Style Transfer by Rigid Alignment in Neural Net Feature Space

Suryabhan Singh Hada
Dept. CSE, University of California, Merced
shada@ucmerced.edu

Miguel Á. Carreira-Perpiñán
Dept. CSE, University of California, Merced
mcarreira-perpinan@ucmerced.edu

Abstract

Arbitrary style transfer is an important problem in computer vision that aims to transfer style patterns from an arbitrary style image to a given content image. However, current methods either rely on slow iterative optimization or fast pre-determined feature transformation, but at the cost of compromised visual quality of the styled image; especially, distorted content structure. In this work, we present an effective and efficient approach for arbitrary style transfer that seamlessly transfers style patterns as well as keep content structure intact in the styled image. We achieve this by aligning style features to content features using rigid alignment; thus modifying style features, unlike the existing methods that do the opposite. We demonstrate the effectiveness of the proposed approach by generating high-quality stylized images and compare the results with the current state-of-the-art techniques for arbitrary style transfer.

1. Introduction

Given a pair of style and a target image, style transfer is a process of transferring the texture of the style image to the target image, keeping the structure of the target image unchanged. *Most of the recent work in the neural style transfer is based on the implicit hypothesis is that working in deep neural network feature space can transfer texture and other high-level information from one image to another without altering the image structure much.* Recent work from Gatys *et al*. [10] (Neural style transfer (NST)) shows the power of the Convolution Neural Networks (CNN) in style transfer.

In just a few years, significant effort has been made to improve NST, either by iterative optimization-based approaches [18, 22, 23] or feed-forward network approximation [16, 28, 27, 19, 6, 4, 20, 24, 30, 29]. Optimization-based methods [10, 18, 22, 23], achieve visually great results, but at the cost of efficiency, as every style transfer requires multiple optimization steps. On the other hand, feed-forward network-based style transfer methods [16, 28, 27, 19, 6, 4, 20, 24, 30, 29] provide efficiency and quality, but at the cost of generalization. These networks are limited to a fixed number of styles.

Arbitrary style transfer can achieve generalization, quality, and efficiency at the same time. The goal is to find a transformation that can take style and content features as input, and produce a stylized feature that does not compromise reconstructed stylized image quality.

However, current works in this regard [14, 21, 5, 25] have failed in the quality of the generated results. Among these [14, 5] use external style signals to supervise the content modification on a feed-forward network. The network is trained by using perpetual loss [16], which is known to be unstable and produce unsatisfactory style transfer results [13, 23].

On the contrary, [21, 5, 25] manipulate the content features under the guidance of the style features in a shared high-level feature space. By decoding the manipulated features back into the image space with a style-agnostic image decoder, the reconstructed images will be stylized with seamless integration of the style patterns. However, these techniques over-distort the content or fail to balance the low level and global style patterns.

In this work, we address the aforementioned issues by modifying style features instead of content features during style transfer. *Our hypothesis is if we consider images as a collection of points in feature space, where each point represents some spatial information, and if we align these points clouds using rigid alignment, we can transform these points without introducing any distortion.* By doing so, we solve the problem of content over-distortion since alignment does not manipulate content features. Similar to [21, 25], our method does not require any training and can be applied to any style image in real-time. We also provide comprehensive evaluations to compare with the prior arbitrary style transfer methods [10, 14, 21, 25], to show that our method achieves state-of-the-art performance.

Our contributions in this paper are threefold: 1) We achieve style transfer by using rigid alignment, which is different from traditional style transfer methods that depend on feature statistics matching. Rigid alignment is well studied in computer vision for many years and has been very

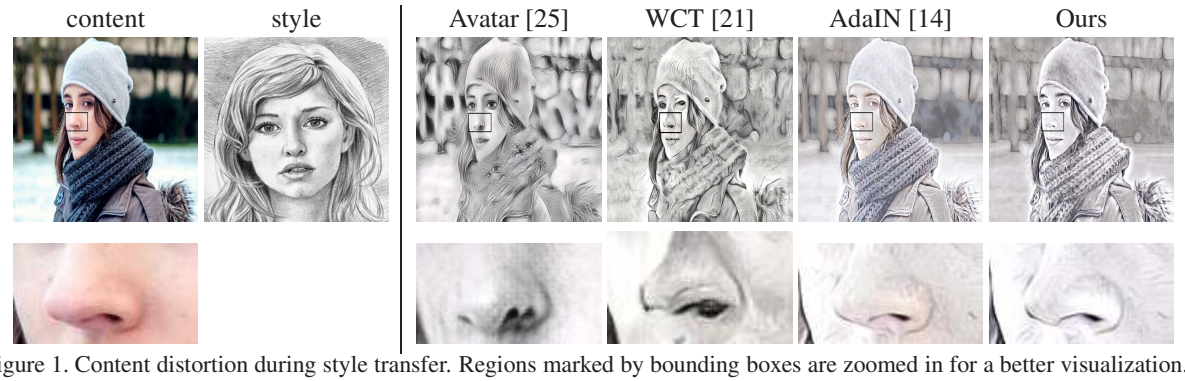

Figure 1. Content distortion during style transfer. Regions marked by bounding boxes are zoomed in for a better visualization.

successful in image registration and many problems of that type. We show that by rearranging the content and style features in a specific manner (each channel (C) as a point in $\mathbb{R}^{HW}$ space, where H is height, and W is the width of the feature), they can be considered as a point cloud of C points. 2) We provide a closed-form solution to the style transfer problem. 3) The proposed approach achieves impressive style transfer results in real-time without introducing content distortion.

2. Related work

Due to the wide variety of applications, the problem of style transfer has been studied for a long time in computer vision. Before seminal work by Gatys *et al*. [10], the problem of style transfer has been focused as *non-photorealistic rendering (NPR)* [17], and closely related to texture synthesis [7, 8]. Early approaches rely on finding low-level image correspondence and do not capture high-level semantic information well. As mentioned above, the use of CNN features in style transfer has improved the results significantly. We can divide the current Neural style transfer literature into four parts.

- **Slow optimization-based methods:** Gatys *et al*. [10] introduced the first NST method for style transfer. The authors created artistic style transfer by matching multi-level feature statistics of content and style images extracted from a pre-trained image classification CNN (VGG [26]) using Gram matrix. Soon after this, other variations were introduced to achieve better style transfer [18, 22, 23], user controls like spatial control and color preserving [11, 23] or include semantic information [9, 3]. However, these methods require an iterative optimization over the image, which makes it impossible to apply in real-time.

- **Single style feed-forward networks:** Recently, [16, 28, 27, 19] address the real-time issue by approximating the iterative back-propagating procedure to feed-forward neural networks, trained either by the perceptual loss [16, 28] or Markovian generative adversarial loss [19]. Although these approaches achieve style transfer in real-time, they require training a new model for every style. This makes them very difficult to use for multiple styles, as every single style requires hours of training.

- **Single network for multiple styles:** Later [6, 4, 20, 24] have tried to tackle the problem of multiple styles by training a small number of parameters for every new style while keeping rest of the network the same. *Conditional instance normalization* [6] achieved it by training channel-wise statistics corresponding to each style. Stylebank [4] learned convolution filters for each style, [20] transferred styles by binary selection units and [24] trained a meta-network that generates a 14 layer network for each content and style image pair. On the other hand, [30] trained a weight matrix to combine style and content features. The major drawback is the model size that grows proportionally to the number of style images. Additionally, there is interference among different styles [15], which affects stylization quality.

- **Single network for arbitrary styles:** Some recent works [14, 21, 5, 25, 12] have been focused on creating a single model for arbitrary style i.e., one model for any style. Gu *et al*. [12] rearrange style features patches with respect to content features patches. However, this requires solving an optimization problem to find the nearest neighbor, which is slow, thus not suitable for real-time use. Chen *et al*. [5] swaps the content feature patches with the closest style feature patch but fails if the domain gap between content and style is large. Sheng *et al*. [25] addresses this problem by first normalizing the features and then apply the patch swapping. Although this improves the stylization quality, it still produces content distortion and misses global style patterns, as shown in fig. 1. WCT [21] transfers multi-level style patterns by recursively applying whitening and coloring transformation (WCT)

to a set of trained auto-encoders with different levels. However, similar to [25], WCT also produces content distortion; moreover, this introduces some unwanted patterns in the styled image [15]. Adaptive Instance normalization (AdaIN) [14] matches the channel-wise statistics (mean and variance) of content features to the style features, but this matching occurs only at one layer, which authors try to compensate by training a network on perpetual loss [16]. Although this does not introduce content distortion, it fails to capture style patterns.

The common part of the existing arbitrary style transfer methods, that they all try to modify the content features during the style transfer process. This eventually creates content distortion. Different from existing methods, our approach manipulates the style features during style transfer. We achieve this in two steps. First, we apply channel-wise moment matching (mean and variance) between content and style features, just as AdaIN [14]. Second, we use rigid alignment (Procrustes analysis [2]) to align style features to content features. This alignment modifies the style features to adapt content structure, thus avoiding any content distortions while keeping its style information intact. In the next sections we describe our complete approach

3. Style transfer in neural net features space

Generally Speaking style transfer as follows Let $\mathbf{z}_c \in \mathbb{R}^{C\times H\times W}$ is a feature extracted from a layer of a pre-trained CNN when the content image passes through the network. Here, H is the height, W is the width, and C is the number of channels of the feature $\mathbf{z}_c$. Similarly, for style image $\mathbf{z}_s \in \mathbb{R}^{C\times H\times W}$ represents the corresponding features.

For any arbitrary style transfer method, we pass $\mathbf{z}_s$ and $\mathbf{z}_c$ to a transformation function $\mathcal{T}$ which outputs styled feature $\mathbf{z}_{cs}$ as described in eq. (1):

$$\mathbf{z}_{cs} = \mathcal{T}(\mathbf{z}_c, \mathbf{z}_s). \tag{1}$$

Reconstruction of $\mathbf{z}_{cs}$ to image space gives the styled image. The difficult part is finding the transformation function $\mathcal{T}$ that is style-agnostic like [25, 5, 21], but unlike these, it captures local and global style information without distorting the content and does not need iterative optimization.

4. Proposed approach

Although AdaIN [14] is not style agnostic, it involves a transformation which is entirely style agnostic: channel-wise moment matching. This involves matching channel-wise mean and variance of content features to those of style features as follows:

$$\mathbf{z}_{c'} = \left(\frac{\mathbf{z}_c - \mathcal{F}_\mu(\mathbf{z}_c)}{\mathcal{F}_\sigma(\mathbf{z}_c)}\right)\mathcal{F}_\sigma(\mathbf{z}_s) + \mathcal{F}_\mu(\mathbf{z}_s). \tag{2}$$

Here, $\mathcal{F}_\mu(.)$ and $\mathcal{F}_\sigma(.)$ is channel-wise mean and variance respectively. Although this channel-wise alignment produces unsatisfactory styled results, it is able to transfer local patterns of style image without distorting content structure as shown in fig. 1. Moment matching does not provide a perfect alignment among channels of style and content features which leads to missing global style patterns and thus unsatisfactory styled results. Other approaches achieve this, either by doing WCT transformation [21] or patch replacement [25, 5], but this requires content features modification that leads to content distortion. We tackle this, by aligning style features to content features instead. In that way, style features get structure of content while maintaining their global patterns.

One simple way of alignment that prevents distortion is rigid alignment [2] and (scaling). This involves shifting, scaling and finally rotation of the points that to be moved (styled features) with respect to the target points (content features after moment matching). For this we consider both features as point clouds of size C with each point is in $\mathbb{R}^{HW}$ space, i.e. $\mathbf{z}_c, \mathbf{z}_s \in \mathbb{R}^{C\times HW}$ (see supplementary section 4 for reasons to choose each point in $\mathbb{R}^{HW}$ space, instead of $\mathbb{R}^{C}$ space). Now, we apply rigid transformation in following steps:

- **Step-I: Shifting.** First, we need to shift both point clouds $\mathbf{z}_c$ and $\mathbf{z}_s$ to a common point in $\mathbb{R}^{HW}$ space. We center these point clouds to the origin as follows:

$$\begin{aligned}\bar{\mathbf{z}}_c &= \mathbf{z}_c - \boldsymbol{\mu}_c \\ \bar{\mathbf{z}}_s &= \mathbf{z}_s - \boldsymbol{\mu}_s.\end{aligned} \tag{3}$$

  Here, $\boldsymbol{\mu}_c$ and $\boldsymbol{\mu}_s \in \mathbb{R}^{HW}$ are the mean of the $\mathbf{z}_c$ and $\mathbf{z}_s$ point clouds respectively.

- **Step-II: Scaling.** Both point clouds need to have the same scale before alignment. For this, we make each point cloud to have unit Frobenius norm:

$$\begin{aligned}\hat{\mathbf{z}}_c &= \frac{\bar{\mathbf{z}}_c}{\|\mathbf{z}_c\|_F} \\ \hat{\mathbf{z}}_s &= \frac{\bar{\mathbf{z}}_s}{\|\mathbf{z}_s\|_F}.\end{aligned} \tag{4}$$

  Here, $\|.\|_F$ represents Frobenius norm.

- **Step-III: Rotation.** Next step involves rotation of $\hat{\mathbf{z}}_s$ so that it can align perfectly with $\hat{\mathbf{z}}_c$. For this, we multiply $\hat{\mathbf{z}}_s$ to a rotation matrix that can be created as follows:

$$\arg\min_{\mathbf{Q}} \|\hat{\mathbf{z}}_s\mathbf{Q} - \hat{\mathbf{z}}_c\|_2^2 \quad \text{s.t.} \quad \mathbf{Q} \text{ is orthogonal.} \tag{5}$$

  Although this is an optimization problem, it can be solved as follows:

$$\|\hat{\mathbf{z}}_s\mathbf{Q} - \hat{\mathbf{z}}_c\|_2^2 = \operatorname{tr}\left(\hat{\mathbf{z}}_s^T\hat{\mathbf{z}}_s + \hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_c\right) - 2\operatorname{tr}\left(\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s\mathbf{Q}\right). \tag{6}$$

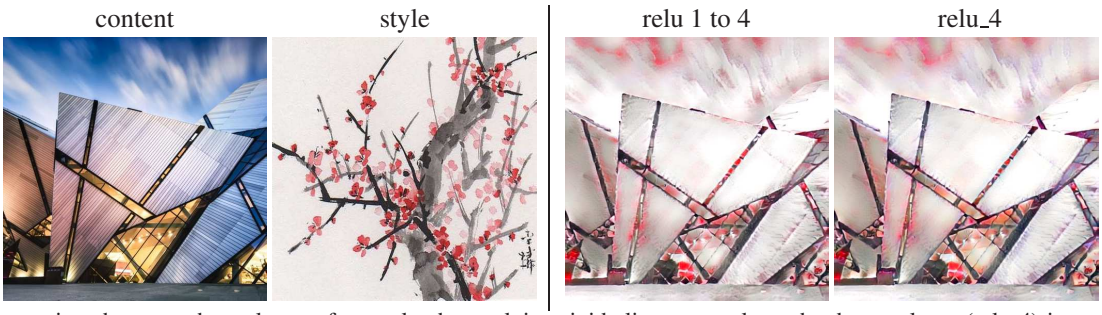

Figure 2. Comparison between the style transfer results, by applying rigid alignment only at the deepest layer (relu_4) instead of every layer. The third image shows the style transfer result by applying alignment at every layer ({relu_1, relu_2, relu_3, relu_4}). On the other hand, the last column shows the style transfer result by applying alignment only at the deepest layer (relu_4). Both produce nearly identical results.

Figure 3. Video stylization using the proposed approach. Similar to WCT [21] and Avatar-Net [25], the proposed method keeps style patterns coherent in each frame. However, unlike the other two, the proposed method does not suffer from content distortion. In the case of WCT [21], the distortion is much worse than Avatar-Net, especially the animal's face. Animations are provided in the supplementary material.

Since, $\mathrm{tr}\left(\hat{\mathbf{z}}_s^T\hat{\mathbf{z}}_s + \hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_c\right)$ term is independent of $\mathbf{Q}$, so eq. (5) becomes:

$$\arg\max_{\mathbf{Q}} \mathrm{tr}\left(\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s\mathbf{Q}\right) \quad \text{s.t.} \quad \mathbf{Q} \text{ is orthogonal.} \tag{7}$$

Using singular value decomposition of $\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s = \mathbf{U}\mathbf{S}\mathbf{V}^T$ and cyclic property of trace we have:

$$\begin{aligned}\mathrm{tr}\left(\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s\mathbf{Q}\right) &= \mathrm{tr}\left(\mathbf{U}\mathbf{S}\mathbf{V}^T\mathbf{Q}\right)\\ &= \mathrm{tr}\left(\mathbf{S}\mathbf{V}^T\mathbf{Q}\mathbf{U}\right)\\ &= \mathrm{tr}\left(\mathbf{S}\mathbf{H}\right).\end{aligned} \tag{8}$$

Here, $\mathbf{H} = \mathbf{V}^T\mathbf{Q}\mathbf{U}$ is an orthogonal matrix, as it is product of orthogonal matrices. Since, $\mathbf{S}$ is a diagonal matrix, so in order to maximize $\mathrm{tr}\left(\mathbf{S}\mathbf{H}\right)$, the diagonal values of $\mathbf{H}$ need to equal to 1. Now, we have:

$$\begin{aligned}\mathbf{H} = \mathbf{V}^T\mathbf{Q}\mathbf{U} &= \mathbf{I}\\ \text{or}, \qquad \mathbf{Q} &= \mathbf{V}\mathbf{U}^T.\end{aligned} \tag{9}$$

- **Step-IV: Alignment.** After obtaining rotation matrix $\mathbf{Q}$, we scale and shift style point cloud with respect to the original content features in the following way:

$$\mathbf{z}_{sc} = \|\mathbf{z}_c\|_F\hat{\mathbf{z}}_s\mathbf{Q} + \boldsymbol{\mu}_c \tag{10}$$

$\mathbf{z}_{sc}$ is the final styled feature.

This alignment makes style features to adapt content structure while keeping its local and global patterns intact.

Note: Above we assume that both $\mathbf{z}_c$ and $\mathbf{z}_s$ are of equal size, so as to make the explanation easy. In case of $\mathbf{z}_c \in \mathbb{R}^{C\times H_cW_c}$ and $\mathbf{z}_s \in \mathbb{R}^{C\times H_sW_s}$, the only change will be in eq. (5) where the orthogonal matrix $\mathbf{Q}$ is rectangular and satisfies $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$ (i.e. $\mathbf{Q} \in \mathbb{R}^{H_sW_s\times H_cW_c}$).

4.1. Multi-level style transfer

As shown in the [10], features from different layers provide different details during style transfer. Lower layer features (*relu_1* and *relu_2*) provide color and texture information, while features from higher layer (*relu_3* and *relu_4*) provide common patterns details (see fig. 1 in supplementary material). Similar to WCT [21], we also do this by cascading the image through different auto-encoders. However, unlike WCT [21] we do not need to do the alignment described in section 4 at every level. We only apply the alignment at the deepest layer (*relu4_1*).

Doing alignment at each layer or only at deepest layer (*relu4_1*) produce identical results as shown in fig. 2. This also shows the rigid alignment of style features to content is perfect.

Once the features are aligned, we only need to take care of local textures at other layers. We do this by applying moment matching (eq. (2)) at lower layers. The complete pipeline is shown in the fig. 1 of the supplementary material.

5. Experiments

5.1. Decoder training

We use a pre-trained auto-encoder network from [21]. This auto-encoder network has been trained for general image reconstruction. The encoder part of the network is the pre-trained VGG-19 [26] that has been fixed, and the decoder network is trained to invert the VGG features to image space. Authors in [21] trained five decoders for reconstructing images from features extracted at different layers of the VGG-19 network. These layers are *relu5_1, relu4_1, relu3_1, relu2_1* and *relu1_1*. However, unlike [21], we use only four decoders in our experiments for multi-level style transfer. These decoders correspond to *relu4_1, relu3_1, relu2_1* and *relu1_1* layers of the VGG-19 network. More details about the decoders are provided in the section 2 of supplementary material.

5.2. Comparison with prior style transfer methods

To show the effectiveness of the proposed method, we compare our results with two types of arbitrary style transfer approaches. The first type is iterative optimization-based [10] and the second type is fast arbitrary style transfer method [21, 24, 14]. We present these stylization results in fig. 4.

Although optimization-based approach [10] performs arbitrary style transfer, it requires slow optimization for this. Moreover, it suffers from getting stuck at a bad local minimum. This results in visually unsatisfied style transfer results, as shown in the third and fourth rows. AdaIN [14] addresses the issue of local minima along with efficiency but fails to capture the style patterns. For instance, in the third row, the styled image contains colors from the content, such as red color on the lips. Contrary to this, WCT [21] and Avatar-Net [24] perform very well in capturing the style patterns by matching second-order statistics and the latter one by normalized patch swapping. However, both methods fail to maintain the content structure in the stylized results. For instance, in the first row, WCT [21] completely destroys the content structure: mountains and clouds are indistinguishable. Similarly, in the second and fifth row, content image details are too distorted. Although Avatar-Net [24] performs better than WCT [21] as in the first and fifth rows, it fails too in maintaining content information, as shown in the second and sixth rows. In the second row, the styled image does not even have any content information.

Method	**Execution time (in sec)** (512×512)
Gatys [10]	58
AdaIN [14]	0.13
WCT [21]	1.12
Avatar-Net [25]	0.34
Ours	0.46

Table 1. Execution time (in seconds) comparison for style transfer among the proposed method and state of the art methods.

On the other hand, the proposed method not only captures style patterns similar to WCT [21] and Avatar-Net [24] but also maintains the content structure perfectly as shown in the first, second, and fifth row where the other two failed.

We also provide a close-up in fig. 1. As shown in the figure, WCT [21] and Avatar-Net [24] distort the content image structure. The nose in the styled image is too much distorted, making these methods difficult to use with human faces. Contrary to this, AdaIN [14] and the proposed method keep content information intact, as shown in the last two columns of the second row. However, AdaIN [14] does not capture style patterns very well. On the other hand, the proposed method captures style patterns very well without any content distortion in the styled image.

In addition to image-based stylization, the proposed method can also do video stylization. We achieve this by just doing per-frame style transfer, as shown in fig. 3. The styled video is coherent over adjacent frames since the style features adjust themselves instead of content, so the style transfer is spatially invariant and robust to small content variations. In contrast, Avatar-Net [25] and WCT [21] contain severe content distortions, with the distortion is much worse in WCT [21].

5.3. Efficiency

We compare the execution time for style transfer of the proposed method with state-of-the-art arbitrary style transfer methods in the table 1. We implement all methods in Tensorflow [1] for a fair comparison. Gatys [10] ap-

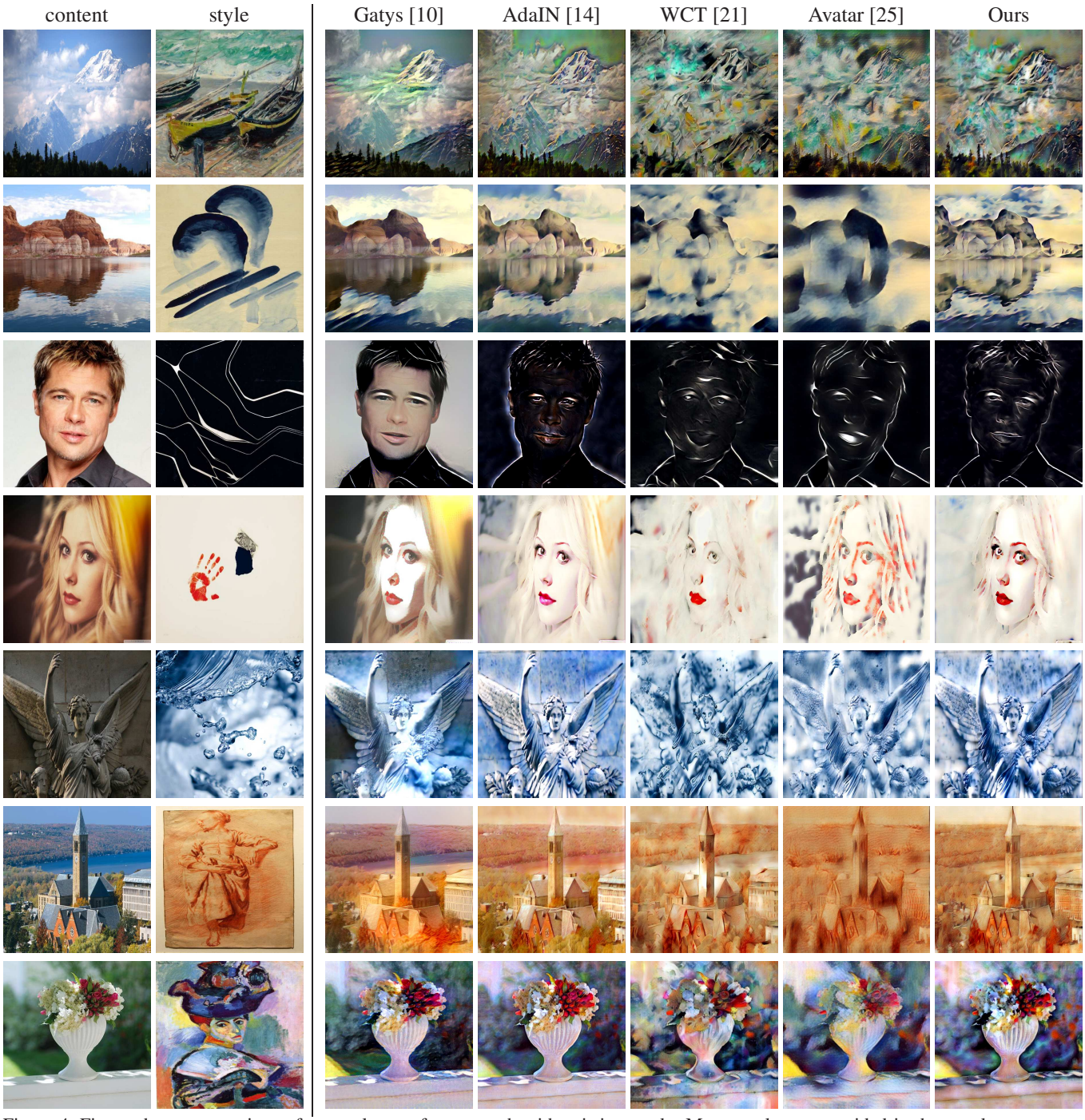

Figure 4. Figure shows comparison of our style transfer approach with existing work. More results are provided in the supplementary material.

proach is very slow due to iterative optimization steps that involve multiple forward and backward pass through a pre-trained network. On the contrary, other methods have very good execution time, as these methods are feed-forward network based. Among all, AdaIN [14] performs best since it requires only moment-matching between content and style features. WCT [21] is relatively slower as it requires SVD operation at each layer during multi-layer style transfer. Avatar-Net [25] has better execution time compared to WCT [21] and ours. This is because of the GPU based style-swap layer and hour-glass multi-layer network.

On the other hand, our method is comparatively slower

Method	$\log(L_c)$	$\log(L_s)$
Gatys [10]	**4.40**	8.28
AdaIN [14]	4.62	8.18
WCT [21]	4.79	7.83
Avatar-Net [25]	4.75	**7.77**
Ours	4.70	7.87

Table 2. Average content and style loss for the styled images in fig. 4. Lower values are better.

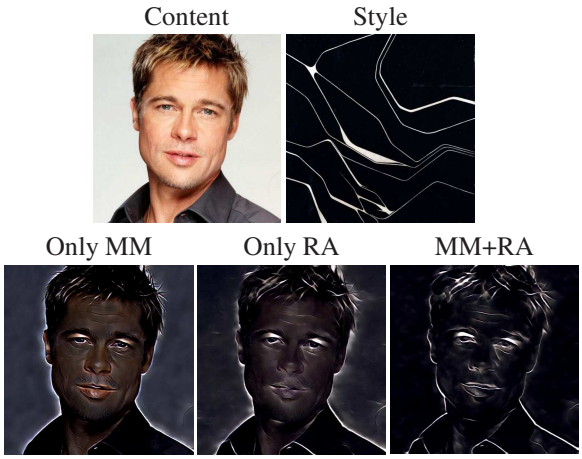

Figure 5. *Row 2:* Style transfer with only moment matching (first column), only rigid alignment (second column), and the proposed method (third column).

than AdaIN [14], and Avatar-Net [25] as our method involves SVD operation at *relu_4*. Additionally, it requires to pass through multiple auto-encoders for multi-level style transfer similar to WCT [21]. However, unlike WCT [21] proposed method needs only one SVD operation as shown in fig. 2 and thus have better execution time compared to WCT [21].

5.4. Numeric comparison

In table 2 we show numerical comparison between different style methods. We provide average content loss (L_c) and style loss (L_s) from [10], for the images in fig. 4:

$$L_c = \frac{1}{2CHW} \sum_{i,j} \left\| \mathbf{z}_{c_{i,j}} - \mathbf{z}_{i,j} \right\|_2^2 \tag{11}$$

$$L_s = \frac{1}{4C^2H^2W^2} \sum_{i,j} \| G_{i,j}(\mathbf{z}_s) - G_{i,j}(\mathbf{z}) \|_2^2. \tag{12}$$

Here, $\mathbf{z}_c$ is the content feature, $\mathbf{z}_s$ is the style feature, $\mathbf{z}$ is the styled feature, and $G(.)$ provides the Gram matrix. As shown in the table 2, WCT [21] and Avatar-Net [25] have smaller style losses because these methods prefer more style patterns in the styled result. However, as shown in fig. 1 and 4 this leads to content distortion. On the other hand,

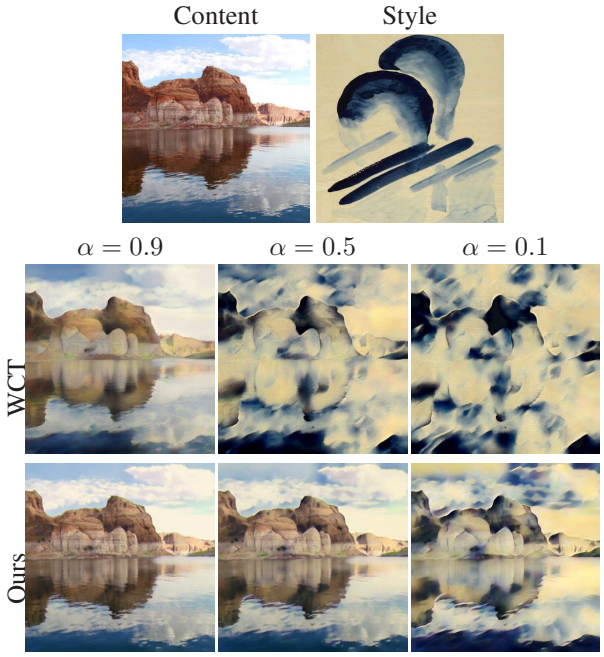

Figure 6. *Row 2:* style transfer using WCT. *Row 3:* style transfer with the proposed approach.

AdaIN [14] performs better in terms of content loss as it maintains more content information, but this produces results with fewer style patterns. So, any method that performs best in either content loss or style loss will produce unsatisfactory styled results. A good style transfer method should perform somewhere in between, which the proposed method achieves. The proposed method not only performs well in terms of content loss but is also on par with WCT [21] and Avatar-Net [25] in terms of style loss. This proves our intuition that by aligning style features to content features, we not only preserve content structure but also effectively transfers style patterns.

Note: Gatys approach [10] should achieve a balanced content and style score similar to ours, but as mentioned in [25] (also shown in third and fourth row in fig. 4) [10] suffers from getting stuck at a bad local minimum. This results in higher style loss as shown in table 2.

6. Ablation study

6.1. Importance of rigid alignment

As described above, our method achieves style transfer by first matching the channel-wise statistics of content features to those of style features and then align style features to content features by rigid alignment. To examine the effect of rigid alignment, we perform the following experiment. We perform style transfer similar to the pipeline described in the section 4.1, but we remove rigid-alignment (RA) in the deepest layer (relu4_1). As shown the fig. 5, moment

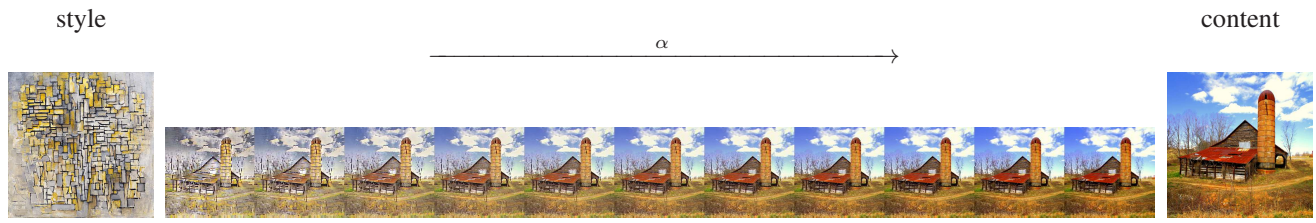

Figure 7. Trade-off between content and style during style transfer. Value of α is increasing from 0 to 1 with an increment of 0.1 from left to right.

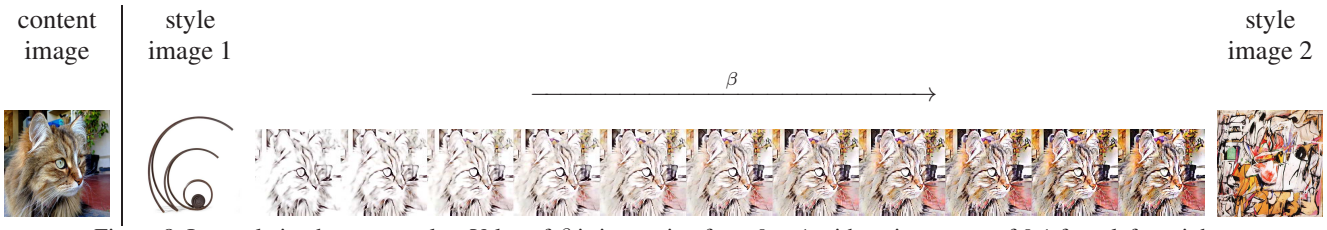

Figure 8. Interpolation between styles. Value of β is increasing from 0 to 1 with an increment of 0.1 from left to right.

matching (MM) only transfers low-level style details (in this case, color) while keeping the content structure intact. On the other hand, if we use only rigid alignment, it mostly transfers global style patterns (white strokes around the hair, second column). Finally, when both are used together (proposed method), the resulting image has both global and local style patterns; and thus achieves better-styled results without introducing content distortion.

6.2. Cost of preserving content with higher content weight

It can be argued that the content structure can be preserved by adjusting the content weight (α). However, having more content weight comes at the cost of ineffective style transfer. In fig. 6, we show one such example, where we compare this trade-off in case of existing work (we use WCT as an example) and the proposed method. In case of previous works, to preserve the content structure, higher content weight is required; but this results in the insufficient transfer of style patterns (first column). On the other hand, for sufficient transfer of style patterns, content weight needs to be reduced; but this creates distorted content in the styled image (third column in the last row). Our method solves this problem effectively; it not only transfers sufficient style patterns, but also preserves the content structure (last column).

7. User control

Like other arbitrary style transfer methods, our approach is also flexible to accommodate different user controls such as the trade-off between style and content, style interpolation, and spatial control during style transfer.

Since our method applies transformation in the feature-space independent of the network, we can achieve trade-off between style and content as follows:

$$\mathbf{z} = \alpha \mathbf{z}_c + (1-\alpha)\mathbf{z}_{sc}. \tag{13}$$

Here, $\mathbf{z}_{sc}$ is the transformed feature from eq. (10), $\mathbf{z}_c$ is content feature and α is the trade off parameter. Fig. 7 shows one such example of content-style trade-off.

Fig. 8 shows an instance of linear interpolation between two styles created by proposed approach. This is done by adjusting the weight parameter (β) between transformation outputs ($\mathcal{T}(\mathbf{z}_c, \mathbf{z}_s)$) as follows:

$$\mathbf{z} = \alpha \mathbf{z}_c + (1-\alpha)(\beta \mathcal{T}(\mathbf{z}_c, \mathbf{z}_{s1}) + (1-\beta)\mathcal{T}(\mathbf{z}_c, \mathbf{z}_{s2})). \tag{14}$$

See section 3 in the supplementary material for details about spatial control during style transfer.

8. Conclusion

In this work, we propose an effective arbitrary style transfer approach that does not require learning for every individual style. By applying rigid alignment to style features with respect to content features, we solve the problem of content distortion without sacrificing style patterns in the styled image. Our method can seamlessly adapt the existing multi-layer stylization pipeline and capture style information from those layers too. Our method can also seamlessly perform video stylization, merely by per-frame style transfer. Experimental results demonstrate that the proposed algorithm achieves favorable performance against the state-of-the-art methods in arbitrary style transfer. As a further direction, one may replace multiple autoencoders for multi-level style transfer by training an hourglass architecture similar to Avatar-Net for better efficiency.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016)*, pages 265–283, Savannah, GA, Oct. 6–8 2016.

[2] Ingwer Borg and Patrick Groenen. *Modern Multidimensional Scaling: Theory and Application*. Springer Series in Statistics. Springer-Verlag, Berlin, 1997.

[3] Alex J. Champandard. Semantic style transfer and turning two-bit doodles into fine artworks. arXiv:1603.01768 [cs.CV], Mar. 16 2016.

[4] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. StyleBank: An explicit representation for neural image style transfer. In *Proc. of the 2017 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'17)*, Honolulu, HI, July 21–26 2017.

[5] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. arXiv:1612.04337, Dec. 16 2016.

[6] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *Proc. of the 5th Int. Conf. Learning Representations (ICLR 2017)*, Toulon, France, Apr. 24–26 2017.

[7] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In Lynn Pocock, editor, *Proc. of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2010)*, pages 341–346, Los Angeles, CA, Aug. 12–17 2010.

[8] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In J. K. Tsotsos, A. Blake, Y. Ohta, and S. W. Zucker, editors, *Proc. 7th Int. Conf. Computer Vision (ICCV'99)*, pages 1033–1038, Kerkyra, Corfu, Greece, Sept. 20–27 1999.

[9] Oriel Frigo, Neus Sabater, Julie Delon, and Pierre Hellier. Split and match: Example-based adaptive patch sampling for unsupervised style transfer. In *Proc. of the 2016 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'16)*, Las Vegas, NV, June 26 – July 1 2016.

[10] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proc. of the 2016 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'16)*, pages 2414–2423, Las Vegas, NV, June 26 – July 1 2016.

[11] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. arXiv:1611.07865, Nov. 16 2016.

[12] Shuyang Gu, Congliang Chen, Jing Liao, and Lu Yuan. Arbitrary style transfer with deep feature reshuffle. In *Proc. of the 2018 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'18)*, Salt Lake City, UT, June 18–22 2018.

[13] Agrim Gupta, Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Characterizing and improving stability in neural style transfer. In *Proc. 17th Int. Conf. Computer Vision (ICCV'17)*, pages 2380–7504, Venice, Italy, Dec. 11–18 2017.

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proc. 17th Int. Conf. Computer Vision (ICCV'17)*, Venice, Italy, Dec. 11–18 2017.

[15] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A Review. arXiv:1705.04058, May 17 2017.

[16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Proc. 14th European Conf. Computer Vision (ECCV'16)*, pages 694–711, Amsterdam, The Netherlands, Oct. 11–14 2016.

[17] Jan Eric Kyprianidis, John Collomosse, Tinghuai Wang, and Tobias Isenberg. State of the "Art": A Taxonomy of Artistic Stylization Techniques for Images and Video. *IEEE transactions on visualization and computer graphics*, 19(5):866–885, July 2012.

[18] Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *Proc. of the 2016 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'16)*, Las Vegas, NV, June 26 – July 1 2016.

[19] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Proc. 14th European Conf. Computer Vision (ECCV'16)*, Amsterdam, The Netherlands, Oct. 11–14 2016.

[20] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Diversified texture synthesis with feed-forward networks. In *Proc. of the 2017 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'17)*, Honolulu, HI, July 21–26 2017.

[21] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 30, pages 386–396. MIT Press, Cambridge, MA, 2017.

[22] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. In *Proc. of the 26th Int. Joint Conf. Artificial Intelligence (IJCAI'15)*, pages 2230–2236, Melbourne, Australia, Aug. 19–25 2017.

[23] Eric Risser, Pierre Wilmot, and Connelly Barnes. Stable and controllable neural texture synthesis and style transfer using histogram losses. arXiv:1701.08893, Jan. 17 2017.

[24] Falong Shen, Shuicheng Yan, and Gang Zeng. Neural style transfer via meta networks. In *Proc. of the 2018 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'18)*, Salt Lake City, UT, June 18–22 2018.

[25] Lu Sheng, Ziyi Lina, Jing Shao, and Xiaogang Wang. Avatar-Net: Multi-scale zero-shot style transfer by feature

decoration. In *Proc. of the 2018 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'18)*, Salt Lake City, UT, June 18–22 2018.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. of the 3rd Int. Conf. Learning Representations (ICLR 2015)*, San Diego, CA, May 7–9 2015.

[27] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proc. of the 33rd Int. Conf. Machine Learning (ICML 2016)*, pages 1349–1357, New York, NY, June 19–24 2016.

[28] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. arXiv:1607.08022, July 16 2016.

[29] Hao Wang, Xiaodan Liang, Hao Zhang, Dit-Yan Yeung, and Eric P. Xing. ZM-net: Real-time zero-shot image manipulation network. arXiv:1703.07255, Mar. 17 2017.

[30] Hang Zhang and Kristin Dana. Multi-style generative network for real-time transfer. arXiv:1703.06953, Mar. 20 2017.