Real-time Coherent Video Style Transfer
Final Report

Name: Gu Derun
UID: 3035140146

April 15, 2018
Abstract

Existing image style transfer models usually suffer from high temporal inconsistency when applied to videos. Some video style transfer models have been proposed to tackle this problem, yet they fail to guarantee high processing speed, desirable perceptual style quality and high temporal coherence at the same time. In this paper, we propose ReCoNet, a novel real-time video style transfer model, which can generate stylized videos in real-time speed with high temporal coherence while being able to reproduce rich artistic styles. In ReCoNet, a novel luminance warping constraint is added to the temporal loss at the output level to capture luminance changes between consecutive frames and increase stylization stability under illumination effects. We also purpose a new feature-map-level temporal loss to further improve temporal consistency on the appearances of traceable objects. Experimental results demonstrate that our model can achieve outstanding performance both qualitatively and quantitatively.

Keywords: video style transfer, real-time style transfer, temporal consistency, optical flow, luminance, deep learning, neural network
Acknowledgement

This final year project is supervised by professor Yizhou Yu. He provided great patience and kindness on helping us get familiar with related works and assured that the project was on track. Gao Chang and Zhang Fangjun are also the best teammates to work with.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Figures</td>
<td>5</td>
</tr>
<tr>
<td>List of Tables</td>
<td>6</td>
</tr>
<tr>
<td>List of Abbreviations</td>
<td>7</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>8</td>
</tr>
<tr>
<td>2 Literature Review</td>
<td>11</td>
</tr>
<tr>
<td>1 Image Style Transfer</td>
<td>11</td>
</tr>
<tr>
<td>2 Video Style Transfer</td>
<td>14</td>
</tr>
<tr>
<td>3 Methodology</td>
<td>18</td>
</tr>
<tr>
<td>1 Motivation</td>
<td>18</td>
</tr>
<tr>
<td>1.1 Luminance Difference</td>
<td>18</td>
</tr>
<tr>
<td>1.2 Feature-map-level Temporal Loss</td>
<td>20</td>
</tr>
<tr>
<td>2 Network Architecture</td>
<td>21</td>
</tr>
<tr>
<td>2.1 Encoder-Decoder Design</td>
<td>21</td>
</tr>
<tr>
<td>2.2 Implicit Usage of Optical Flow</td>
<td>22</td>
</tr>
<tr>
<td>2.3 Perceptual Loss Network</td>
<td>23</td>
</tr>
<tr>
<td>3 Loss Functions</td>
<td>23</td>
</tr>
</tbody>
</table>
6 Conclusion and Future Works

References
List of Figures

1.1 An example of temporal inconsistency ............... 9

2.1 Network proposed by Gatys for image style transfer . . 12
2.2 An example of image style transfer ...................... 13
2.3 Perceptual losses model for image style transfer ........ 14
2.4 Network structure proposed by Chen et al on video style transfer ......................................................... 15
2.5 Network proposed by Huang et al on video style transfer 16
2.6 Network proposed by Gupta et al on video style transfer 17

3.1 Histograms of temporal warping error in different datasets and color spaces. ...................................................... 20
3.2 The pipeline of ReCoNet ........................................ 21

4.1 Loss curves of the training process ...................... 28
4.2 Graphical User Interface of ReCoNet .................. 30

5.1 Video style transfer results using ReCoNet ............ 32
5.2 Visual effects compared with previous models ........ 35
5.3 User study full results ........................................ 36
5.4 Ablation study on luminance warping constraint .... 38
5.5 Effect of feature-map-level temporal loss ............. 41
List of Tables

3.1 Network layer specification ........................................... 22
5.1 Numerical comparison between different models ............ 33
5.2 User study aggregated results ................................. 37
5.3 Ablation study on multi-level temporal loss ............... 40
Abbreviations

ReCoNet  Real-time Coherent Video Style Transfer Network
GPU    Graphics Processing Unit
CNN    Convolution Neural Network
ReLU   Rectified Linear Unit
cuDNN  NVIDIA CUDA Deep Neural Network library
Chapter 1

Introduction

Recent image style transfer methods based on neural networks [1, 2] have achieved excellent visual results and real-time processing speed, i.e. above 24 frames per second (FPS). However, when applying those algorithms to a video frame by frame, the stylized output has very severe temporal inconsistency. This incoherent visual effect is usually shown as color flickering, inconsistent stylization of moving objects, etc [3]. Figure 1.1(a) and 1.1 (b) demonstrate such temporal inconsistency in video style transfer.

To reduce such effect, several video style transfer models have been proposed [3, 5, 6, 7, 8, 9] by using optical flows and occlusion masks explicitly or implicitly. Among them, Anderson et al [9] and Ruder et al [7] proposed temporal loss optimization methods guided by optical flows and occlusion masks. Although their methods can provide good coherence, it can take up to several minutes to process each video frame due to on-the-fly optimization in inference stage. Although Ruder et al later proposed an accelerated model [8], the processing speed (around 1 to 2 frames per second) is still far below the real-time standard. Later, some other researchers have proposed models [3, 5, 6] based on feed-forward networks to improve the speed of video style transfer, yet they failed to guarantee satisfying perceptual style quality and coherent stylization output at the same time.

Acknowledged the strengths and weakness of the above models, we propose ReCoNet, a real-time coherent video style transfer network, in our Final-year Project to incorporate rich perceptual style, good temporal coherence and fast processing speed at the same time. ReCoNet is a
Figure 1.1: Temporal inconsistency in video style transfer. (a) The style target *Edtaonisl* (Francis Picabia, 1913) and two consecutive video frames from Videvo.net [4] (b) Style transfer results by Chen et al [3] (c) Style transfer results by ReCoNet. The circled regions show that our model can better suppress temporal inconsistency, while Chen et al’s model generates inconsistent color and noticeable flickering effects.

We find that real-world videos and animations may not strictly follow the brightness constancy assumption [11] for optical flow estimation. This means that the same object may alter its luminance appearance due to illumination effects in consecutive frames, which cannot be taken into account by optical flow. In ReCoNet, we first propose a luminance warping constraint in the output-level temporal loss to specifically consider the luminance difference in the input video. Experiments in Section 4 shows that this constraint can improve stylizing stability in areas with illumination effects and help suppress overall temporal inconsistency.

From stylization results of previous methods [3, 5, 6, 7, 8], we have also observed inconsistent stylizing output for the same object in consecutive
frames. By checking the output of network middle layers, we found inconsistency in the high-level feature maps of the same object, which may because of the perturbation in surrounding background. Inspired by this, we apply a feature-map-level temporal loss to the output of the encoder in our network to further improve temporal consistency.

In summary, ReCoNet is a highly efficient feed-forward network that can generate stylized videos with high coherence and excellent perceptual styles. It also possesses the generality to reproduce various artistic styles on the diverse real-world videos.

In this report, related work for image and video style transfer will be first reviewed in Chapter 2. Detailed motivations, network architecture, and loss functions will be presented in Chapter 3. In Chapter 4, details of our implementation and the graphical user interface will be introduced. In Chapter 5, the experiment results will be reported and analyzed, where our model shows outstanding performance both qualitatively and quantitatively. The conclusion and future works are further discussed in Chapter 6.
Chapter 2

Literature Review

1 Image Style Transfer

Similar topics were first emerging as Non-photorealistic Rendering (NPR) since 1990s. Several works showed successes in rendering styles such as oil paintings [12, 13, 14], but such algorithms were difficult to generalized to the diverse artistic styles. The similar results were also achieved by another class of methods related to texture synthesis [15, 16, 17]. However, when encountered style targets with rich perceptual information and unique artistic styles, such method may not bring a visually satisfied quality.

Gatys et al [18] first developed an algorithm based on convolution neural network for image style transfer with any given specific target style. One of the biggest advantage of this method is that it gives a good visual quality on any provided style with no prior information. When given the content and style image, the network can refine a random noise to a stylized image iteratively as shown in Figure 2.1. The perceptual quality is also more desirable compared with previous methods.

The network makes use of a total loss function that contains a content loss and a style loss to guide the refinement of the result image.

\[ \mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style} \]

The content loss is defined as the \( L_2 \) distance between features of the
result image in the current iteration and the given content image.

\[ \mathcal{L}_{\text{content}} = \frac{1}{D} \| R^l - I^l \|^2 \]  

(2.1)

\( R^l \) and \( I^l \) represents the feature map at layer \( l \) in result image and content image respectively. The constant \( D = HWC \) is used for normalization, where \( H, W, C \) represent the height, width, and number of channels of the feature map in layer \( l \).

The style loss is defined as the Frobenius norm of the Gram matrix difference.

\[ \mathcal{L}_{\text{style}} = \frac{1}{G^2} \| G^l - A^l \|^2 \]  

(2.2)

\( G^l \) and \( A^l \) represent the Gram matrix of the result and style image. The definition of Gram matrix is as follows.

\[ G^l_{i,j} = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} F^l_{i,h,w} F^l_{j,h,w} \]  

(2.3)

\( G^l_{i,j} \) represents the inner product of \( i^{th} \) and \( j^{th} \) features in \( l^{th} \) layer of the result image, more formally, \( G^l_{i,j} = \sum_{k} F^l_{i,k} F^l_{j,k} \).
The author proposes that the similarity between styles can be indicated by the correlation similarity between features extracted from networks, and the Gram matrix is a good way to describe it mathematically.

The $\alpha$ and $\beta$ in the total loss function control the amount of information from the content and style images. For instance, when $\alpha/\beta$ increases, the image preserves more content details and learn less about the style. Empirically, the ratio of $\alpha$ and $\beta$ is set to $10^{-3}$ for good visual effect.

In each iteration, the network calculates loss values in the forward process, and then update the result image through backward propagation. This iteration process stops either after enough iterations or the content and style losses are both lower than a threshold.

This work achieves great success in terms of the result image quality. (see Fig. 2.2) But the drawback is also obvious. It is highly computational inefficient, requiring up to tens of seconds to compute a 720P image using normal commercial GPU [18].

Later many other neural network models [1, 2, 20, 21, 22, 23, 24, 25] were developed based on [18]. Among them, one of the most successful one was proposed by Johnson et al [1], using an end-to-end design in stylizing network and a pre-trained VGG network [10] as perceptual loss network (see Figure 2.3).

Different from the model proposed by Gatys et al [18] using random initialized image as the initial network input, in Johnson et al’s model, the content image becomes the input of the stylizing network and the output is directly the stylized image. Since this method does not need any iterative refinement, it is much faster and than algorithms based on on-the-fly optimization. In inference stage, it only takes one forward
pass to generate stylized image.

However, although the model proposed by Johnson et al. achieves both good perceptual quality and real-time inference speed, it can have severe flickering artifacts when applied to videos frame by frame.

## 2 Video Style Transfer

Afterwards, Anderson et al. [9] and Ruder et al. [7] introduced a temporal loss function to explicitly using optical flow to improve temporal consistency. This model makes use of optical flow and occlusion mask to get the translation of traceable pixels, and minimize the temporal loss function iteratively for each frame so that all the traceable pixels are consistent across frames. This method achieve good temporal consistency than the model proposed by Johnson, but it is quite slow due to iterative optimization. In inference stage, it usually takes up to 3 minutes for each frame, which is not feasible for real-time applications. Although Ruder et al. [8] later accelerated the model through several improvements, the speed is still far below the real-time standard.

Some efforts were made by researchers afterwards to improve the processing speed. Chen et al. [3] later proposed a feed-forward model based on the network proposed by Johnson et al. [1]. The stylizing network is fixed as the perceptual losses model, with an additional flow network and a mask network that needs to be trained. (see Fig. 2.4).
The flow network is used to generate optical flow between consecutive two frames, which indicates how each pixel in the first frame is moved to the second one. The generated optical flow is then down-sampled to the size of feature map at the last residual block. This optical flow will warp the feature map of the previous frame to the current frame. The mask network then generates occlusion masks which is either 0 or 1 describing whether each pixel in the first frame is occluded in the second frame or not. The occlusion mask, warped feature map, and the original feature map will together determine the final feature map using the following formula:

$$F_t^o = M \odot W_{t-1}^t(F_{t-1}^o) + (1 - M) \odot F_t$$  \hspace{1cm} (2.4)$$

Here, $F_t^o$ means the output feature map at time $t$, $M$ means the occlusion mask generated between time $t - 1$ and $t$, $W_{t-1}^t$ is the warping function determined by the forward optical flow between frame at time $t - 1$ and $t$. $\odot$ means element wise operation.

The final loss function contains the difference in the output image and estimated optical flow.

$$L_{total} = M^g \odot \|O_t - W_{t-1}^t(S_{t-1})\|^2 + (1 - M^g) \odot \|O_t - S_t\|^2 + \|W_t - W_t^g\|^2$$  \hspace{1cm} (2.5)$$

The $O_t$ is the output at time $t$ using the entire network including flow and mask networks, $S_t$ is the output at time $t$ only using stylizing network. $M^g$ is the ground truth occlusion mask, and $W_t^g$ is the ground truth optical flow at time $t$.

This work achieves acceptable coherence while being much faster than the model proposed by Ruder et al [8]. Nevertheless, due to the assembly
Figure 2.5: Network proposed by Huang et al. It only contains a stylizing network similar to the model proposed by Johnson et al. The optical flow is only used in the training stage to generate the temporal loss. It is not involved in the testing stage.

of large networks, it is still not efficient enough.

Another model focusing on video style transfer was proposed by Huang et al [6] around the same time of the model by Chen et al [3]. Instead of estimating optical flow by a subnetwork, this model only involves ground truth optical flow in the training process for loss calculation. The stylizing network is also slightly smaller than the model proposed by Johnson et al [1]. This makes it faster than the model proposed by Chen et al during inference stage. In the training of this network, there is also no explicit warping of feature maps. The temporal information is all included implicitly through back propagation of the temporal loss function. The function formula is listed as below.

\[
L_{\text{temporal}} = \| O_t - W_{t-1}^t (O_{t-1}) \|^2 
\]

(2.6)

where \( O_t \) is the stylized output and \( W_{t-1}^t \) is the warping function derived from forward optical flow.

Although this network can achieve good coherence and fast processing speed, it is not good enough to capture rich perceptual information due to its simple network structure and the requirement to optimize five loss functions at the same time. In order to maintain good temporal coherence, it sets a large loss weight ratio of temporal losses to perceptual losses, leading to a downgrade of the style.

In addition to the above two models [3, 6] aiming at real-time video style transfer, Gupta et al [5] also proposed a recurrent model guided by optical flow and occlusion mask with similar ideas (see Figure). In each iteration the network takes in the current video frame as well as the
previous stylized frame. The speed of this network can barely achieve real-time standard, but a very significant drawback is that it is difficult to parallelized since the next iteration must wait for the result previous iteration. The main network structure of this model adopts a similar one with model proposed by Johnson et al [1], and the loss function is exactly the same as temporal loss in [6]. Therefore in the later experiment section, our model is only compared with models proposed by Chen et al and Huang et al.
Chapter 3

Methodology

This chapter will discuss the methodology used in this final year project. The motivation of the modifications adopted in this project will be first introduced, followed by network architecture and loss functions.

1 Motivation

In this project, two key modifications are proposed to improve the video style transfer quality. The first one is to add a novel luminance warping constraint to the existing output level temporal loss. The second one is to add an additional feature-map-level temporal loss to improve consistency of traceable pixels. This section will mainly discuss how those two modifications are formed from sketch.

1.1 Luminance Difference

Real-world videos usually possess illumination effects to some extent. This includes but not limited to changes in light sources intensity, direction and color, changes in surface property due to moving camera, changes in shading caused by occlusion, etc. However, when calculating optical flow for such videos, many algorithms [26, 27] adopts the brightness constancy assumption [11] on image data. Therefore, when using optical flow to explicitly warp the image, it commonly exists an error map on traceable pixels. The same situation also applies to an-
In animations, many datasets, e.g. MPI Sintel Dataset [28] use the albedo pass to calculate ground-truth optical flows first, while adding illuminations including smooth shading and specular reflections to generate the final image frames. This leads to a slight mismatch between optical flow and the corresponding image sequences. As a direct result, differences on luminance and chromaticity are commonly existed in these animation datasets.

To further examine the illumination difference, we computed the absolute value of temporal warping error over MPI Sintel Dataset and 50 real-world videos download from Videvo.net [4]. The formula to calculate temporal warping error is

\[ e_{\text{warping}} = \| I_t - W_{t-1}^t(I_{t-1}) \| \]

where \( W_{t-1}^t \) is warping function determined by the forward optical flow and \( I_t \) is the input image frame at time \( t \). We used FlowNet2 [29] to first generate optical flows for downloaded videos. The occlusion masks are then obtained based on the estimated optical flows using method of Sundaram et al [30] with equations:

\[ |w + \hat{w}|^2 < 0.01(|w|^2 + |\hat{w}|^2) + 0.5 \]  \hspace{1cm} (3.1)

\[ |\nabla u|^2 + |\nabla v|^2 > 0.01|w|^2 + 0.002 \]  \hspace{1cm} (3.2)

The occlusion masks contains two parts: the occlusion regions and motion boundaries. The equation 3.1 is to detect occlusion regions, where \( w = (u(x, y), v(x, y)) \) is the forward optical flow from \( t \) to \( t+1 \) for each position \((x, y)\) and \( \hat{w} = (\hat{u}(x, y), \hat{v}(x, y)) \) is the backward optical flow from \( t+1 \) to \( t \) for each position \((x, y) + (u(x, y), v(x, y))\). When a pixel was not occluded in the second image, then \( w + \hat{w} \) should be very close to zero, and vise versa. The equation 3.2 is to detect motion boundaries. When the optical flow has a sudden “jump” in any of the \( u \) and \( v \), then it is very likely to be on motion boundaries.

The experimental results on temporal error map is demonstrated in histograms in Figure 3.1. We can draw two conclusions based on the results. First, RGB channels share similar warping error distributions with no bias of changes in each color channel. (Referenced as “no bias” assumption below.) Second, despite changes in relative luminance channel \( Y \), the chromaticity channels \( X \) and \( Z \) in XYZ color space also contribute to
Based on this finding, a novel luminance constraint is proposed in our temporal loss. Unlike the temporal loss used by previous methods [7, 8, 3, 6, 5] that assume the temporal warping difference to be zero (see Equation 2.6), this constraint tend to encourage the stylized frames and the input frames to share the same luminance changes. This modification can make training easier and reduce unstable color changes under illumination effects. Experiments in Chapter 5 show that this new constraint can bring significant improvements to the stability of the stylized videos while maintaining excellent perceptual quality.

1.2 Feature-map-level Temporal Loss

Besides luminance warping constraint, this project also proposes a new loss function for feature-map-level consistency. The inspiration is from the intuition that high-level representations of the same object should be similar for consecutive images. A similar idea is also mentioned by Girshick in [31]. Therefore, although warping frames directly at the output level may not be accurate as examined in the previous subsection, the same method can be quite useful on feature-map level since the aggregated information usually suffers less from different kinds of noises compared with RGB images [3]. Based on this belief, a feature-map level temporal loss is also added to the final loss function. We use bilinear sampling to down-sample the ground-truth optical flows and occlusion masks to calculate the feature-map differences between the warped feature maps and the current ones. Experiments in Chapter
Figure 3.2: The pipeline of ReCoNet. $I_t$, $F_t$, $O_t$ denote the input image, encoded feature maps, and stylized output image at time frame $t$. $M_t$ and $W_t$ denote the occlusion mask and the optical flow between frames at $t - 1$ and $t$. Style denotes the artistic style image. The dashed box represents the prediction results of the previous frame, which will only be used in the training process. Red arrows and texts denote loss functions.

5 show that this new loss can further improve the stability of object appearances in consecutive frames in addition to the involvement of luminance warping constraint.

2 Network Architecture

ReCoNet adopts a pure CNN-based design that enables it to process input image extremely fast at inference stage. The temporal information is also utilized implicitly in the training process at different levels. The whole pipeline of ReCoNet is shown in Figure 3.2.

2.1 Encoder-Decoder Design

The stylization network adopts a similar structure as the model proposed by Johnson et al [1]. Further more, it is divided to an encoder and a decoder for different purposes. The encoder contains three convolution layers and four residual blocks [32] to encode the input video frame to high-level representations with aggregated information. This information is used to compute feature-map level temporal loss. The decoder contains two up-sampling convolution layers and one final convolution
<table>
<thead>
<tr>
<th>Layer</th>
<th>Layer Size</th>
<th>Stride</th>
<th>Output Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoder</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conv + InsNorm + ReLU</td>
<td>$48 \times 9 \times 9$</td>
<td>1</td>
<td>$48 \times 640 \times 360$</td>
</tr>
<tr>
<td>Conv + InsNorm + ReLU</td>
<td>$96 \times 3 \times 3$</td>
<td>2</td>
<td>$96 \times 320 \times 180$</td>
</tr>
<tr>
<td>Conv + InsNorm + ReLU</td>
<td>$192 \times 3 \times 3$</td>
<td>2</td>
<td>$192 \times 160 \times 90$</td>
</tr>
<tr>
<td>(Res + InsNorm + ReLU ) ×4</td>
<td>$192 \times 3 \times 3$</td>
<td>1</td>
<td>$192 \times 160 \times 90$</td>
</tr>
<tr>
<td>Decoder</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Up-sample</td>
<td></td>
<td>1/2</td>
<td>$192 \times 320 \times 180$</td>
</tr>
<tr>
<td>Conv + InsNorm + ReLU</td>
<td>$96 \times 3 \times 3$</td>
<td>1</td>
<td>$96 \times 640 \times 360$</td>
</tr>
<tr>
<td>Up-sample</td>
<td></td>
<td>1/2</td>
<td>$96 \times 640 \times 360$</td>
</tr>
<tr>
<td>Conv + InsNorm + ReLU</td>
<td>$48 \times 3 \times 3$</td>
<td>1</td>
<td>$48 \times 640 \times 360$</td>
</tr>
<tr>
<td>Conv + Tanh</td>
<td>$3 \times 9 \times 9$</td>
<td>1</td>
<td>$3 \times 640 \times 360$</td>
</tr>
</tbody>
</table>

Table 3.1: Network layer specification. Layer and output sizes are denoted as channel × height × width. Conv, Res, InsNorm, ReLU, Tanh denote convolutional layer, residual block [32], instance normalization layer [34], ReLU activation layer [35], and Tanh activation layer respectively.

There are also several specific optimization that can be beneficial to the output video quality. We replaced the traditional deconvolution layers in the decoder to avoid checkerboard artifacts [33]. At the same time, instance normalization [34] is also added after each convolution layer but before activation layer for better stylization quality. Reflection padding is used at each convolution layer.

### 2.2 Implicit Usage of Optical Flow

Unlike the model proposed by Chen et al [3], ReCoNet does not use a separate network to estimate optical flow for explicit warping on feature maps. Instead, it use the optical flow information implicitly in the multi-level loss function to ensure good coherence of both high-level feature maps and the output stylized images.
2.3 Perceptual Loss Network

The loss network is a VGG-16 network [10] pre-trained on the ImageNet dataset [36]. In each iteration, the loss network takes in the input video frame, stylized output from decoder, and the style target in three independent passes. The features of those three images are then extracted from the middle layer of the VGG-16 network. The content and style losses are then computed based on the extracted features.

3 Loss Functions

Our loss function is composed by a multi-level temporal loss function and a perceptual loss function. The multi-level loss function considers both feature-map level temporal differences and the output level temporal differences. At the feature-map level, a strict optical flow warping is adopted to minimize the feature map difference between same objects in consecutive frames. At the output level, a luminance warping constraints is added to the optical flow warping result to simulate real-world luminance changes. The perceptual losses design shares the same components as in perceptual losses model [1].

In the training stage, a two-frame synergic training mechanism [6] is used for loss calculation. In each iteration, the network processes the first video frame and the second video frame in two runs. The temporal losses are computed using the feature maps and stylized output of both frames, and the perceptual losses are computed using each output of the loss network independently and then added together. In the inference stage, the process is just like a normal network with only one video frame as input and its stylized result as a direct output.

3.1 Output-level Temporal Loss

Luminance changes is very common in various videos. However, previous models [3, 5, 6, 7, 8] usually do not take this into account. In ReCoNet, we consider the relative luminance $Y = 0.2126R + 0.7152G + 0.0722B$, same as $Y$ in XYZ color space, as a good capture of the luminance difference. This constraint is added equally to all channels in RGB color.
space based on the “no bias” assumption in Section 1.1 of this chapter:

\[
\mathcal{L}_{temp,o}(t-1, t) = \sum_c \frac{1}{D} M_t \| (O_t - W_t(O_{t-1}))_c - (I_t - W_t(I_{t-1}))_Y \|^2 \tag{3.3}
\]

In this formula, \( c \in [R, G, B] \) is each of the RGB channels of the image, \( Y \) is the relative luminance channel, \( I_t \) and \( O_t \) are the input video frame and stylized output at time \( t \), \( W_t \) and \( M_t \) are the ground-truth forward optical flow and the binary occlusion mask, \( D = H \times W \) is the normalization term with \( H \) and \( W \) the height of the video frame. The ablation study in Chapter 5 further discusses different choices of the luminance constraint and the output-level temporal loss.

### 3.2 Feature-map-level Temporal Loss

Small variations can appear in feature maps of the middle layers before reaching the final stylized output. The feature-map-level temporal loss is then designed to enforce temporal coherence in the output of encoder. The exact loss function is

\[
\mathcal{L}_{temp,f}(t-1, t) = \frac{1}{D} M_t \| F_t - W_t(F_{t-1}) \|^2 \tag{3.4}
\]

where \( F_t \) is the feature maps generated by the encoder at time \( t \), \( W_t \) and \( M_t \) the down-sampled ground-truth forward optical flow and occlusion mask, \( D = C \times H \times W \) is the normalization term with \( C, H, W \) as the height width and channel number of the feature map.

Experimental results in Chapter 5 justifies the effect of the feature-map-level temporal loss in improving temporal coherence.

### 3.3 Perceptual Losses

The perceptual losses contains content loss \( \mathcal{L}_{content}(t) \), style loss \( \mathcal{L}_{style}(t) \) as well as total variation loss \( \mathcal{L}_{tv}(t) \). The loss functions are adopted from the model proposed by Johnson et al [1]. The content loss (see Equation 2.1) utilizes feature maps at \( relu3\_3 \) layer of the VGG-16 [10] and the style loss (see Equation 2.2) utilizes feature maps at \( relu1\_2\),
relu2_2, relu3_3, relu4_3]. The total variation loss takes the form as

\[ \mathcal{L}_{tv}(t) = \sum_{i,j} (\|O_{i+1,j}^t - O_{i,j}^t\|^2 + \|O_{i,j+1}^t - O_{i,j}^t\|^2)^{1/2} \] (3.5)

where \(O_{i,j}^t\) is the pixel with coordinate \((i, j)\) in the output frame at time \(t\). This loss is to make the output image more smooth without large contrast between horizontal and vertical connected pixels.

### 3.4 Summary

By combining the loss functions discussed in the above sections, the total loss function in the two-frame synergic training is:

\[
\mathcal{L}(t-1, t) = \sum_{i \in \{t-1, t\}} (\alpha \mathcal{L}_{content}(i) + \beta \mathcal{L}_{style}(i) + \gamma \mathcal{L}_{tv}(i)) \\
+ \lambda_f \mathcal{L}_{temp,f}(t - 1, t) + \lambda_o \mathcal{L}_{temp,o}(t - 1, t) \] (3.6)

where \(\alpha, \beta, \gamma, \lambda_f\) and \(\lambda_o\) are hyper-parameters for the training process.
Chapter 4

Implementation Details

This chapter mainly discuss the implementation details of this project and some important tricks that help to realize the methodology discussed in Chapter 3.

1 Dataset Preparation

The datasets preparation is divided into two parts including input images for network training and the style targets. This section will introduce these two parts separately in details.

1.1 Content Images

The datasets containing content images used in this project are MPI Sintel [28], FlyingThings3D and Monkaa [37]. The Sintel dataset is generated from a ten-minute animation, which represents animated movies. This is also used as our testing dataset. In training process, Monkaa dataset is also used to simulate animations. Compared with Sintel dataset containing around 1000 images, Monkaa dataset contains more than 8640 images with both small and large motions between frames. This makes it a very good dataset for network training purpose. Besides animations, FlyingThings3D is also used as a complementary of Monkaa to simulate real-world scenes. It is a dataset containing objects moving in front of rotated real-world backgrounds along random
3D trajectories. This dataset is very large containing more than 20 thousands, which is very good for preventing overfitting. Combine both datasets, ReCoNet obtains a good stylization quality in both animation and real-world videos as demonstrated in Chapter 5.

The ground-truth optical flows and motion boundaries are provided within the dataset. Based on them, occlusion masks can be obtained by using Equation 3.1 and Equation 3.2 for both training and testing datasets. During the training process, We use all the data in Monkaa and FlyingThings3D datasets to make the training data as much as possible to avoid overfitting. Consecutive frames are paired up for the two-frame synergic training [3] and horizontal flip is acted randomly on each pair. The resulting pairs are also shuffled before training the network. During evaluation, we use MPI Sintel dataset to verify the temporal consistency and perceptual styles of our stylization results. This setting is the same as most previous works [3, 5, 6], which allows easy model comparison.

1.2 Style Images

The common styles used in previous works [7, 8, 3, 5, 38] includes Candy, Udnie, Mosaic, Dream, Composition, Girl, Starry Night (Vincent van Gogh, 1889), Autoportrait (Picasso, 1907) and Edtaonisl (Francis Picabia, 1913). Especially, the numerical results were all generated using Candy. In this project, Udnie is first used to test the network design due its simple perceptual strokes. Later, numerical results are computed using Candy in order to compare with previous works. More styles are tested later to check the generalization ability of the final model.

2 Environment Setup

The machine learning framework used in this project is PyTorch 0.3 [39] with cuDNN 7 [40] as the main machine learning library. Since PyTorch has a set of comprehensive useful functions as well as easy grammar, it can greatly facilitate the development efficiency. This was highly helpful for the various experiments in the later evaluation stage.

The graphics card used in this project is Nvidia GTX 1080Ti, which is
the latest publicly available commercial graphics card on market for research purpose. The development system is Linux Desktop 16.04, which is also commonly used in research due to its good compatibility with multiple software including PyTorch. Besides, Git is used for version control in this project.

### 3 Network Hyper-parameters

Hyper-parameters including learning rate, learning rate decay, loss weight, regularization, number of epochs and many others can have a great influence on the final performance. This section mainly presents the key hyper-parameter settings of this project that can have a significant impact on the final model quality.

Figure 4.1 gives a visualized result of the final loss curves shapes for the training process of the style Edtaonisl.
3.1 Learning Rate and Learning Rate Decay

The initial learning rate is set to $1e^{-3}$ at this stage, and the optimizer is chosen as Adam [41]. Usually, an additional parameter learning rate decay is needed to avoid gradient explosion. It usually decrease the learning rate $l = l \cdot e^r$, where $r = \lfloor \frac{N}{epoch\_size} \rfloor$ in an exponential manner. However, by using Adam optimizer in ReCoNet, a decay is provided by the optimizer itself for every single parameter in the network. Therefore, no additional learning rate decay is set for this project.

3.2 Loss Weight

We use a high style to content loss weight ratio chosen among the previous works [3, 6, 5] to ensure our model can eventually learn a good perceptual style. Since the temporal loss value is usually very small after normalized to a pixel level, we use a large weight ratio to adjust the temporal loss in the same level of perceptual losses. The default training hyper-parameters are finalized to be $\alpha = 1, \beta = 10, \gamma = 10^{-3}, \lambda_f = 10^7, \lambda_o = 2 \times 10^3$ in Equation 3.6.

3.3 Regularization and Activation

Regularization is a term usually added to the loss function to avoid overfitting. It is defined as $\sum ||W||_p$, which is the $L_p$ norm for all layers parameters $W$. Usually, $p$ is set to 1 or 2, and $L_1$ norm is more common since it can encourage the network parameters to be sparse and reduce size of parameter space.

However, existing research works have found that ReLU activation [35] function also has the advantage to make the network sparse and avoid overfitting compared with other activation functions. Another advantage of it is that it can also avoid gradient vanish and gradient explosion. In this project, ReLU is used as the activation function for all layers besides the last layer. Since the final output image should have a upper and lower bound in pixel values, we use sigmoid activation function to obtain a value between 0 and 1 and later rescale it to $0 \sim 255$. No regularization is used in this project at this moment.
3.4 Batch Size and Epoch Number

The batch size used for all training processes are set to 2. We train the model for around 30000 steps, which is roughly two epochs using all images in Monkaa and FlyingThings3D datasets [37]. Since the performance improvement after certain iterations become smaller and smaller, which is not worth for the time consumed, we finalized the training iterations as the current setting considering the trade-off between model quality and consuming time.

4 Graphical User Interface

For easier usage of our model, we implemented a GUI (see Figure ) based on our finalized version. The video can be captured by an external camera and is extracted using OpenCV-Python package. The graphical interface is drawn using PyQt package, which also helps to show the stylized video output. Currently six representative styles including Autoportrait, Candy, Composition, Edtaonisl, Girl and Udnie are included in the desktop application.
Chapter 5

Experiments and Results

In this chapter, we will first validate the generalization capacity of our model on real-world videos with different styles. Then the comparison between ReCoNet and the previous models are illustrated in details. Both quantitative and qualitative results are presented in this section.

1 Generalization Capacity

Since both training and testing dataset uses images of unrealistic scenes, i.e. animation scenes in MPI Sintel [28] and Monkaa [37], and computer generated random scenes in FlyingThings3D [37], the stylization quality on real-world videos is not considered before the experiments. Therefore, in this section, we demonstrate the designed experiment in testing the generalization ability of ReCoNet.

Before the experiment, 5 real-world videos from Videvo.net [4] are downloaded randomly within each of the all 10 different categories. A demonstration of the data is shown in Figure 5.1. There are four different styles Mosaic, Dream, Autoportrait, and Candy. For each of the style, there are three consecutive video frames and its stylized results. From the figure, it can be observed that ReCoNet is able to reproduce diverse color, strokes and textures of different style target on randomly chosen real-world scenes. At the same time, the stylized frames are visually coherent as well.
Figure 5.1: Video style transfer results using ReCoNet. The first column contains consecutive image frames in videos downloaded from Videvo.net [4]. Each video frames are followed by the style targets and their corresponding stylized results.
2 Comparison to Methods in the Literature

In this section, comparisons between ReCoNet and models proposed by Chen et al [3] and Huang et al [6] will be introduced. The quantitative analysis analyze the temporal consistency and inference speed of different models, in which ReCoNet shows the best inference speed while maintaining good temporal coherence.

2.1 Quantitative Analysis

Table 5.1 shows the temporal error $e_{stab}$ of four video style transfer models on five scenes in MPI Sintel Dataset using Candy as the style target. $e_{stab}$ is the square root of output-level temporal error (Equation 3.3) over one whole scene:

$$e_{stab} = \sqrt{\frac{1}{T-1} \sum_{t=1}^{T} \frac{1}{D} M_t \|O_t - W_t(O_{t-1})\|^2} \quad (5.1)$$

where $T$ is the total number of frames, and other variables are identical to Equation 3.3. This error function demonstrated the average temporal consistency of the stylized image across all scenes. The test images are resized to $640 \times 360$, and all experiments are running on a single Nvidia GTX1080 Ti GPU.

From the table, Ruder et al [7] is not able to use in real-time scenarios due to its slow inference speed, although it has the best temporal stability. On the contrary, ReCoNet and Huang et al.’s model achieve far better inference speed than the others. Compared with model proposed

<table>
<thead>
<tr>
<th>Model</th>
<th>Alley-2</th>
<th>Ambush-5</th>
<th>Bandage-2</th>
<th>Market-6</th>
<th>Temple-2</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chen et al [3]</td>
<td>0.0934</td>
<td>0.1352</td>
<td>0.0715</td>
<td>0.1030</td>
<td>0.1094</td>
<td>22.5</td>
</tr>
<tr>
<td>ReCoNet</td>
<td>0.0846</td>
<td>0.0819</td>
<td>0.0662</td>
<td>0.0862</td>
<td>0.0831</td>
<td>235.3</td>
</tr>
<tr>
<td>Huang et al [6]</td>
<td>0.0439</td>
<td>0.0675</td>
<td>0.0304</td>
<td>0.0553</td>
<td>0.0513</td>
<td>216.8</td>
</tr>
<tr>
<td>Ruder et al [7]</td>
<td>0.0252</td>
<td>0.0512</td>
<td>0.0195</td>
<td>0.0407</td>
<td>0.0361</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Table 5.1: Five scenes from MPI Sintel Dataset are selected for validation for temporal consistency using $e_{stab}$. The average FPS in the inference stage is also shown in the last column of the table. All the evaluations are made with style Candy.
by Chen et al [3] and other recurrent models [5, 8], the feed-forward model design adopted by ReCoNet has the advantage that the current iteration is independent of previous frames and does not need to wait for any of the previous frame to be fully processed. This makes it very easy to be accelerated with parallelism by cuDNN [40] library on graphics card.

Further considering the rest models that reach the real-time standard (around or above 24 FPS), it can be observed that ReCoNet achieves lower temporal error than the model proposed by Chen et al [3] due to the introduction of the multi-level temporal loss. Although ReCoNet has a slightly lower numerical measurements on stability compared with model proposed by Huang et al [6], ReCoNet shows strong capacity to reproduce strokes and minor textures of a style compared with Huang’s model. This is further shown in details in the following qualitative analysis.

2.2 Qualitative Analysis

We examine our style transfer results qualitatively with the models proposed by Chen et al [3] and Huang et al [6].

2.2.1 Analysis on Stylized Images

Figure 5.2(a) shows the stylization comparison between ReCoNet and model proposed by Huang et al [6]. Although Huang et al’s model can learn the color information in the style target, it fails to capture much about the perceptual textures and strokes. Based on the figure, two possible reasons can be drawn to explain the weak perceptual style of model proposed by Huang et al.

- First, they use a low weight ratio between perceptual losses and temporal loss to maintain temporal coherence, which brings obvious reduction to the quality of output style. However, ReCoNet uses a multi-level temporal losses with luminance warping constraints on the output-level temporal loss, which makes the whole network easy to be trained. This design leaves room for a larger loss weight ratio between perceptual losses to temporal losses while
maintaining good temporal coherence. As shown in the first example in Figure 5.2, compared with result of Huang’s model in Composition style, the stylized image generated by ReCoNet successfully reproduces the distinctive color blocks, especially on the rugged sand surfaces and the rolling sea waves.

- Second, the model proposed by Huang et al uses feature maps at a very high level (relu4_2 layer) in the VGG-16 [10] loss network to calculate the content loss. This makes it difficult to capture low-level features such as edges. As shown in the second example, although sharp bold contours are characteristic in the Girl style image, their model fails to clearly reproduce such style.
Different from the model proposed by Huang et al., the model proposed by Chen et al. can preserve perceptual style information to a large extent as shown in Figure 1.1 and 5.2(b). However, from the figure, there is also obvious inconsistency in their stylized results, leading to unpleasant visual flickering. This coincides with its relatively high temporal errors of their model in the quantitative comparison compared with ReCoNet.

2.2.2 User Study

Besides analysis on images, we conducted a user study as well between ReCoNet and models proposed by Chen et al. and Huang et al. This helps to compare the style transfer quality on a whole video level. The comparisons are made one by one for ReCoNet versus model proposed by Chen et al. and ReCoNet versus model proposed by Huang et al. In each of the comparison, 4 different styles on 4 different random video clips downloaded from Videvo.net [4] are used. There are in total 50
people to answer questions regarding all 8 videos in both comparison. There are three questions designed to examine the stylization quality of each video in terms of temporal coherence, perceptual style quality and overall effect. The exact description of the questions are as below.

- (Q1) which model perceptually resembles the style image more, regarding the color, strokes, textures, and other visual patterns;
- (Q2) which model is more temporally consistent such as fewer flickering artifacts and consistent color and style of the same object;
- (Q3) which model is preferable overall.

Figure 5.3 gives the detailed user study results. A summarization of each comparison can be found in Table 5.2. Compared with model proposed by Chen et al, our model can achieve more excellent temporal coherence while maintaining similar perceptual style. In the second comparison with model proposed by Huang et al, we can achieve much better style quality with only slightly worse temporal coherence. This is due to the involvement of multi-level temporal loss as demonstrated in Chapter 3. The result shows that ReCoNet has the ability to capture main style information such as color, strokes, textures, and visual patterns, while maintaining good temporal coherence. At the same time, this also validates again its good generalization ability on real-world videos.

### 3 Ablation Study

In this section, two sets of experiments are implemented and discussed to validate the effect of the feature-map-level temporal loss and the lumi-
Figure 5.4: Style transfer results using three different approaches described in Section 3. The style target is Candy, and the validation scenes are same as Table ?? for temporal error $e_{stab}$ calculation. The total and the luminance-wise temporal error maps show the absolute value of temporal errors in all color channels and in the relative luminance channel respectively.

nance warping constraint on the output-level-temporal loss. The results shows that the luminance warping constraints can bring significant improvement to the video stability while feature-map-level temporal loss also helps to reduce incoherence of object appearance.

### 3.1 Different Luminance Warping Constraints

The luminance difference at the output level can be taken into account in several different ways. To further examine this, three experiments are implemented:

1. Same relative luminance warping constraint on each RGB channel (Formula 3.3);

2. First change the color space of output into XYZ color space, and then add a relative luminance warping constraint to Y channel:

   $$
   \mathcal{L}_{temp}^o = \frac{1}{D} M_t(||(O_t - W_t(O_{t-1})))_Y - (I_t - W_t(I_{t-1}))_Y||_2 \\
   + ||(O_t - W_t(O_{t-1})))_{X,Z}||_2
   $$

   (5.2)
where \( X, Y, Z \) are the XYZ channels;

3. No luminance constraint:

\[
L_{temp}^o = \frac{1}{D} M_t \| (O_t - W_t(O_{t-1}))_{R,G,B} \|_2
\]

The each symbol bears the same meaning as in Formula 3.3.

As shown in Figure 5.4, all three approaches obtains reasonable style information when considering visual effect of the whole image. However, when focused on the circled region, the first approach has a much similar luminance-wise temporal error map compared with the input video. This shows the first approach can preserve proper luminance changes between consecutive frames as those in the input, and therefore leads to more natural stylizing outputs.

At the mean time, the total temporal error map of the first approach is also closer to zero than the results of other two approaches, implying more stable stylized results. This may because by adding the luminance warping constraint, the training objective function can also be easier than using an pure output-level temporal loss without such constraint. The original loss function forces the network to generate output video close the an abedo pass without any illumination effect. However, since the input video always bear some illumination effects, the training can be very hard. On the contrary, the output-level temporal loss with a luminance warping constrains is then more natural for network training since the network only needs to capture style and temporal information without care much about reducing illumination. This in turn makes the training effect better. From the numerical result of the same figure, it is also validated that the first approach also produces a much lower overall temporal error. This justifies our analysis.

Based on both qualitative and quantitative analysis, the final default luminance warping constraint used in this project is set to the first approach as shown in Equation 3.3. The results show that it can generate smoother color change on areas with illumination effects and achieve better temporal coherence than the other two approaches.
### 3.2 Effect of Using Multi-level Temporal Loss

This section discusses the effects of feature-map-level temporal loss and output-level temporal loss. Both numerical and visual results are analyzed in this section to give a full picture of their contributions to the final model.

#### 3.2.1 Numerical Results

We use Candy in the experiment to study the effect of multi-level temporal loss. Since Candy is a difficult style to learn, the effect of a loss can be more obvious in the setting. The same $e_{stab}$ metric (see Equation 5.1) is used to evaluate the temporal coherence on five scenes in MPI Sintel Dataset. To better compare each loss, there are in total three experiments performed:

- (1) Only using feature-map-level temporal loss;
- (2) Only using output-level temporal loss;
- (3) Using both feature-map-level temporal loss and output-level temporal loss.

As shown in Table ??, compared experiments (1), the $e_{stab}$ error in experiments (3) is significantly reduced with the help of output-level temporal loss, meaning that the stylized video has much better temporal coherence. At the same time, compared with experiments (2), with the help of feature-map-level temporal loss, there is also improvements on the temporal consistency. This shows that both feature-map-level and output-level temporal losses can have a positive effect to the final output quality.

<table>
<thead>
<tr>
<th>Loss Levels</th>
<th>Alley-2</th>
<th>Ambush-5</th>
<th>Bandage-2</th>
<th>Market-6</th>
<th>Temple-2</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature-map only</td>
<td>0.1028</td>
<td>0.1041</td>
<td>0.0752</td>
<td>0.1062</td>
<td>0.0991</td>
<td>0.0975</td>
</tr>
<tr>
<td>Output only</td>
<td>0.0854</td>
<td>0.0840</td>
<td>0.0672</td>
<td>0.0868</td>
<td>0.0820</td>
<td>0.0813</td>
</tr>
<tr>
<td>Both</td>
<td>0.0846</td>
<td>0.0819</td>
<td>0.0662</td>
<td>0.0862</td>
<td>0.0831</td>
<td>0.0804</td>
</tr>
</tbody>
</table>

Table 5.3: Temporal error $e_{stab}$ with style Candy for different temporal loss settings in ReCoNet. Five scenes from MPI Sintel Dataset are selected for validation.
3.2.2 Visual Results

Going deeper into the experiments in the previous section, more specific visual changes can be summarized for the involvement of each new loss. Figure 5.5 demonstrates the effect of feature-map-level temporal loss. When only using output-level temporal loss as shown in sub-figure (a), the same object can alter its color affected by the surrounding environment. However, in sub-figure (b), there is no such inconsistency since the features of the object are preserved in consecutive frames due to feature-map-level temporal loss. This gives a more coherent stylized output.

The result of approach 1 in Figure 5.4 also demonstrates that the new output-level temporal loss can properly handle the illumination effects and reduce the overall temporal error map. This also contribute to a more natural stylized output and more stable stylized video as well.
Chapter 6

Conclusion and Future Works

This report discusses a novel feed-forward neural network architecture ReCoNet for video stylized transfer. The literature review shows the gap existed in previous works between fast processing speed and pleasant visual output. ReCoNet is proposed to fit the gap and achieved fast speed, high temporal consistency and rich artistic styles at the same time. A luminance warping constraint and a feature-map-level temporal loss are involved to ensure the final output quality. Experimental results and ablation studies verify the intuition in model design and show our advantages compared with the existing works.

In the future, we plan to further investigate the possibility of utilizing chromaticity changes in the inter-frame result for better video style transfer model. More luminance warping constraints can also be studied for any potential improvements. The network structure and parameters can be further fine-tuned for better output quality. The desktop application can also be extended to be deployed on mobile platforms.
References


