and baseline [25]

Quantitative Results. To validate the effectiveness of our method, we test and compare using two existing stylization networks [25, 10]. The baseline for comparison is to apply their networks to process each frame independently. As shown in Table 1, for all the four styles, our method obtains much lower stability error than the baseline [25, 10]. As for the runtime, our method is around $2.5 \sim 2.8\times$ slower than the baseline, due to our network requiring extra computation in both flow and mask sub-networks. Nevertheless, our method is still near real-time (15 fps in Titan X).

As a reference, we also compare our method with the optimization method [37]. Ours has larger temporal coherence errors compared to theirs, because our network is trained for all videos while theirs is optimized for just one. However, our method is thousands of times faster.

We further plot the curve of e_{stab} over time for each video sequence in the *MPI Sintel* dataset. Though these curves vary with the motions in different video sequences, ours is almost consistently better than the baseline [25](Figure 5 left). To prove it, we compute the e_{stab} ratio $((e_{stab}^{baseline} - e_{stab}^{our})/e_{stab}^{baseline})$ over all video frame pairs for the above 4 styles. Figure 5 right is the distribution percentages over four intervals, which shows our e_{stab} is better than the baseline for 99.6% frame pairs.

(a) (b) (c)

Figure 6. Comparison of our results (b) and results of [4] (c) on the same inputs (a). Their post-processing scheme results in ghosting and blurring artifacts for video style transfer.

$A+B$ with [25]	e_{stab}	$A+B$ with [10]	e_{stab}
Scream + *Scream*	0.0034	*Scream* + *Scream*	0.0031
Scream + *Candy*	0.0042	*Scream* + ***multiple***	0.0032
Candy + *Scream*	0.0137	*Candy* + *Candy*	0.0121
Candy + *Candy*	0.0105	*Candy* + ***multiple***	0.0123

Table 2. Cross comparison of transferring flow and mask sub-networks of *A* to *B*. *A* represents the style of pretrained style sub-network, and *B* is the style which flow and mask sub-networks are trained for. In [10], *B* can be multiple styles.

For visual quality, especially faithfulness to the original style, we conduct two user studies. We compare our method ([25] as the style sub-network) with the baseline [25] and optimization method [37] respectively. In each user study, we randomly select 5 videos for each of the 4 styles, then ask 20 participants to answer "Which is more faithful or equal in quality?". In the first user study, our method wins 23.1% of the time while baseline [25] wins 13.5% of the time, the remaining 63.4% results are equal. It indicates that our method can keep the faithfulness of the baseline in most cases. In the second user study, our method wins 49.2% of the time while [37] wins 34.6% of the time, the remaining 16.2% results are equal. The reason is that a few more users think our results retain more rich texture patterns from the input styles than [37].

Qualitative Results. In Figure 7, we show three examples with different kinds of representative motions to visually compare our results with per-frame processing models [10, 25]. These results clearly show that our methods successfully reduce temporal inconsistency artifacts which appear in these per-frame models. In the nearly static scene (*First Row*), ours can keep the scene unchanged after stylization while the per-frame models fail. As for scenes with motion, including both camera motions (*Second Row*) and object motions (*Third Row*), our method keeps the coherence between two frames except for occluded regions. More video results can be found on Youtube[3]).

We further compare our method with a post-processing method [4], which is applied to the per-frame stylized results of [25]. As shown in Figure 6, the results produced from the post-processing method [4] look blurry, and produce ghosting artifacts. This is because optimizing temporal coherence after stylization may not be able to obtain the global optima for both temporal coherence and stylization.

4.4. Ablation Study

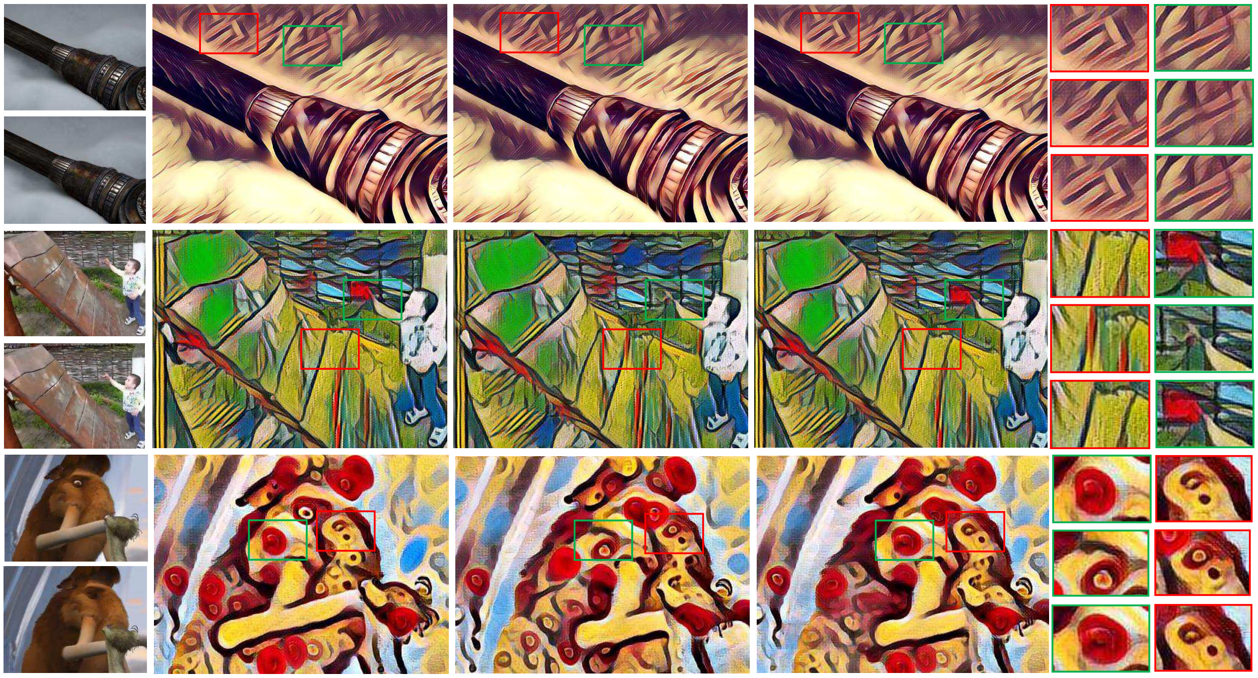
Layer Choice for Feature Composition. To study which layer of the style sub-network is the best for our feature propagation and composition, we try different layers for integration. For the basic style network [25], we find 5 intermediate feature layers from input to output (respectively with 1,1/2,1/4,1/2,1 times the original resolution), which allow our flow and mask sub-networks to be integrated. The five settings are trained and tested on the same database and with the same style.

In this experiment, we measure the sharpness of their stylization results by *Perceptual Sharpness Index* (PSI) [16], in addition to the stability error (Equation (8)). Table 3 clearly shows that stability improves from input to output layers, while sharpness decreases. This may result from the stylization networks (*e.g.*, [25]) amplifying image variances as shown in Figure 1. When feature flow estimation and composition happen closer to the input layer, small inconsistencies in composite features would also be amplified, causing incoherent results. When closer to the output layer, blending already amplified differences becomes more difficult and may introduce strong ghosting artifacts. To strike for a balance between stability and image sharpness, we recommend integrating our sub-networks into the middle layer of stylization networks, *i.e.*, r1/4(E). In this layer, the image content is compressed as much as possible, which may be beneficial for robust flow estimation and feature composition.

Fixed Flow Sub-network. In our experiment, *FlowNet* [17] is adopted as our flow sub-network. Original *Flownet* is trained in the image domain for optical flow. It needs to be fine-tuned for our task, since the flow would be further improved by jointly learning stylization and temporal coherence. Here, we compare fixed and fine-tuned flow sub-networks. As shown in Table 1, the fixed flow sub-network obtains less temporally coherent results than the fine-tuned one.

Transferability. To know whether our trained flow and mask sub-networks can be used for a new style (not appearing in training), we conduct two experiments on per-style-per-net [25] and multiple-style-per-net [10]. In per-style-per-net [25], we use two different styles, called A and B for cross experiments. One combination is a style sub-network learned from A, and our flow and mask sub-networks learned from B. The other combination is reversed. As shown in Table 2 (*First Column*), it is hard to preserve the original stability when our sub-networks trained on one style are applied to another. By contrast, in multiple-style-per-net [10], our trained sub-networks can be directly used to two new styles without re-training, while preserving the

[3]https://www.youtube.com/watch?v=vMyMUNvsGfQ



(a) (b) (c) (d) (e)

Figure 7. Qualitative comparison results: (a) Consecutive frames pair (top: frame t, bottom: frame $t+1$); (b) Stylization result of frame t; (c) Stylization result of frame $t+1$ from baseline; (d) Stylization result of frame $t+1$ from our method; (e) Top down: dilated marked regions corresponding to (b),(c),(d) respectively. The top row is with [10] for a nearly static scene, and the bottom two rows are with [25] for a scene with camera motion or object motion. Compared to baseline of [25, 10], our results are all more temporally coherent.

	Baseline	r1(E)	r1/2(E)	r1/4(E)	r1/2(D)	r1(D)
e_{stab}	0.0199	0.0187	0.0180	0.0121	0.0058	0.0038
PSI	0.4851	0.4846	0.4839	0.4825	0.4187	0.4086

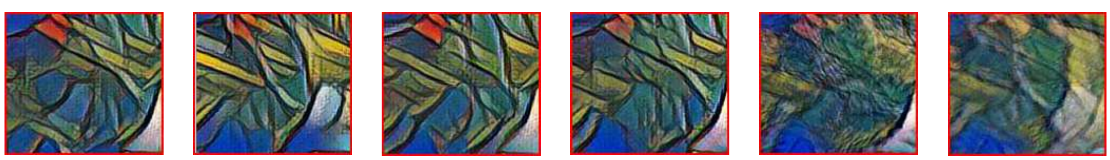


Table 3. Layer choice for feature composition. r1, r1/2, r1/4 represent different layers whose feature map resolution is $1, 1/2, 1/4\times$ of the original image, and E and D represent encoder and decoder respectively. The top table shows stability error e_{stab} and PSI for different settings. One visual example is shown on the bottom row.

original stability, as shown in Table 2 (*Second Column*). This suggests that our sub-networks learned with multiple-style-per-net [10] can be independent of styles, which is beneficial to real applications.

5. Conclusion and Discussion

In this paper, we propose the first video style transfer network by leveraging temporal information. It achieves near real-time speed on modern GPUs, which is thousands of times faster than optimization-based methods ([1, 37]). Moreover, our network achieves long-term temporal coherence through the propagation of short-term ones, which enables our model for online processing. It can be employed in existing stylization networks [25, 10], and even be directly used for new styles without re-training.

There are still some limitations in our method. For instance, limited by the accuracy of ground-truth optical flow (by DeepFlow2 [43]), our results may suffer from some incoherence where the motion is too large to track. And after propagation over a long period, small flow errors may accumulate, causing blurriness and inconsistencyenen. These open questions will require further exploration in future work.

Acknowledgement

This work is partially supported by the National Natural Science Foundation of China(NSFC, NO.61371192)

References

[1] A. G. Anderson, C. P. Berg, D. P. Mossing, and B. A. Olshausen. Deepmovie: Using optical flow and deep neural networks to stylize movies. *arXiv preprint arXiv:1605.08153*, 2016.

[2] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[3] N. Bonneel, K. Sunkavalli, S. Paris, and H. Pfister. Example-based video color grading. *ACM Trans. Graph.*, 32(4):39–1, 2013.

[4] N. Bonneel, J. Tompkin, K. Sunkavalli, D. Sun, S. Paris, and H. Pfister. Blind video temporal consistency. *ACM Trans. Graph. (Proc. of SIGGRAPH Asia)*, 34(6):196, 2015.

[5] A. Bousseau, F. Neyret, J. Thollot, and D. Salesin. Video watercolorization using bidirectional texture advection. In *ACM Trans. Graph. (Proc. of SIGGRAPH)*, volume 26, page 104. ACM, 2007.

[6] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proc. ECCV*, pages 25–36. Springer, 2004.

[7] T. Brox and J. Malik. Large displaczement optical flow: descriptor matching in variational motion estimation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(3):500–513, 2011.

[8] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *Proc. ECCV*, pages 611–625. Springer, 2012.

[9] A. J. Champandard. Semantic style transfer and turning two-bit doodles into fine artworks. *arXiv preprint arXiv:1603.01768*, 2016.

[10] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua. Stylebank: An explicit representation for neural image style transfer. In *Proc. CVPR*, 2017.

[11] X. Dong, B. Bonev, Y. Zhu, and A. L. Yuille. Region-based temporally consistent video post-processing. In *Proc. CVPR*, pages 714–722, 2015.

[12] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

[13] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proc. ACM SIGGRAPH*, pages 341–346. ACM, 2001.

[14] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proc. ICCV*, volume 2, pages 1033–1038. IEEE, 1999.

[15] J. L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

[16] C. Feichtenhofer, H. Fassold, and P. Schallauer. A perceptual image sharpness metric based on local edge gradient analysis. *IEEE Signal Processing Letters*, 20(4):379–382, 2013.

[17] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. *arXiv preprint arXiv:1504.06852*, 2015.

[18] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[19] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman. Controlling perceptual factors in neural style transfer. *arXiv preprint arXiv:1611.07865*, 2016.

[20] J. Hays and I. Essa. Image and video based painterly animation. In *Proc. of NPAR*, pages 113–120. ACM, 2004.

[21] A. Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proc. ACM SIGGRAPH*, pages 453–460. ACM, 1998.

[22] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proc. ACM SIGGRAPH*, pages 327–340. ACM, 2001.

[23] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[24] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *arXiv preprint arXiv:1612.01925*, 2016.

[25] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *arXiv preprint arXiv:1603.08155*, 2016.

[26] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[27] J. E. Kyprianidis, J. Collomosse, T. Wang, and T. Isenberg. State of the" art: A taxonomy of artistic stylization techniques for images and video. *IEEE Trans. on Visualization and Computer Graphics*, 19(5):866–885, 2013.

[28] M. Lang, O. Wang, T. Aydin, A. Smolic, and M. Gross. Practical temporal consistency for image-based graphics applications. *ACM Trans. Graph. (Proc. of SIGGRAPH)*, 31(4):34, 2012.

[29] C. Li and M. Wand. Combining markov random fields and convolutional neural networks for image synthesis. *arXiv preprint arXiv:1601.04589*, 2016.

[30] C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. *arXiv preprint arXiv:1604.04382*, 2016.

[31] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang. Diversified texture synthesis with feed-forward networks. In *Proc. CVPR*, 2017.

[32] P. Litwinowicz. Processing images and video for an impressionist effect. In *Proc. ACM SIGGRAPH*, pages 407–414. ACM, 1997.

[33] J. Lu, P. V. Sander, and A. Finkelstein. Interactive painterly stylization of images, videos and 3d animations. In *Proc. of I3D*, pages 127–134. ACM, 2010.

[34] S. Paris, S. W. Hasinoff, and J. Kautz. Local laplacian filters: Edge-aware image processing with a laplacian pyramid. *ACM Trans. Graph. (Proc. of SIGGRAPH)*, 30(4):68, 2011.

[35] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. *arXiv preprint arXiv:1611.00850*, 2016.

[36] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Proc. CVPR*, pages 1164–1172, 2015.

[37] M. Ruder, A. Dosovitskiy, and T. Brox. Artistic style transfer for videos. In *Proc. GCPR*, pages 26–36. Springer, 2016.

[38] A. Selim, M. Elgharib, and L. Doyle. Painting style transfer for head portraits using convolutional neural networks. *ACM Trans. Graph. (Proc. of SIGGRAPH)*, 35(4):129, 2016.

[39] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *Proc. ECCV*, pages 438–451. Springer, 2010.

[40] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. *arXiv preprint arXiv:1603.03417*, 2016.

[41] C.-M. Wang, Y.-H. Huang, and M.-L. Huang. An effective algorithm for image sequence color transfer. *Mathematical and Computer Modelling*, 44(7):608–627, 2006.

[42] J. Weickert, A. Bruhn, T. Brox, and N. Papenberg. A survey on variational optic flow methods for small displacements. In *Mathematical models for registration and applications to medical imaging*, pages 103–136. Springer, 2006.

[43] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *Proc. CVPR*, pages 1385–1392, 2013.

[44] S.-H. Zhang, X.-Y. Li, S.-M. Hu, and R. R. Martin. Online video stream abstraction and stylization. *IEEE Transactions on Multimedia*, 13(6):1286–1294, 2011.

[45] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei. Deep feature flow for video recognition. *arXiv preprint arXiv:1611.07715*, 2016.