1. Algorithm Details

1.1. Network architectures

We provide the architectural details of our mask enhancement network, guidance learner, spatiotemporal feature transfer, and grid learner in Table 1. As shown in Figure 1, we extract the VGG-19 features from four scales (CONV1_1, CONV2_1, CONV3_1, and CONV4_1) and match the multi-resolution feature statistics via three splatting blocks [9]. Note we remove the global scene summary path in Xia et al. [9] as we aim to transfer style between local regions, where the global features will not help. In each splatting block, the weights of the first and the last convolutional layers are shared for the content and style feature paths. The output of each ST-AdaIN layer is propagated to the next frame for blending the temporal information. After the feature transferring, we apply two convolutional layers to predict the bilateral grids.

1.2. Soft grid mask

In Algorithm 1, we provide the detailed steps to compute the soft grid mask for blending the foreground and background grids.

1.3. Implementation details

We implement our model in Tensorflow [1]. For training, we use the training set from DAVIS 2017 [8] which contains 60 videos and each video has 70 frames on average. We optimize the model using Adam [3] with hyperparameters $\alpha = 10^{-4}, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$, and a batch size of 2 video clips. At each iteration, we take five consecutive frames as a clip and randomly crop the original frames into a resolution of $256 \times 256$. To train the mask enhancement network, we generate noisy input masks by applying erosion and dilation with random kernel sizes to the ground-truth object masks. Since mask enhancement is a largely orthogonal task, we first pre-train this module for 20000 iterations and freeze the weights. We then train the remainder of the network end-to-end for 90000 iterations. By limiting the “full-res” input to also be $256 \times 256$, we can significantly reduce training time, which takes about two days on a single NVIDIA Tesla V100 GPU with 16 GB RAM. At inference time, the trained model can be applied to arbitrary input resolution, because of the usage of bilateral grid.

2. Grid Sub-sampling

The proposed method can be further sped up by sub-sampling the bilateral grids in the temporal domain. As the nearby frames typically have similar color, the bilateral grids can also be shared to render the stylized result.
Specifically, we can generate the bilateral grid for every \(r\) frames and estimate the intermediate grids by a linear interpolation, as shown in Figure 2(a). In Figure 2(c) and (d), we show that our method can still render high-quality results up to a sub-sampling rate of \(r = 8\). For \(r = 16\), the estimated grid may have a larger spatial mismatch with the content, resulting in undesired visual artifacts. Such a temporal sub-sampling strategy is suitable for the proposed method and Xia et al. [9]. Other approaches, e.g., WCT\(^2\) [10], is not able to generate reasonable results by simply interpolating the output frames, as shown in Figure 2(b).

By utilizing such a grid sub-sampling strategy, we can reduce the computational cost and speed up the processing time during inference. As shown in Table 2, our model with sub-sampling rate \(r = 8\) is about \(8 \times\) faster at four different image resolutions. The video comparisons are provided in the supplementary videos grid_sampling.mp4 in the video_mp4 folder.

### Table 2: Execution time. An asterisk denotes that the technique of Lai et al. [4] was applied to improving temporal stability. OOM indicates out of memory.

<table>
<thead>
<tr>
<th>Image Size</th>
<th>512 \times 512</th>
<th>1024 \times 1024</th>
<th>2000 \times 2000</th>
<th>3000 \times 4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lai et al. [4]</td>
<td>0.0024s</td>
<td>0.0031s</td>
<td>0.0073s</td>
<td>OOM</td>
</tr>
<tr>
<td>LST* [5]</td>
<td>0.2753s</td>
<td>0.8356s</td>
<td>OOM</td>
<td>OOM</td>
</tr>
<tr>
<td>PhotoWCT* [6]</td>
<td>0.6366s</td>
<td>1.5185s</td>
<td>OOM</td>
<td>OOM</td>
</tr>
<tr>
<td>WCT(^2)* [10]</td>
<td>3.8599s</td>
<td>6.1375s</td>
<td>OOM</td>
<td>OOM</td>
</tr>
<tr>
<td>Xia et al. * [9]</td>
<td>0.0058s</td>
<td>0.0068s</td>
<td>0.0117s</td>
<td>OOM</td>
</tr>
<tr>
<td>Ours ((r = 1))</td>
<td>0.0378s</td>
<td>0.0380s</td>
<td>0.0414s</td>
<td>0.0464s</td>
</tr>
<tr>
<td>Ours ((r = 8))</td>
<td>0.0048s</td>
<td>0.0049s</td>
<td>0.0052s</td>
<td>0.0058s</td>
</tr>
</tbody>
</table>

### 3. More Qualitative Comparisons

#### 3.1. Effect of TC-AdaIN

Figure 3 shows a visual comparison, where the results without the TC-AdaIN have a visible color shift. We can see
3.3. Anti-distortion module

Recent photorealistic style transfer methods [6, 5, 10] are based on encoder-decoder architecture, which often cause spatial distortions or unrealistic visual artifacts when reconstructing an image from the low-resolution deep features. Therefore, extra smoothing steps or modules are required to minimize those spatial distortions. PhotoWCT [6] optimizes a matting affinity to ensure spatially consistent stylization. Similarly, LST [5] applies a spatial propagation network (SPN) [7] on the reconstructed images to smooth the results. On the other hand, WCT² [10] replaced max-pooling layers with wavelet pooling where the high-frequency components are skipped to the decoder directly so that all edges and corners are preserved.

Here, we show visual comparisons to existing methods with and without applying their anti-distortion modules. Figure 5 shows the comparison with PhotoWCT. Although the smoothing step helps reduce local artifacts, the smoothed result becomes blurry and hazy. In contrast, our result preserves all the edges and image structures well. In Figure 6, the result of LST without applying SPN has severe spatial distortions. The SPN recovers some image details but cannot suppress all the distortions, resulting in an unrealistic result. Figure 7 shows the results of WCT² with and without the skip connections for high-frequency components. It is clear that the skip connections help bring back the image details and preserve the photorealism, showing comparable results to the proposed method. More video comparisons are provided in the supplementary videos visual_comparison.mp4 in the video.mp4 folder.

3.4. High-resolution style transfer

Although the proposed method is efficient when processing high-resolution (e.g., 2000 × 2000) videos, extracting the object segmentation masks could be computationally expensive. For example, it takes 2.61 seconds for a segmentation model, DeepLab [2], to process a single 1920 × 1080 frame. Therefore, we aim to understand the feasibility of applying low-resolution segmentation masks for high-resolution style transfer. First, we compute the object masks from a low-resolution (256 × 256) input frame, which takes only 0.47 seconds for DeepLab. Then, we resize the masks to 1920 × 1080 through nearest neighbor interpolation for transferring styles. We compare the proposed method with WCT² in Figure 8. The results from WCT² have clear visual artifacts on the object boundaries as the upscaled masks are noisy. On the other hand, our method is able to generate high-quality results, which are comparable to the ones produced by using the ground-truth object masks. We demonstrate that our model performs well on high-resolution videos even if the segmentation masks is extracted from a low-resolution space. Please refer to the supplementary videos HD_video_w_upscaled_mask_*.mp4.
Figure 4: Effect of mask enhancement. When the input object mask is too noisy, the stylized results may still contain visible artifacts on the boundaries. Mask enhancement lets us render high-quality boundaries visually comparable to the result using the ground truth mask.

Figure 5: Visual comparison with PhotoWCT [6].

in the video.mp4 folder for video comparisons.

4. Failure Cases

Our method requires reliable image-space statistics and can cope with modest amounts of segmentation noise. However, it may fail when the selected areas are too small (e.g., a textureless region with a single color). On the other hand, when the foreground objects cannot be detected properly due to occlusion or reflection, our method may not be able to separate the styles of foreground and background very well, as shown in Figure 9.
Figure 6: Visual comparison with LST [5].

Figure 7: Visual comparison with WCT².
Figure 8: High-resolution video stylization using low-resolution masks.

Figure 9: Failure cases.
References


