Supplementary Material of "ArtFlow: Unbiased Image Style Transfer via Reversible Neural Flows"

A. Pseudocode of ArtFlow

Algorithm 1 Unbiased style transfer process of ArtFlow

Require: an input content image I_c, an input style image I_s, an unbiased style transfer module T, the proposed new network PFN;
- feed I_c and I_s to PFN;
- perform the forward propagation of PFN and obtain the content feature f_c and the style feature f_s, respectively;
- obtain the style-transferred feature f_{cs} based on T in the manner of $f_{cs} = T(f_c, f_s)$;
obtain the style-transferred image I_{cs} by running the reverse propagation of PFN.
return I_{cs}

B. Proof of the Theorem 1

Theorem 1 *The adaptive instance normalization in AdaIN is an unbiased style transfer module.*

Proof B.1 *Without loss of generality, we assume both f_c and f_s is centered. Therefore, we have,*

$$f_{cs} = \frac{f_c}{\sigma(f_c)}\sigma(f_s), \tag{1}$$

where,

$$C(f) = \frac{f}{\sigma(f)},\ S(f) = \sigma(f). \tag{2}$$

Since,

$$\sigma(f_{cs}) = \sigma(\frac{f_c}{\sigma(f_c)})\sigma(f_s) = 1 \cdot \sigma(f_s) = \sigma(f_s), \tag{3}$$

we have,

$$C(f_{cs}) = \frac{f_{cs}}{\sigma(f_{cs})} = \frac{f_c}{\sigma(f_c)} = C(f_c), \tag{4}$$

$$S(f_{cs}) = \sigma(f_{cs}) = \sigma(f_s). \tag{5}$$

Therefore, the adaptive instance normalization in AdaIN is unbiased. ■

C. Proof of the Theorem 2

Theorem 2 *The whitening and coloring transforms in WCT is an unbiased style transfer module.*

Proof C.1 *Without loss of generality, we assume both f_c and f_s is centered. Therefore, we have,*

$$\text{Whitening}: \hat{f}_c = E_c D_c^{-1/2} E_c^T f_c, \tag{6}$$

$$\text{Coloring}: f_{cs} = E_s D_s^{1/2} E_s^T \hat{f}_c. \tag{7}$$

where f_c, f_{cs} represent the content and style-transferred features, $f_c f_c^T = E_c D_c E_c^T$ and $f_s f_s^T = E_s D_s E_s^T$. Therefore, the style-transferred features f_{cs} can be expressed as,

$$f_{cs} = E_s D_s^{1/2} E_s^T E_c D_c^{-1/2} E_c^T f_c. \tag{8}$$

Here

$$C(f) = E D^{-1/2} E^T f,\ S(f) = E D^{1/2} E^T \tag{9}$$

Since,

$$f_{cs} f_{cs}^T = f_s f_s^T = E_s D_s E_s^T, \tag{10}$$

we have,

$$C(f_{cs}) = E_s D_s^{-1/2} E_s^T f_{cs} \tag{11}$$

$$= E_c D_c^{-1/2} E_c^T f_c \tag{12}$$

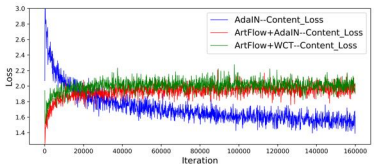
$$= C(f_c), \tag{13}$$

$$S(f_{cs}) = E_s D_s^{1/2} E_s^T = S(f_s). \tag{14}$$

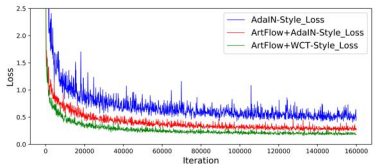
Therefore, the whitening and coloring transforms in WCT is unbiased. ■

D. Training Loss Trends

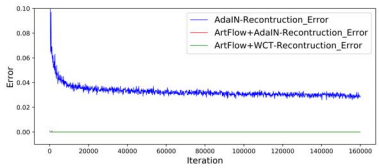
In Fig. 1, we show the training losses of the proposed ArtFlow in comparison with AdaIN. Due to the reversible property of the proposed PFN, our content loss shown in (a) starts from a low value, and the remaining iterations trade-off between the content and style losses. It is worth noting that the converged style loss and the reconstruction error of ArtFlow is significantly lower than AdaIN, which demonstrates that the proposed PFN has a stronger representation



(a) Content Loss

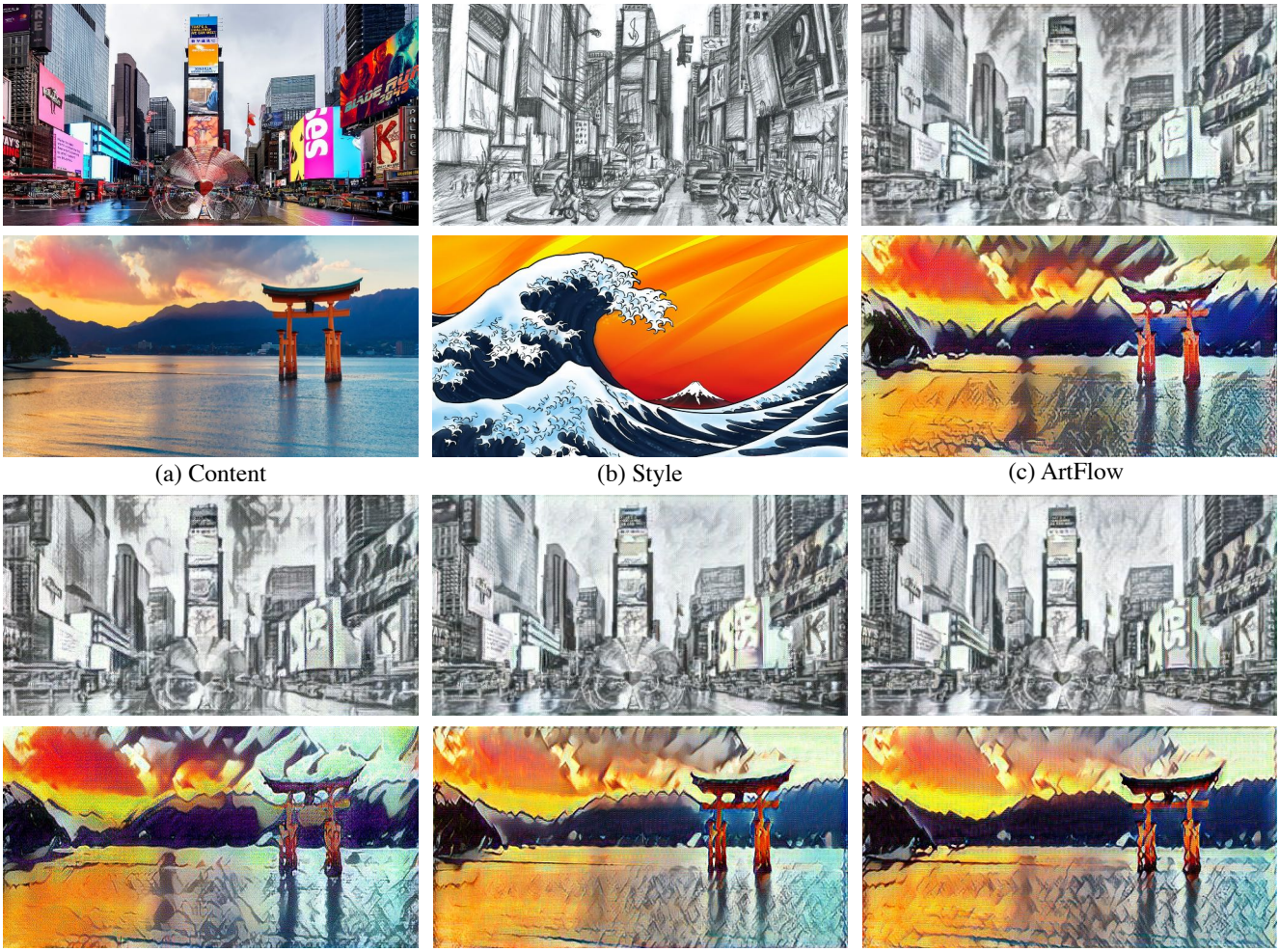


(b) Style Loss



(c) Reconstruction Error

Figure 1. A comparison of the training losses of AdaIN and the proposed ArtFlow.



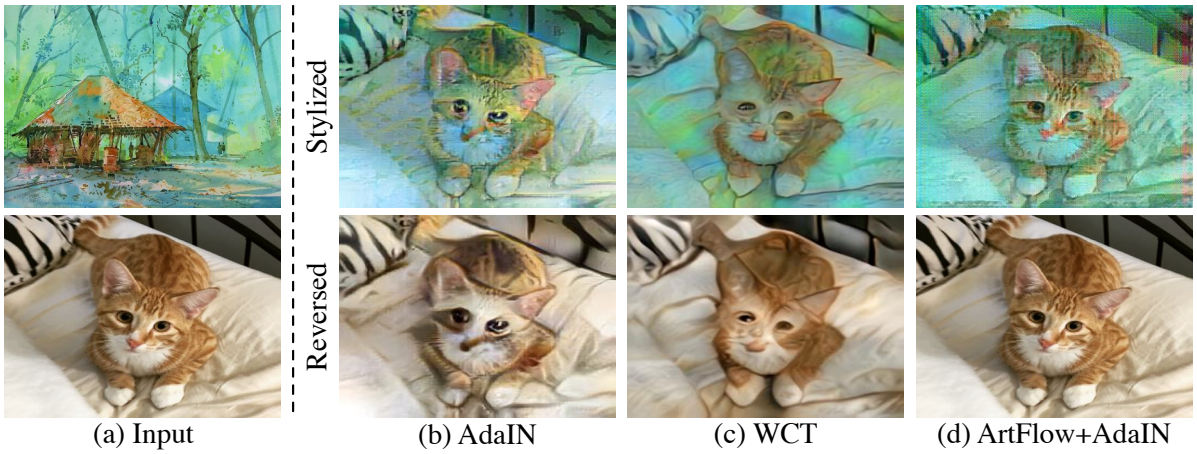
(a) Content (b) Style (c) ArtFlow

(d) ArtFlow-Flow4-Block4 (e) ArtFlow-Flow8-Block1 (f) ArtFlow-Flow16-Block1

Figure 2. A comparison of the style transfer result by different neural flow architectures.

Table 1. Quantitative evaluation results different flow architectures.

Method	ArtFlow (PFN)	ArtFlow-Flow4-Block4	ArtFlow-Flow8-Block1	ArtFlow-Flow16-Block1
SSIM ↑	0.45	0.401	**0.482**	0.480
Gram Loss ↓	**0.00098**	0.00130	0.00139	0.00137



(a) Input (b) AdaIN (c) WCT (d) ArtFlow+AdaIN

Figure 3. A comparison of reverse style transfer results.

ability in making style transfer than existing auto-encoder based framework. Moreover, ArtFlow maintains the image reconstruction error to be zero across the training loop, which reinforces that the proposed PFN can achieve lossless image projection/recovery.

E. Network Structure Analysis of PFN

In addition to the proposed PFN, we also conduct experiments on three alternative network architectures based on neural flows: ArtFlow-Flow8-Block1, ArtFlow-Flow16-Block1, and ArtFlow-Flow4-Block4, where the number after "Flow" represents the number of cascades flow modules within a flow block, while the number after "Block" denotes the number of flow blocks. In the experimented network, each flow is a stack of an Actnorm layer, an invertible 1×1 convolution, and an additive coupling layer. According to this naming scheme, the proposed PFN is ArtFlow-Flow8-Block2. Fig. 2 shows the style transfer results by the proposed PFN and all the compared architectures. As Fig. 2 (e) and (f) show, the style transfer results by ArtFlow-Flow16-Block1 and ArtFlow-Flow16-Block1 are not as colorful as other compared architectures, which indicates that these two architectures have compromised stylization abilities. Regarding ArtFlow-Flow4-Block4, it can produce comparable style transfer results to the proposed PFN. However, in terms of the details, the proposed PFN is better than ArtFlow-Flow4-Block4. Please zoom in to see the details of the Itsukushima Shrine, where the proposed PFN creates more colorful results. We also conduct a quantitative comparison between the proposed ArtFlow and three alternative architectures. As Tab. 1 shows, the proposed PFN achieves a better trade-off between the SSIM index and the Gram loss.

E.1. Reverse Style Transfer

Because the proposed PFN can achieve lossless and unbiased image projection and reversion, ArtFlow enables a very interesting application, where if we perform style transfer on a style-transferred image with the original content image as the new style image, we can recover the original content image in a lossless manner. We name this application the *Reverse Style Transfer*. Fig. 3 shows the reverse style transfer results by AdaIN, WCT, and the proposed ArtFlow. It is remarkable that only the proposed ArtFlow can reverse the style transfer process and recover all the details of the original content image.

F. More Style Transfer Results

We show more comparisons of style transfer results between the proposed ArtFlow and state-of-the-art algorithms in Fig. 4, Fig. 5, Fig. 6, Fig. 7, and Fig. 8.

G. Portrait Style Transfer Results

We use the proposed ArtFlow to make style transfer on portrait images. To train the portrait style transfer model, we use images of the FFHQ [**?**] dataset as the content and images of the Metfaces [**?**] dataset as the style. Fig. 9 and Fig. 10 show the portrait style transfer results by the proposed ArtFlow.

H. More Content Leak Results

We show more comparisons of the *Content Leak* phenomenon in Fig. 11 and Fig. 12.

I. More Content Factor Reconstruction Results

We show more comparisons of the content factor reconstruction results in Fig. 13.

(a) Content (b) Style

(c) StyleSwap (d) AdaIN (e) WCT

(f) LinearWCT (g) OptimalWCT (h) Avatar-Net

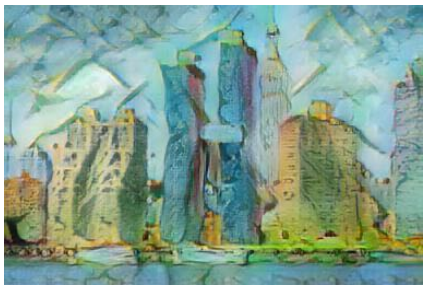
(i) ArtFlow+AdaIN (j) ArtFlow+WCT

Figure 4. A comparison of the style transfer results between the proposed ArtFlow and state-of-the-art algorithms.

(a) Content

(b) Style

(c) StyleSwap



(d) AdaIN

(e) WCT

(f) LinearWCT

(g) OptimalWCT

(h) Avatar-Net

(i) ArtFlow+AdaIN

(j) ArtFlow+WCT

Figure 5. A comparison of the style transfer results between the proposed ArtFlow and state-of-the-art algorithms.

(a) Content

(b) Style

(c) StyleSwap

(d) AdaIN

(e) WCT

(f) LinearWCT

(g) OptimalWCT

(h) Avatar-Net

(i) ArtFlow+AdaIN

(j) ArtFlow+WCT

Figure 6. A comparison of the style transfer results between the proposed ArtFlow and state-of-the-art algorithms.

(a) Content (b) Style

(c) StyleSwap (d) AdaIN (e) WCT

(f) LinearWCT (g) OptimalWCT (h) Avatar-Net

(i) ArtFlow+AdaIN (j) ArtFlow+WCT

Figure 7. A comparison of the style transfer results between the proposed ArtFlow and state-of-the-art algorithms.

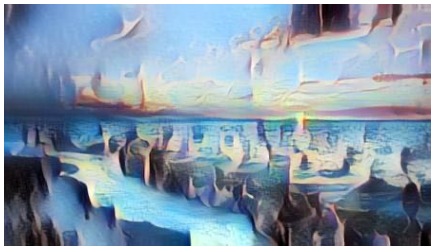
(a) Content

(b) Style

(c) StyleSwap

(d) AdaIN

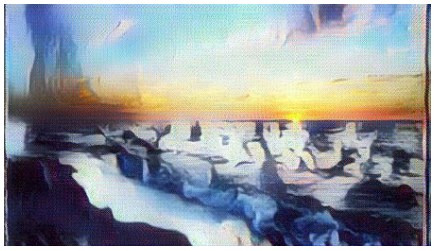
(e) WCT



(f) LinearWCT

(g) OptimalWCT

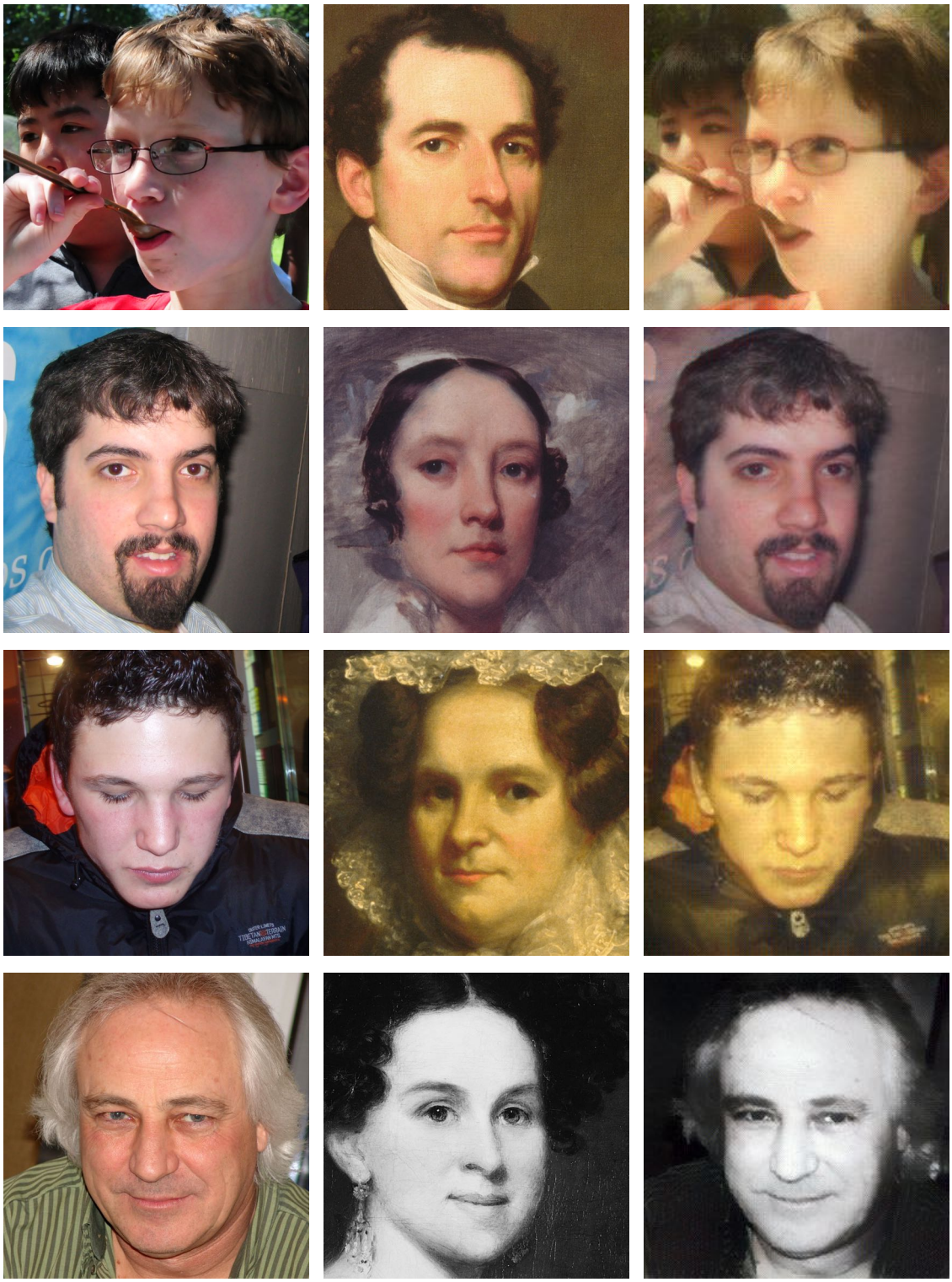
(h) Avatar-Net



(i) ArtFlow+AdaIN

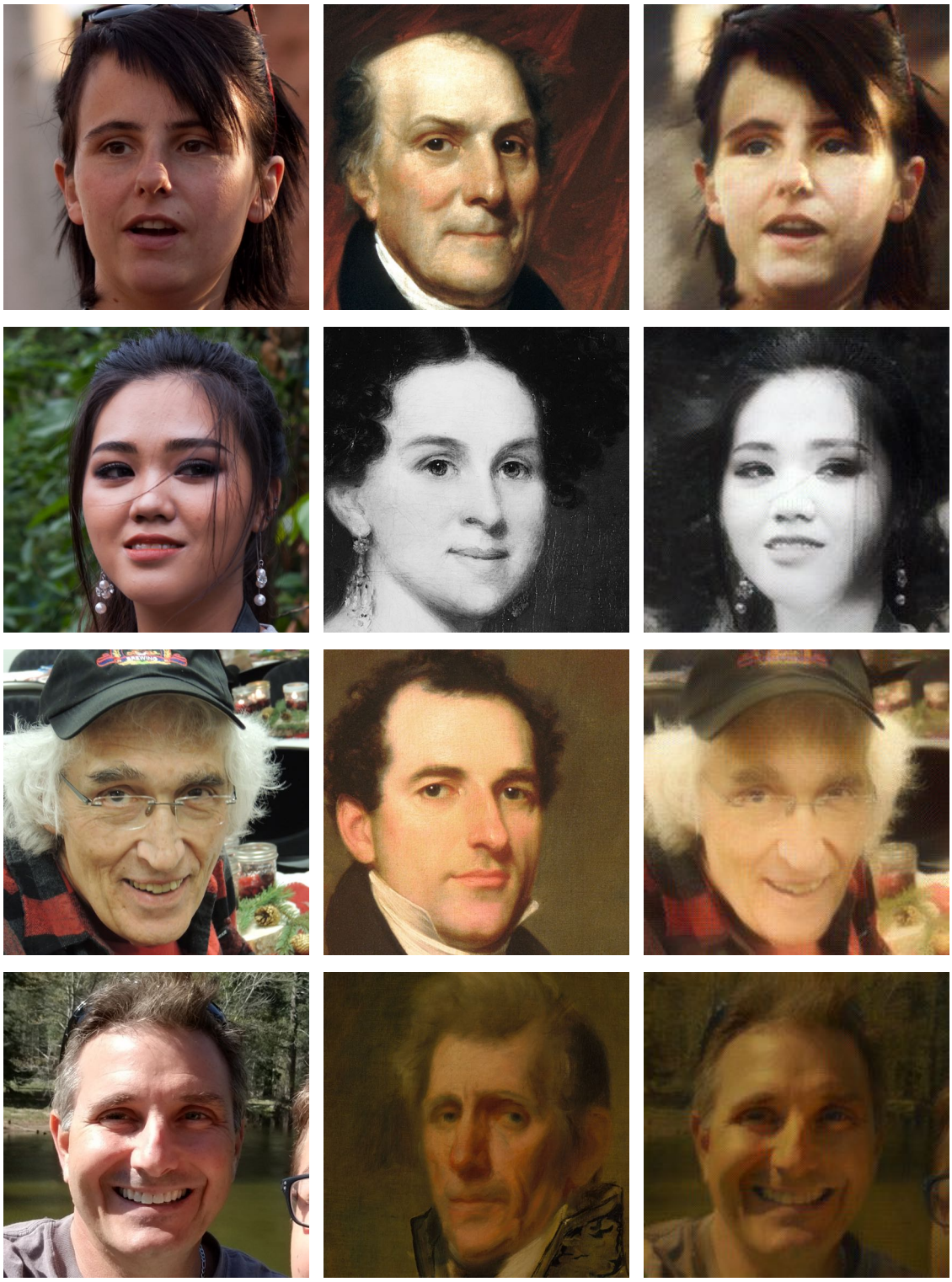
(j) ArtFlow+WCT

Figure 8. A comparison of the style transfer results between the proposed ArtFlow and state-of-the-art algorithms.



(a) Content (b) Style (c) ArtFlow

Figure 9. Portrait style transfer results by the proposed ArtFlow.

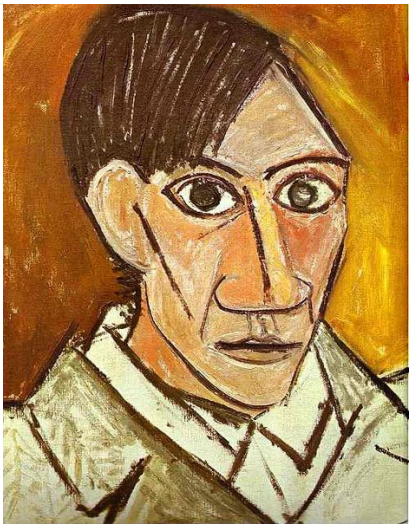


(a) Content (b) Style (c) ArtFlow

Figure 10. Portrait style transfer results by the proposed ArtFlow.

Figure 11. A comparison of the *Content Leak* phenomenon.

Content



Style

Figure 12. A comparison of the *Content Leak* phenomenon.

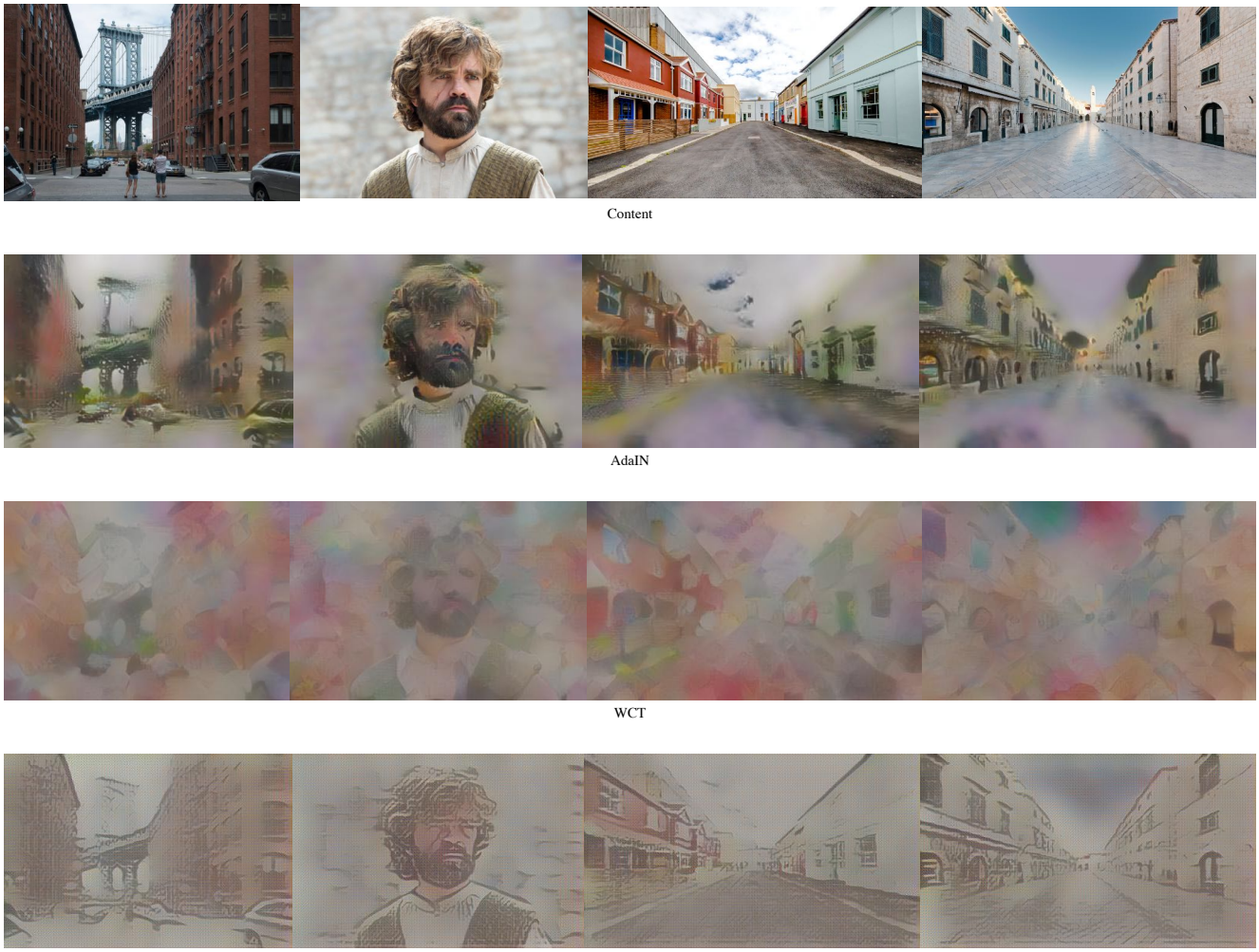


Figure 13. A comparison of the reconstructed content factor.