CLIPstyler: Image Style Transfer with a Single Text Condition

Gihyun Kwon[1] Jong Chul Ye[1,2]
Dept. of Bio and Brain Engineering[1], Kim Jaechul Graduate School of AI[2], KAIST
{cyclomon,jong.ye}@kaist.ac.kr

Figure 1. Our style transfer results on various text conditions. Translated images have spatial structure of the content images with realistic textures corresponding to the text.

Abstract

Existing neural style transfer methods require reference style images to transfer texture information of style images to content images. However, in many practical situations, users may not have reference style images but still be interested in transferring styles by just imagining them. In order to deal with such applications, we propose a new framework that enables a style transfer 'without' a style image, but only with a text description of the desired style. Using the pre-trained text-image embedding model of CLIP, we demonstrate the modulation of the style of content images only with a single text condition. Specifically, we propose a patch-wise text-image matching loss with multiview augmentations for realistic texture transfer. Extensive experimental results confirmed the successful image style transfer with realistic textures that reflect semantic query texts.

1. Introduction

Style transfer aims to transform a content image by transferring the semantic texture of a style image. The seminar work of neural style transfer proposed by Gatys *et al.* [7] uses a pre-trained VGG network to transfer the style texture

by calculating the style loss that matches the Gram matrices of the content and style features. Their style loss has become a standard in later works including stylization through pixel optimization for a single content image [3], arbitrary style transfer which operates in real-time for various style images [9, 16, 18, 27], and optimizing feedforward network for stylizing each image [10, 24].

Although these approaches for style transfer can successfully create visually pleasing new artworks by transferring styles of famous artworks to common images, they require a reference style image to change the texture of the content image. However, in many practical applications, reference style images are not available to users, but the users are still interested in 'imitiating' the texture of the style images. For example, users can imagine being able to convert their own photos into Monet or Van Gogh styles without ever owning paintings by the famous painters. Or you can convert your daylight images into night images by mere imagination. In fact, to overcome this limitation of the existing style transfer and create a truly creative artwork, we should be able to transfer a completely novel style that we imagine.

Toward this goal, several methods have attempted to manipulate images with a text condition which conveys the desired style. Using pre-trained text-image embedding models, these method usually deliver semantic information of text condition to the visual domain. However, these methods often have disadvantages in that semantics are not properly reflected due to the performance limitations of the embedding model [28, 29], and the manipulation is restricted to a specific content domain (such as human face) as the method heavily rely on pre-trained generative models [20].

To address this, here we propose a novel image style transfer method to deliver the semantic textures of text conditions using recently proposed text-image embedding model of CLIP [21]. Specifically, rather than resorting to pixel-optimization or manipulating the instance normalization layer as in AdaIN [9], we propose to train a lightweight CNN network that can express the texture information with respect to text conditions and produce realistic and colorful results. More specifically, the content image is transformed by the lightweight CNN to follow the text condition by matching the similarity between the CLIP model output of transferred image and the text condition. Furthermore, when the network is trained for multiple content images, our method enables text-driven style transfer regardless of content images.

Our method comes from several technical innovations in the implementation. First, instead of optimizing the loss by using the image directly, we propose to use a patch-wise CLIP loss to guide the network to function as a brushstroke. Specifically, to calculate the proposed loss, we first sample patches of the output image and apply augmentation with different perspective views. Afterwards, we obtain the CLIP loss by calculating the similarity between the query text condition and the processed patches. By applying this patch-wise CLIP loss, we found that we can transfer styles to each local area of the content image. Furthermore, the augmentation induces the patch style to be more vivid and diverse. Additionally, to overcome the patch-dependent over-stylization problem, we propose a novel threshold regularization so that the patches with abnormally high scores do not affect the network training.

Extensive experimental results show that our model can transfer a variety of unique styles based on text conditions, which enable a wider range of style transfer than the existing methods using style images.

2. Related works

2.1. Style Transfer

Inspired by Gatys et al. [7] who proposed an iterative pixel-optimization by jointly minimizing content and style losses, Johnson et al. [10] and Ulyanov et al. [24] proposed to train a stylization feed-forward network with using the same loss function by Gatys et al. [7]. By extending the aforementioned single-content style transfer approaches, Li et al. [16, 17] proposed the whitening and coloring transform (WCT) method to transform the content features to follow the statistic of style features. Huang et al. [9] proposed Adaptive Instance Normalization (AdaIN) in which the mean and standard deviation of the style image features are applied to normalized feature statistics of content images. Li et al. [15] proposed a linear transformation between content and style features for fast style transfer on images and videos.

Recently, Yoo et al. [27] proposed a wavelet transform based WCT for better preservation of content information for photo-realistic style transfer. Park et al. [19] proposed a style attentional network (SANet) so that the style can refer the content feature information. Svoboda et al. [23] proposed style transfer with graph convolutional network to combine style and content in latent spaces. Deng et al. [2] suggested multiple adaptation module which use spatial attention as content feature and channel attention as style features.

More recently, Liu et al. [18] proposed adaptive attention normalization as a improved attention-based style transfer of SANet [19]. Xu et al. [26] proposed a new framework of dynamic residual block to integrate style and content features of generative models. Hong et al. [8] introduce a domain-aware style transfer method in which the model transfer the domain-aware information along with style. Kotovenko et al. [13] focused on the brushstroke property of Bezier curves, and proposed to optimize parameters of modelled quadratic Bezier curves instead of pixels.

Although these methods have shown successful results,

the methods require style images in order to make the content image to follow the texture of the target style.

2.2. Text-guided synthesis

In the existing text-guided image synthesis, the encoders for text embedding work as guide conditions for generative models. Zhang et al. [28, 29] integrated text conditions to multi-scale generative model for high-quality image synthesis. AttnGAN [25] further improved the performance with attention mechanism on text and image features. ManiGAN [14] proposed a modules for simultaneously embedding the text and image features.

Recently, OpenAI introduced CLIP [21] which is a high-performance text-image embedding models trained on 400M text-image pairs. CLIP model have shown a state-of-the-art performance on connecting text and image domain. With powerful representation embedding of CLIP, there are several approaches which can manipulate the image with text conditions. StyleCLIP [20] performed attribute manipulation with exploring learned latent space of StyleGAN [12]. They successfully controlled the generation process by finding an appropriate vector direction towards the given text condition. However, StyleCLIP has limitation as the latent exploration can manipulate images within the trained domain. Therefore, StyleGAN-NADA [6] proposed a model modification method with using text condition only, and modulates the trained model into a novel domain without additional training images.

Although these models could manipulate the images with text conditions, they are heavily dependent on pre-trained generative models. Therefore, the generated images are confined to trained image domains. In our model, we can transfer the texture of text conditions to the source image regardless of the domain of images, which was not considered in the aforementioned generative model based manipulations.

3. Method

3.1. Basic framework of CLIPstyler

As emphasized before, the purpose of our framework is to transfer the semantic style of target text t_{sty} to the content image I_c through the pre-trained text-image embedding model CLIP [21]. The major difference from the existing methods is that in our model, there is no style image I_s to use as a reference.

Since our model aims to obtain semantically transformed images with a sole supervision from CLIP, we have several technical issues to solve: 1) how to extract the semantic 'texture' information from CLIP model and apply the texture to the content image, and 2) how to regularize the training so that the output image is not qualitatively impaired.

The specified architecture of our method is shown in Fig. 2. When a content image I_c is given, we aim to obtain the style transfer output I_{cs}. Unfortunately, we experimentally found that the desired texture is not reflected when using traditional pixel optimization method. To solve the problem, we introduced a CNN encoder-decoder model f that can capture the hierarchical visual features of the content image and simultaneously stylize the image in deep feature space to obtain a realistic texture representation. Therefore, our stylized image I_{cs} is $f(I_c)$, and our final goal is to optimize the parameter of f so that it can make the output to have target texture.

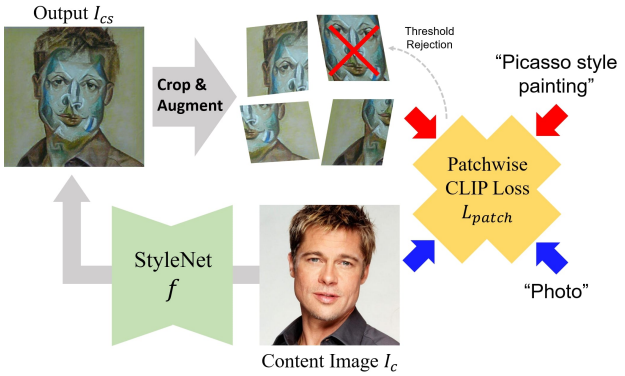

Figure 2. Overall schematics of our proposed patch-wise CLIP loss. We optimize the neural network f using the loss functions.

3.2. Loss function

CLIP loss: To guide the content image to follow the semantic of target text, the simplest CLIP-based image manipulation approach [20] is to minimize the global clip loss function which is formulated as:

$$L_{global} = D_{CLIP}(f(I_c), t_{sty}), \tag{1}$$

where D_{CLIP} is the CLIP-space cosine distance. This loss function transforms the output image of the whole frame to follow the semantic of the textual condition. However, when such global CLIP loss is used, often the output quality is corrupted, and the stability is low in the optimization process.

To solve the problem, StyleGAN-NADA [6] proposed a directional CLIP loss that aligns the CLIP-space direction between the text-image pairs of source and output. So, we also employ the directional CLIP loss, which can be defined in our case as:

$$\begin{aligned} \Delta T &= E_T(t_{sty}) - E_T(t_{src}), \\ \Delta I &= E_I(f(I_c)) - E_I(I_c), \\ L_{dir} &= 1 - \frac{\Delta I \cdot \Delta T}{|\Delta I||\Delta T|}, \end{aligned} \tag{2}$$

where E_I and E_T are the image and text encoders of CLIP, respectively; and t_{sty}, t_{src} are the semantic text of the style

target and the input content, respectively. When we use natural images for content, t_{src} is simply set as "Photo".

PatchCLIP loss: Although the proposed L_{dir} shows good performance in modifying the pre-trained generative model, it does not perfectly match to ours because our goal is to apply the semantic texture of t_{sty} to a given content image. It is also shown in our result that solely using L_{dir} in our framework decreased the quality of the output.

To overcome the shortcomings of existing CLIP losses, we propose a novel PatchCLIP loss for texture transfer. Inspired by the idea of the original style loss in Gatys et al. [7], which is to deliver the spatially invariant information, we found that a similar effect can be obtained by minimizing CLIP loss functions with respect to group of patches that are extracted from I_{cs} in arbitrary locations.

Specifically, we randomly crop sufficient number of patches from I_{cs}. At this stage, the size of the cropped image is fixed. For all of N cropped patches $\hat{I}_{cs}^{N}$, we apply random geometrical augmentation to the cropped patches before calculating the CLIP directional loss. Inspired by Frans et al. [5], we found that using augmentations on each patch assist the network to represent more vivid and diverse textures.

Although there are many possible types of augmentations, we propose to use perspective augmentation. With using perspective augmentation, all patches are guided to have the same semantic when viewed in multiple points so that the semantic information of the CLIP model can be reconstructed as more 3D-like structures.

Threshold rejection: Due to the stochastic randomness of patch sampling and augmentations, our method often suffer from over-stylization in which the style network f is optimized on specific patches that are easy to minimize the loss scores. To alleviate the problem, we include regularization to reject the gradient optimization process for high-scored patches. With a given threshold value τ, we simply nullify the calculated loss of corresponding patches. Therefore, our proposed patch-wise CLIP loss is defined as:

$$\begin{aligned} \Delta T &= E_T(t_{sty}) - E_T(t_{src}), \\ \Delta I &= E_I(aug(\hat{I}_{cs}^{i})) - E_I(I_c), \\ l_{patch}^{i} &= 1 - \frac{\Delta I \cdot \Delta T}{|\Delta I||\Delta T|}, \\ L_{patch} &= \frac{1}{N}\sum_{i}^{N} R(l_{patch}^{i}, \tau) \qquad (3) \\ \text{where} \quad R(s,\tau) &= \begin{cases} 0, & \text{if} \quad s \le \tau \\ s, & \text{otherwise} \end{cases} \end{aligned}$$

where $\hat{I}_{cs}^{i}$ is the i-th cropped patch from the output image, aug is random perspective augmentation, and $R(\cdot,\cdot)$ represents the threshold function.

Total loss: For overall loss function, we use four different losses. First, we use standard directional CLIP loss L_{dir} to modulate the whole part of the content image (e.g. color tone, global semantics). Second, we add our proposed PatchCLIP loss L_{patch} for local texture stylization. On top of CLIP loss functions, to maintain the content information of input image, we include the content loss L_c with calculating the mean-square error between the features of content and output images extracted from the pre-trained VGG-19 networks similar to the existing work by Gatys et al. [7]. Finally, to alleviate the side artifacts from irregular pixels, we include the total variation regularization loss L_{tv}. Therefore, our total loss function is formulated as:

$$L_{total} = \lambda_d L_{dir} + \lambda_p L_{patch} + \lambda_c L_c + \lambda_{tv} L_{tv} \qquad (4)$$

4. Results

4.1. Experiment settings

Our content image can have any resolution sizes, but considering the resource capacity, we use 512×512 resolution for all content images. For training, we set λ_d, λ_p, λ_c, and λ_{tv} as 5×10^2, 9×10^3, 150, and 2×10^{-3}, respectively. For the content loss, similar to Gatys et al. [7], we use the features of layers "conv4_2" and "conv5_2" for the content loss.

For the neural network f, we use lightweight U-net [22] architecture which has three downsample and three upsample layers, in which the channel sizes are 16,32 and 64 for each downsample layers. For stable training, we include sigmoid function at the last layer of f, so that the pixel value range is within the range of $[0, 1]$. For training the network, we use Adam optimizer with learning rate of 5×10^{-4}. The total training iteration is set as 200, and we decreased the learning rate to half at the iteration of 100. The training time is about 40 seconds per text on a single RTX2080Ti GPU.

For patch cropping, we use the patch size of 128 as our default setting since it shows best perceptual quality. We can obtain various effects with varying the crop size. The total number of cropped patches is set as $n = 64$. For perspective augmentation, we use the function provided by Pytorch library. Detailed implementation is in Supplementary Materials. For threshold rejection, we set τ as 0.7 in which the result has the best visual quality.

In order to reduce the noise of text embedding, we use a prompt engineering technique proposed by Radford et al. [21]. Specifically, we make several texts with same meaning, and feed them to the text encoder. Then we use the averaged embedding instead of original single text conditions. Finally, for better visualization for readers, we apply same contrast enhancement techniques for all outputs including baselines. Please refer to our Github repository: https://github.com/cyclomon/CLIPstyler.

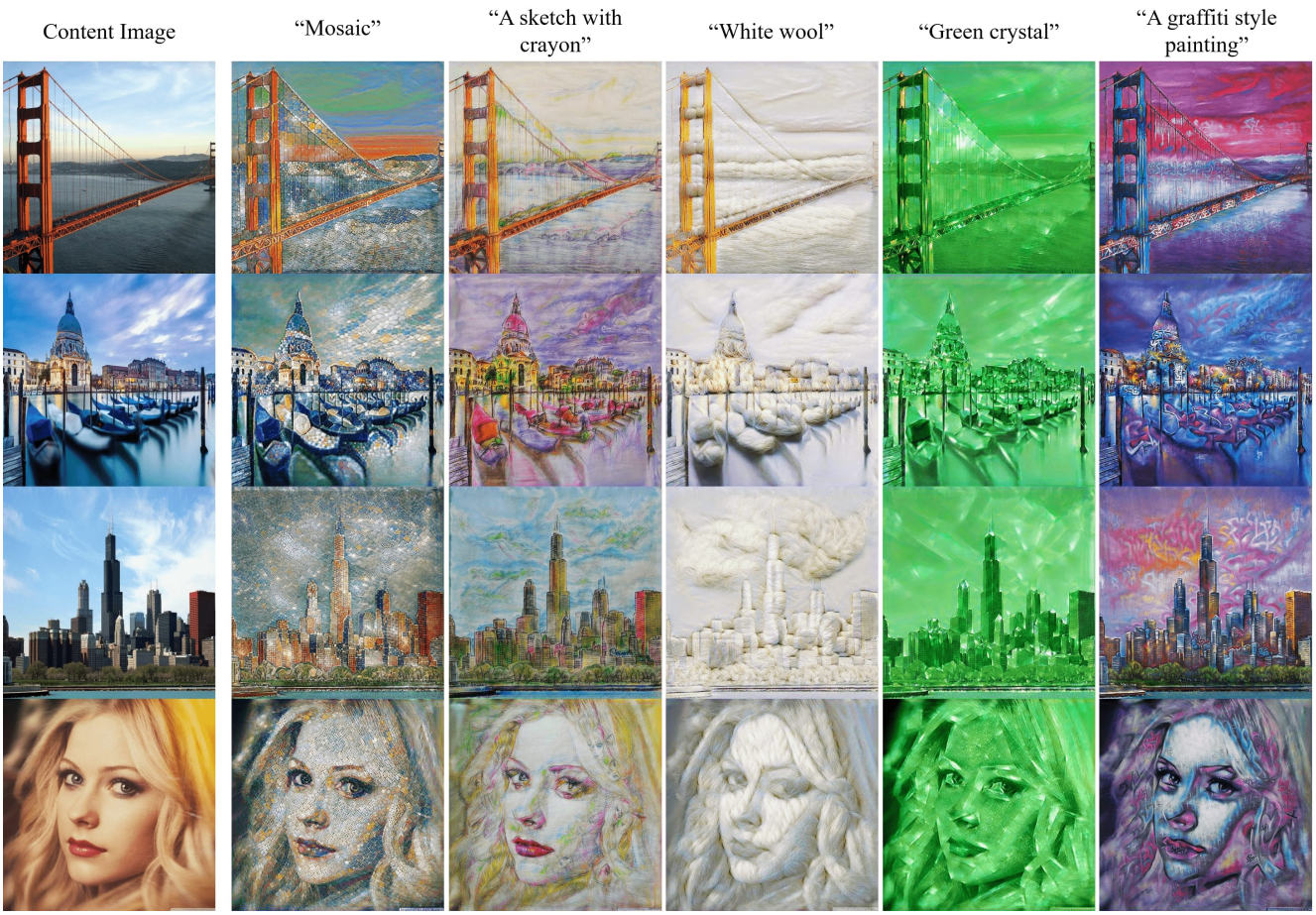

Figure 3. Style transfer results on various query text conditions. Our method can synthesize realistic textures which reflect the text conditions. Additional results are in our Supplementary Materials.

4.2. Qualitative evaluations

Figure 1 and 3 show representative style transfer results from our methods. With corresponding text conditions, we can successfully convert image style that matches the text condition without changing the content of the image. In the result images, we can do style transfer not only for artistic styles, but also for a wide range of general texture styles. In particular, since we give conditions through text conditions, it is possible to control not only the texture type, but also the detailed conditions of each style. For example, we can give additional information of color with textures in using text conditions such as "white wool" and "green crystal". We can also choose which kind of 'object' to use as target texture. In the results of Figure 1, we can choose the texture type ("oil painting") and pattern object ("flowers") simultaneously, so that more abundant effects can be obtained by our method.

4.3. Comparison with baselines

Comparison with existing style transfer: Although our method does not follow the conventional framework of style transfer which require style images, we can indirectly compare our results with existing style transfer methods. Since the CLIP model is trained on a wide range of natural images as well as artistic images, we can transfer the styles of famous artworks with corresponding text conditions and compare them with the existing methods. For baselines, we choose various state-of-the-art artistic style transfer methods which includes arbitrary style transfer (AdaAttn [18], SANet [19], CST [23], and AdaIN [9]), and pixel optimization [7].

In Fig. 4, the results from our method show similar style transfer results to the baseline method in spite of using only text conditions. In the first to third rows, we compare the artistic style transfer performance of our text-guided style transfer with those of the baselines. For the case of early baseline models of AdaIN [9] and Gatys et. al. [7], the outputs show limitation in expressing the target texture style with only concentrating on major color changes. The results by the recent style transfer methods of AdaAttn [18], SANet [19], and CST [23] have complex texture information with preserving the structure of content images. Our

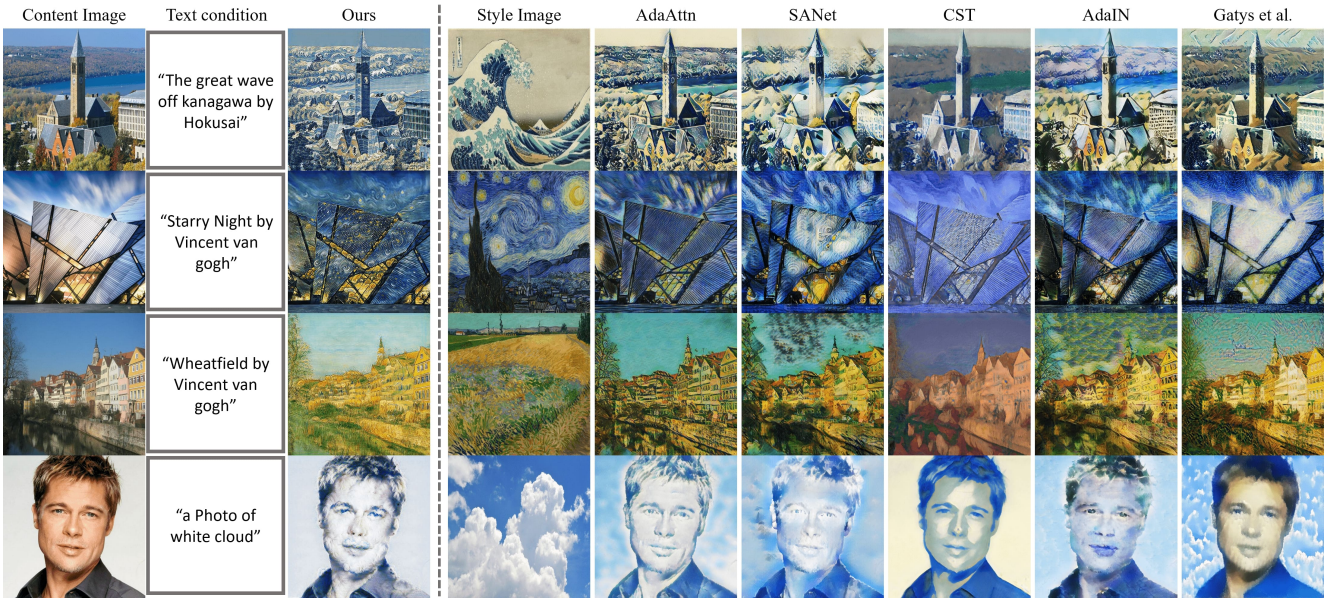

Figure 4. Comparison results with baseline style transfer methods. In contrast to the baselines, the results from our method is stylized only using a text condition without any style image. Although our method does not have style reference, the output image has complex texture which follows the semantics of the text conditions.

result has also vivid texture patterns of artworks in various locations, without damaging the structure of original content shape.

For further comparison with non-artistic styles, we include style transfer result in the last row of Fig. 4. Since the baseline style transfer methods are mostly trained on artworks, we can see that the baseline style transfer methods have difficulty in transferring the texture of the non-artistic styles. However, our method can successfully extract the semantic textures from the query text and apply them to the content image.

Comparison with text-guided manipulation models: Figure 5 shows the result of comparison between our model and the baseline text-guided manipulation methods. As for baselines, we choose state-of-the-art methods which use CLIP model and a pre-trained StyleGAN. Since the baseline models are trained on human face, we experimented with images of CelebA-HQ [11]. In the results, our method can express realistic textures on the content images, which are matched to the query text conditions.

Specifically, in the result of StyleGAN-NADA [6], the content images are modified to follow the text condition, but only a part of the content image can be changed, or the changes do not sufficiently reflect the semantics of the text condition. The result in the last row of Figure 5 shows how our approach and StyleGAN-NADA differ in the way they accept text conditions. Since our model focus on textures, the model apply artistic style (e.g. Van Gogh style) to the entire image, but Style-NADA changed the face identity as it focused on the text condition itself. In the results of Style-

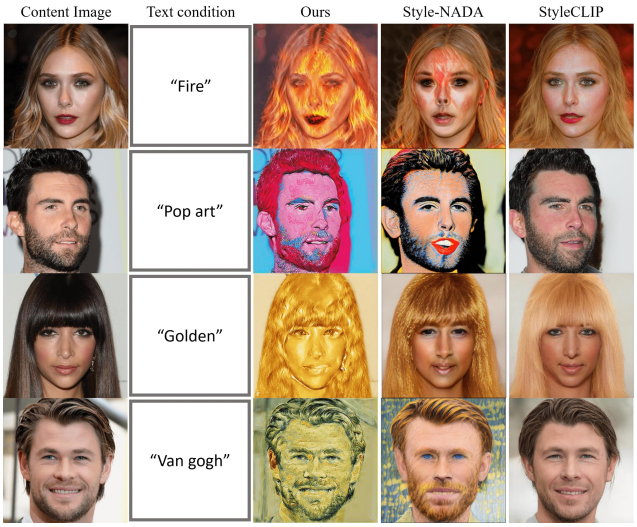

Figure 5. Comparison with other text-guided manipulation models. Our results have more realistic textures on the entire location. Baseline models show limited change, and in some cases the contents of the image are modified.

CLIP [20], since the method can manipulate images within the learned latent domain, all of the results except for the third row failed to transform the images.

For further comparison, as our method is based on the network weight optimization (or fine-tuning) using CLIP loss, we also investigate whether other manipulation approaches can produce better style transfer by combining with CLIP loss. First, we combine CLIP-loss with the pixel optimization similar to Gatys et al. [7]. Second, we only

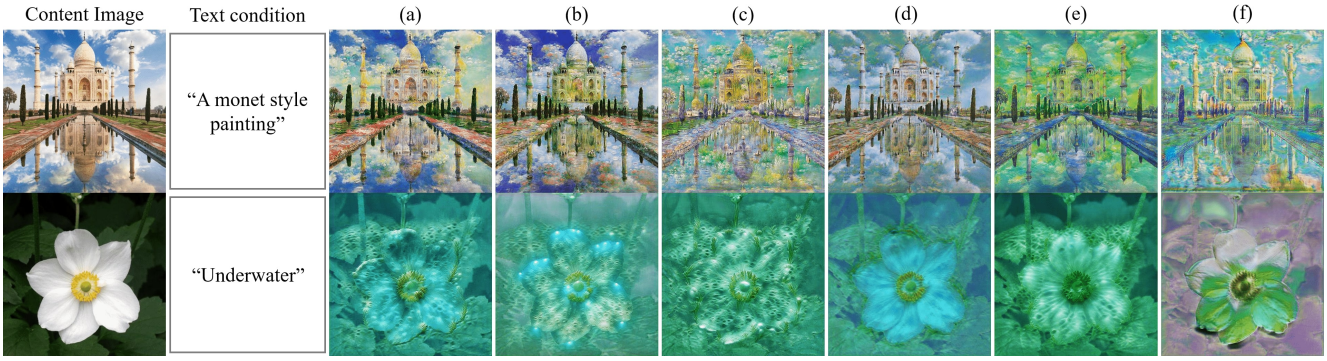

Figure 6. Ablation study results. Each columns annotated with alphabets are the style transfer outputs (a) with using entire losses, (b) with removing L_{dir}, (c) with removing threshold rejection, (d) with removing augmentation, (e) with replacing perspective augmentation to random affine transformation, (f) with removing our proposed L_{patch}.

Figure 7. Comparison study on various manipulation approaches. Top: outputs with “A fauvism style painting”. Bottom: outputs with “Leather”.

apply AdaIN code optimization using CLIP-loss on a pre-trained style transfer network by Huang et al. [9]. Finally, we also compare with the latent code optimization on a pre-trained VQGAN [4] using CLIP-loss by using the source code of VQGAN-CLIP[1]. The results in Fig. 7 show that our method of optimizing network f shows superior quality compared to baselines. More specifically, with pixel optimization, the image could not reflect the semantic of text conditions. In case of VQGAN+CLIP and AdaIN+CLIP, the textures are applied to the contents, but the content structures are severely deteriorated. In Supplementary Materials, we additionally show comparison results with two different baselines of 1) applying the existing style transfer method by using the text- retrieved image as a style image, and 2) using the image generated by the text-to-image generation model as a style image. More comparisons with quantitative user study are also included in our Supplementary Materials.

4.4. Ablation studies

In order to verify the necessity of each component in our method, we conducted ablation studies. Figure 6 shows our ablation study results. When we use all the proposed loss functions (Fig. 6(a)), we can obtain the best results in perceptual domain. In particular, as well as three-dimensional texture, we could obtain a clean image without artifacts in terms of color. If the whole image CLIP loss L_{dir} is not used (Fig. 6(b)), the global semantic cannot be captured, so we can see the color mapped in irregular patterns. When the threshold rejection is removed (Fig. 6(c)), the image is over-focused on a specific patch and an over-stylized image is derived. When augmentation was not used (Fig. 6(d)), the three-dimensional realistic texture was not reflected. When using the commonly used affine transform instead of our proposed perspective augmentation (Fig. 6(e)), there are unwanted artifacts. Finally, when only directional CLIP Loss which is a loss function for the whole image was used, there is only little change in texture except for color (Fig. 6(f)). For further evaluation, we show user study results on various ablation experiments in Supplementary Materials.

[1]https://github.com/nerdyrodent/VQGAN-CLIP

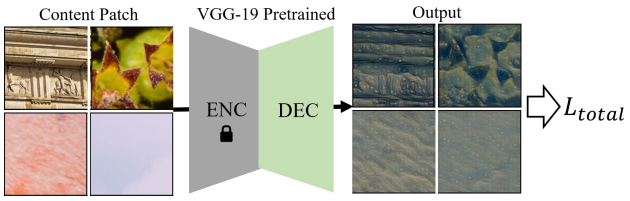

Figure 8. Training framework of our fast style transfer methods. We use cropped patches as content images, and used the same proposed loss functions to train the decoder of VGG19 networks.

5. Further Extensions

5.1. Fast style transfer

In our default framework, we should train the style network f for single content image to apply a given style. To overcome this, we are interested in a method that trains f using various texture patches instead of single content image. Once the network is trained in such a way, the trained network can be used in various content images.

Our proposed fast training scheme is illustrated in Fig. 8. As a training set, we randomly cropped the patches from

Figure 9. Results from our fast style transfer methods.

high-resolution texture images of DIV2k [1]. For faster training, we employ a pre-trained VGG encoder-decoder network rather than the U-Net as the style network f, and only the decoder network is fine-tuned. In training step, we used the same loss functions, but we did not crop the sub-patches because the input patches are already cropped from large images. We update the model for 200 steps with Adam optimizer using learning rate of 1×10^{-4}. The overall training time is $\sim$40 seconds. For inference, it take less than 0.5sec per content image. Further details are in Supplementary Material.

Fig. 9 is the results from our fast style transfer method. Since we trained the model with diverse texture inputs, we can conduct style transfer on arbitrary content images in real-time. We can see that the result images reflected the semantic texture of the text condition. Comparing with our results trained with single content image in Figs. 1 and 3, we can obtain similar texture transfer quality in our faster style transfer settings. Also, the results can adapt to any kind of content inputs with different structural diversities. However, in some cases (e.g. "Neon Light","Fire"), the images have textures on unnecessary regions such as background. Still, the results show that our fast transfer enables high-quality style transfer for arbitrary content images.

Figure 10. High-resolution style transfer results from our fast style transfer methods. Left: content image. Right: style transfer output with text condition of "A sketch with black pencil". Image resolution is 3000×2000. It takes about 4 seconds to process the image.

5.2. High-resolution style transfer

Since our fast style transfer can be adapted to any kinds of content images with patch-based training, we can transfer the style with higher resolution content images. The training scheme is same as our fast style transfer method. After training the VGG model, we can feed the high-resolution image to trained network for style transfer. In Figure 10, we can change the style of input to given text conditions while maintaining the details of the content. More results are in Supplementary Materials.

6. Conclusion

In this paper, we proposed a novel image style transfer framework to transfer the semantic texture information only using text condition. Using novel patchCLIP loss and augmentation scheme, we obtained realistic style transfer results by simply changing the text conditions without ever requiring any style images. Experimental results demonstrated that our framework produced the state-of-the-art image style transfer. For the discussions on limitations and social impacts, please refer to our Supplementary Materials.

Acknowledgement: This research was supported by Field-oriented Technology Development Project for Customs Administration through National Research Foundation of Korea(NRF) funded by the Ministry of Science & ICT and Korea Customs Service(NRF-2021M3I1A1097938), and supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program(KAIST))

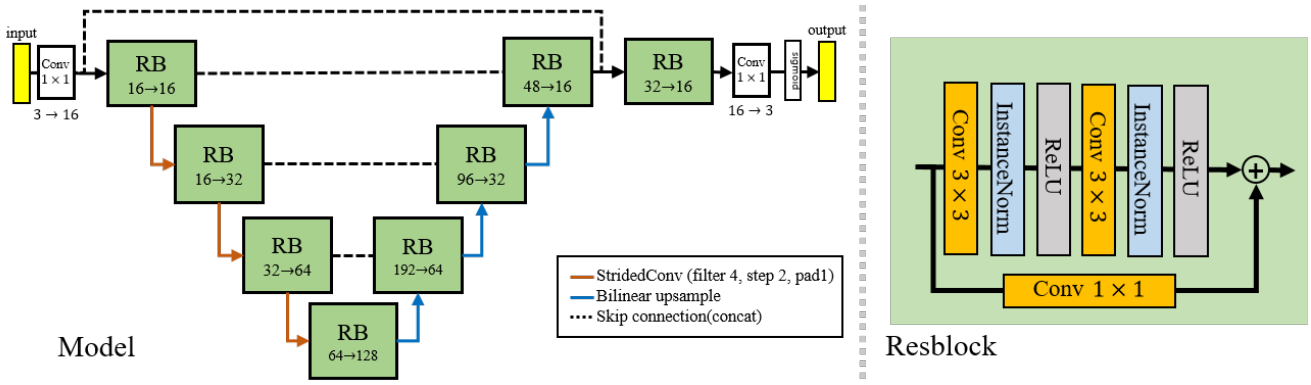

Figure 11. Our style network model architecture.

Supplementary Material

A. More details of implementation

Network architecture: For our style network f, we use a lightweight U-net which is described in the Fig. 11. We discovered that using residual block improves the content preservation and training stability. Since we train the network with only 200 iterations and the resolution of input image is relatively high, our network have to be lightweight. Therefore, we limited the maximum channel numbers to 128, and the highest resolution layer has only 16 channels. To better preserve the content information, we included another skip connection between input-output features.

Implementation details: Since CLIP model receives the images with resolution of 224×224, we resized all of the images including patch and whole image before feeding to the CLIP model. For augmentations, we use perspective augmentation which is implemented on Pytorch torchvision library. We directly used `RandomPerspective(distortion_scale=0.5)`. The random perspective function is implemented in `torchvision.transforms`.

Details for fast style transfer: For training our fast style transfer model, we used the cropped patches from DIV2K [1], and the crop size is 224. We used batch size of 4 and Adam optimizer with learning rate of 1×10^{-4}. Similar to our basic method, we used learning rate decay strategy. For augmentation, we applied random perspective augmentation for 16 times, therefore our total training patch number per iteration is $N = 64$. We also calculated directional CLIP loss L_{dir} with using patches before applying augmentations. Therefore, the loss functions of our fast style transfer is defined as:

$$L_{total} = \lambda_d L_{dir} + \lambda_p L_{patch} + \lambda_c L_c + \lambda_{tv} L_{tv},$$

which is same as the loss of our basic method. For hyperparameters, we set λ_d, λ_p, λ_c, and λ_{tv} as 1, 10, 1, and 1×10^{-4}, respectively. For threshold rejection, we set τ as 0.7.

For detailed implementation, please refer to our source code[2].

B. Additional comparison results

B.1. Comparison with other baselines

As a further comparison with other baselines, we show additional comparison results with two simple approaches: 1) to apply the existing style transfer method by using the text-retrieved image as a style image, and 2) using the image generated by the text-to-image generation model as a style image. For 1), we used the pre-trained text-to-image retrieval model[3] and chose the image with the highest CLIP score as a style image. For 2), we used the image generated through VQGAN+CLIP[4] as a style image. For image style transfer method, we chose widely used transfer model SANet [19] in both of 1) and 2).

In Fig. 12, we can observe that the outputs of baseline 1) and 2) both show reasonable results in the upper row. However, if the retrieved or generated style image is not adequate for style image (bottom row), we can see that the outputs are degraded. Although the frameworks are efficient in inference time, the results show that 1) and 2) methods have major disadvantages in that they are highly dependent on the performance of retrieval and generation models.

B.2. User study

Experiment Details: For quantitative comparison, we conducted a user study. For baseline models, we selected six different text-guided manipulation methods: StyleGAN-NADA [6], StyleCLIP [20], AdaIN+CLIP, VQGAN+CLIP, and the previous approaches 1), 2). Since the conventional

[2]https://github.com/paper11667/CLIPstyler
[3]https://github.com/rom1504/clip-retrieval
[4]https://github.com/nerdyrodent/VQGAN-CLIP

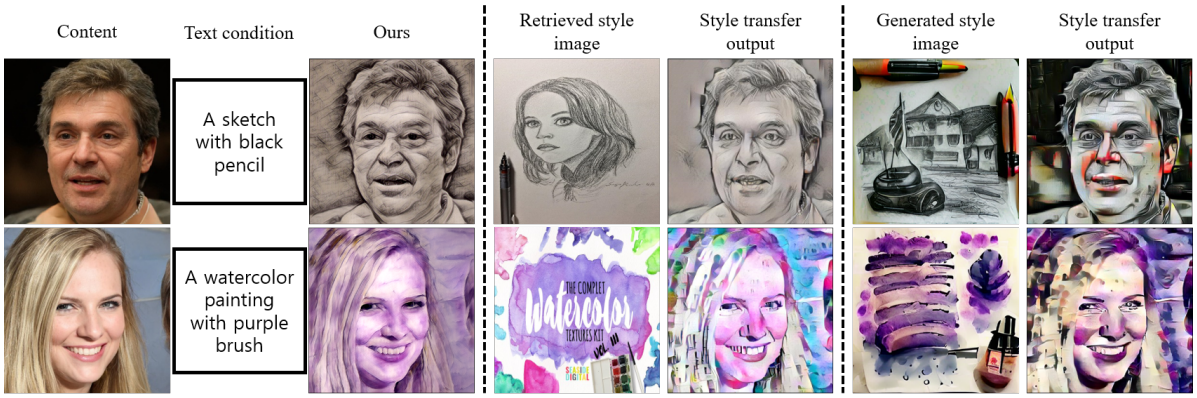

Figure 12. (a) Our results. (b) Retrieved style image and style transfer output. (c) Generated style image and style transfer output.

Methods	User study	
	Style ↑	Content ↑
Retrieve + Sty 1)	3.01	3.38
Text2Image + Sty 2)	2.80	3.61
StyleCLIP	1.47	**4.35**
StyleGAN-NADA	2.66	3.65
AdaIN+CLIP	**3.32**	2.97
VQGAN+CLIP	2.75	2.31
Ours	**3.78**	**4.18**

Table 1. User study results on various text-guided image manipulation models. The values marked with blue color refer to the best scores, and the values with red color are the second best scores.

style transfer methods use style images, we did not carry out experiments using these methods, and just selected the CLIP-based methods for fair comparison.

For user study, we generated 160 different stylized images with 10 different text conditions for each model (total 1,120 images). We used both of artistic style (e.g. "A sketch with black pencil") and non-artistic style (e.g. "Leather") text conditions. Since StyleGAN-NADA and StyleCLIP provided the models pre-trained on human face dataset, we used randomly sampled human face images as content images for fair comparison.

With generated images, we created 20 different questions. Specifically, to quantify the user preference, we asked participants questions about content preservation and the expression of textures that match the text conditions. In order to collect the detailed opinions of users, we used a custom-made opinion scoring system using Google Form. More specifically, we provided a total of 25 users with stylized images and asked them to rate the level of content preservation and text customization. We set minimum score as 1, and the maximum scores is 5. For each question, users can choose the scores among 5 different options: 1-very low , 2-low, 3-middle, 4-high, 5-very high. The 25 different subjects come from the age group between 20s and 40s, who are randomly recruited online. Then we reported the average values.

Results: Table 1 shows the user study results on various text-guided models. Our model outperformed the baseline models in content and style scores. More specifically, StyleGAN-NADA showed decent score in content preservation, but the stylization score was relatively lower than other baselines. StyleCLIP showed the best score in content preservation, but it showed the worst score in stylization. StyleGAN-NADA and StyleCLIP showed bad results in style change as they are strongly confined to the pre-trained dataset using a pre-trained StyleGAN. For baselines of 1) and 2), the method showed decent scores in content preservation, but the methods failed to successfully transfer the target style with showing relatively low values in style scores.

In AdaIN+CLIP, the model shows much better style transfer performance with scoring the second best among all the models, but it has disadvantages in content preservation. For VQGAN+CLIP, the model shows degraded performance with the worst content preservation score, which means that the model hardly reflected the shape of the input contents.

For our model, we obtained the second best score in content preservation with little difference to StyleCLIP, and obtained the best score in stylization. The user study results show that our model showed best performance considering both of content preservation and stylization.

B.3. Patch-wise CLIP score

In order to strengthen the evaluation, we measured additional quantitative metrics. To measure the correspondence between text condition and texture, we calculated cosine similarity between the CLIP embeddings of output patches and target texts. With single output image of 512×512 resolution size, we randomly cropped 64 patches which have various resolution sizes (64×64 ∼ 224×224). As a validation set, we used the same images in the user study questions.

In Tab. 2, we show the averaged CLIP scores on various models. Again, our model outperformed baseline mod-

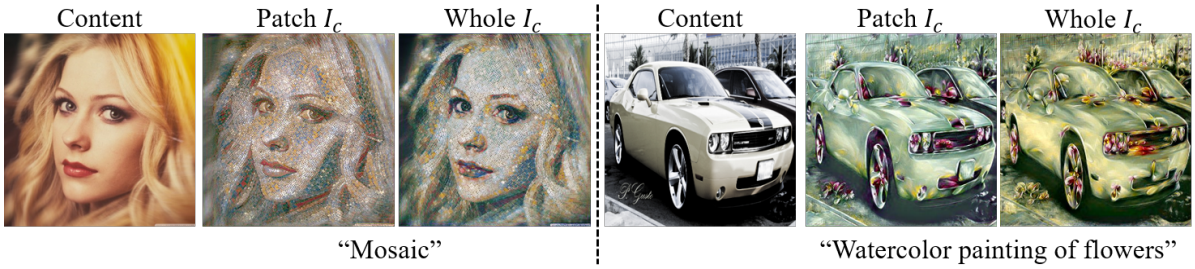

Figure 13. Comparison result with patch cropping on content image I_c. The results show there is almost no difference in perceptual quality.

Methods	CLIP score ↑
Retrieve + Sty 1)	0.2317
Text2Image + Sty 2)	0.2213
StyleCLIP	0.1982
StyleGAN-NADA	0.2252
AdaIN+CLIP	**0.2487**
VQGAN+CLIP	0.2249
Ours	**0.2515**

Table 2. Quantitative comparison results on patch-wise CLIP scores. Blue-second best, Red-best

els with showing highest CLIP scores. More specifically, StyleCLIP showed the worst performance with the lowest CLIP score, and AdaIN+CLIP scored second best among all of the baseline models. This shows that the CLIP score results have almost same tendencies as the stylization score of user study.

C. Additional ablation studies

C.1. Patch crop on content images

In order to figure out whether patch cropping on content image I_c affects the output quality, we compared the outputs using proposed CLIP loss with cropped patches from both of content and output images. In Fig. 13, we see little difference in perceptual quality, therefore we did not crop the patches from content images due to computation time. Note that to obtain the loss with patches from the content image, more images should be embedded into the CLIP space, and it increases the run time (∼10 seconds in our case) for each style transfer.

C.2. User study

For more thorough ablations study, we conducted another user study for ablation study in Tab. 3. For baselines, we use four different settings: training without L_{dir}, without proposed L_{patch}, without proposed threshold rejection, and without augmentations. For each setting, we generated 80 different stylized images with 10 text conditions (total 400 images).

Settings	Preference Score↑
no Augment	2.61
no Thresh	**3.02**
no L_{patch}	1.69
no L_{dir}	2.58
L_{total}	**3.92**

Table 3. User study for ablation study. Blue-second best, Red-best

With generated images, we created 10 different questions. In order to collect the detailed opinions of users, we used a custom-made opinion scoring system using Google Form. More specifically, we provided a total of 20 users with stylized images and asked them to score the images with overall preference. Users can choose the scores among 5 options (1-very bad, 2-bad, 3-neutral, 4-good, 5-very good). The results show that the users prefer our best setting. The 20 different subjects come from the age group between 20s and 40s, who are randomly recruited online.

The results show that our best setting outperforms other baselines with obtaining higher preference scores. We can see that without using our proposed patch-wise CLIP loss (no L_{patch}), the images showed the worst score among all of the baselines, which means that L_{patch} is the most important loss in stylization.

D. Effect of patch size

For calculating our proposed L_{patch}, we need to decide the proper patch crop size. Although we choose patch size of 128 as our default setting, we can obtain various effects with changing the patch size. The results in Fig. 14 shows the effect of different patch sizes. If we use larger patch size in training, we can have larger 'brushstroke' which can stylize the content image in a rough scale. With small patch sizes, we can apply much finer style patterns to the content images.

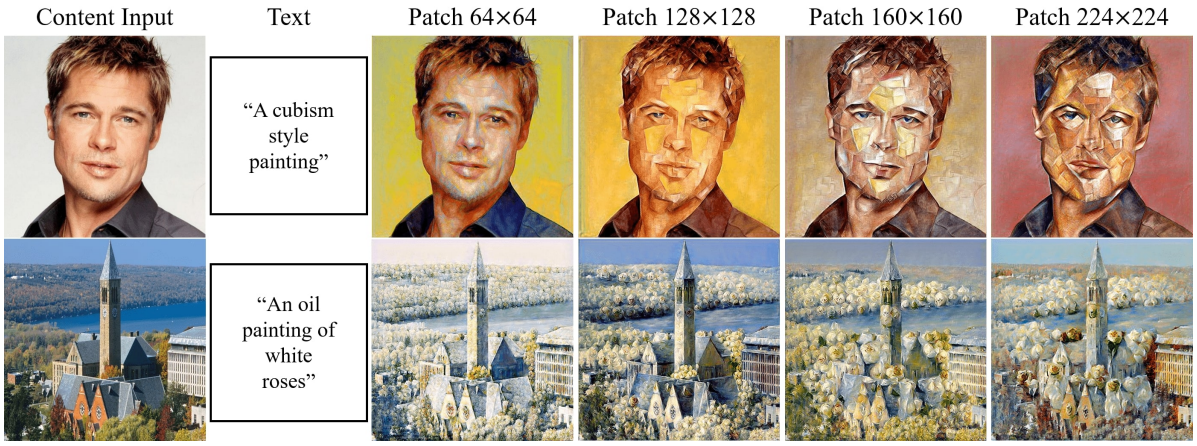

Figure 14. CLIPstyler outputs with different patch crop sizes. With a smaller patch size, we can obtain finer stylization results. With a large patch size, the outputs have large style patterns.

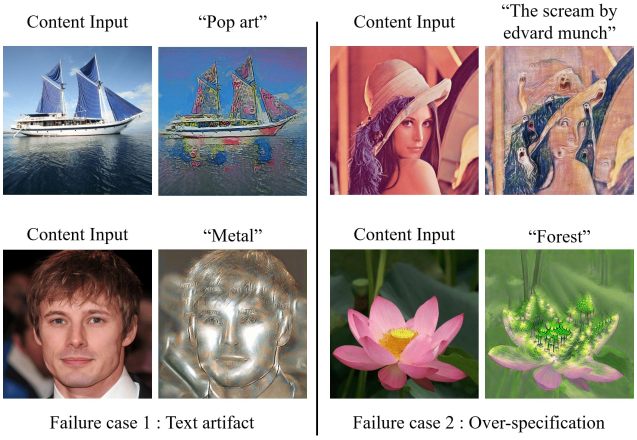

Figure 15. Failure cases of our CLIPstyler.

E. Failure cases

In this section, we show several failure cases of our model. First, as our model is based on random patch sampling and augmentations, failure cases often occur when bad patches are sampled. In the left panel of Fig. 15, we can see that there are artifacts which are caused by direct visualization of the text condition itself. For example, there are direct visualization of the text “Pop art” on the first image. Although we found out that using slightly larger value of total variation loss can partially alleviate the artifact, it is not a perfect solution. In our future work, we plan to improve the sampling algorithm so that the sampled patches are perceptually meaningful in training the network.

In the second case, output images with too specific text conditions may have unwanted patterns which are perceptually bad. In the right panel of Fig. 15, the model overly focused on reconstructing the text-conditioned target ‘object’, instead of applying the ‘texture’ of text conditions. Since this artifact is mainly caused by the embedding aspect of pre-trained CLIP model, we need to detour the effect with using different text conditions of similar meanings, such as “A painting of forest”, instead of “Forest”.

F. Additional results

More results of CLIPstyler: In Fig. 16, we show our text-guided style transfer outputs on various content images. The results clearly demonstrate that our method can apply realistic textures to content images. We also show additional results on our fast style transfer method in Fig. 17.

More comparison results: For further qualitative comparison, we provide style transfer outputs from various baseline models which use CLIP to manipulate the images. We compare the results from our model, Retrieve + Sty 1), Text2Image + Sty 2), StyleGAN-NADA, StyleCLIP, VQGAN+CLIP, and AdaIN+CLIP. Since StyleGAN-NADA and StyleCLIP are trained on human face dataset, we carried out experiments on various human face images. Part of provided results are also used in our user study questions. In Fig. 18 and 19, the results from our model show the best style transfer quality in both of content preservation and text-guided texture synthesis.

More high-resolution results: We also provide additional high-resolution output images from our fast style transfer method. The results in Figs. 20,21,22,23 shows the results. Our model can synthesize the realistic textures for high-resolution images from text conditions.

G. Limitations and Negative social impact

Although our method shows high-quality results with single text condition, there are several remaining technical issues. First, our transfer method requires network training

with each given text condition, so real-time style transfer is still not possible. While our fast style transfer method can partially address this issue, we observe that the resulting quality of the style transfer is still inferior to our default single image based optimization. Second, as shown in our failure cases, low quality results can be produced when bad patches are sampled. As a future work, we will try to solve the problems explained.

Since our method manipulate the content images with various text conditions, when the user manipulate the content images with malicious words such as obscene expressions, it may bring negative social effects. In addition, when such malicious style transfer is applied to personal photos containing sensitive information, the impact may be greater.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 8, 9

[2] Yingying Deng, Fan Tang, Weiming Dong, Wen Sun, Feiyue Huang, and Changsheng Xu. Arbitrary style transfer via multi-adaptation network. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2719–2727, 2020. 2

[3] Michael Elad and Peyman Milanfar. Style transfer via texture synthesis. *IEEE Transactions on Image Processing*, 26(5):2338–2351, 2017. 2

[4] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021. 7

[5] Kevin Frans, LB Soros, and Olaf Witkowski. Clipdraw: exploring text-to-drawing synthesis through language-image encoders. *arXiv preprint arXiv:2106.14843*, 2021. 4

[6] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021. 3, 6, 9

[7] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016. 1, 2, 4, 5, 6

[8] Kibeom Hong, Seogkyu Jeon, Huan Yang, Jianlong Fu, and Hyeran Byun. Domain-aware universal style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14609–14617, 2021. 2

[9] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Int. Conf. Comput. Vis.*, pages 1501–1510, 2017. 2, 5, 7

[10] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 2

[11] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 6

[12] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4401–4410, 2019. 3

[13] Dmytro Kotovenko, Matthias Wright, Arthur Heimbrecht, and Bjorn Ommer. Rethinking style transfer: From pixels to parameterized brushstrokes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12196–12205, 2021. 2

[14] Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip HS Torr. Manigan: Text-guided image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7880–7889, 2020. 3

[15] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3809–3817, 2019. 2

[16] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. *arXiv preprint arXiv:1705.08086*, 2017. 2

[17] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 453–468, 2018. 2

[18] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6649–6658, 2021. 2, 5

[19] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5880–5888, 2019. 2, 5, 9

[20] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021. 2, 3, 6, 9

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 2, 3, 4

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4

[23] Jan Svoboda, Asha Anoosheh, Christian Osendorfer, and Jonathan Masci. Two-stage peer-regularized feature recombination for arbitrary image style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13816–13825, 2020. 2, 5

[24] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6924–6932, 2017. 2

[25] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324, 2018. 3

[26] Wenju Xu, Chengjiang Long, Ruisheng Wang, and Guanghui Wang. Drb-gan: A dynamic resblock generative adversarial network for artistic style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6383–6392, 2021. 2

[27] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9036–9045, 2019. 2

[28] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915, 2017. 2, 3

[29] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1947–1962, 2018. 2, 3

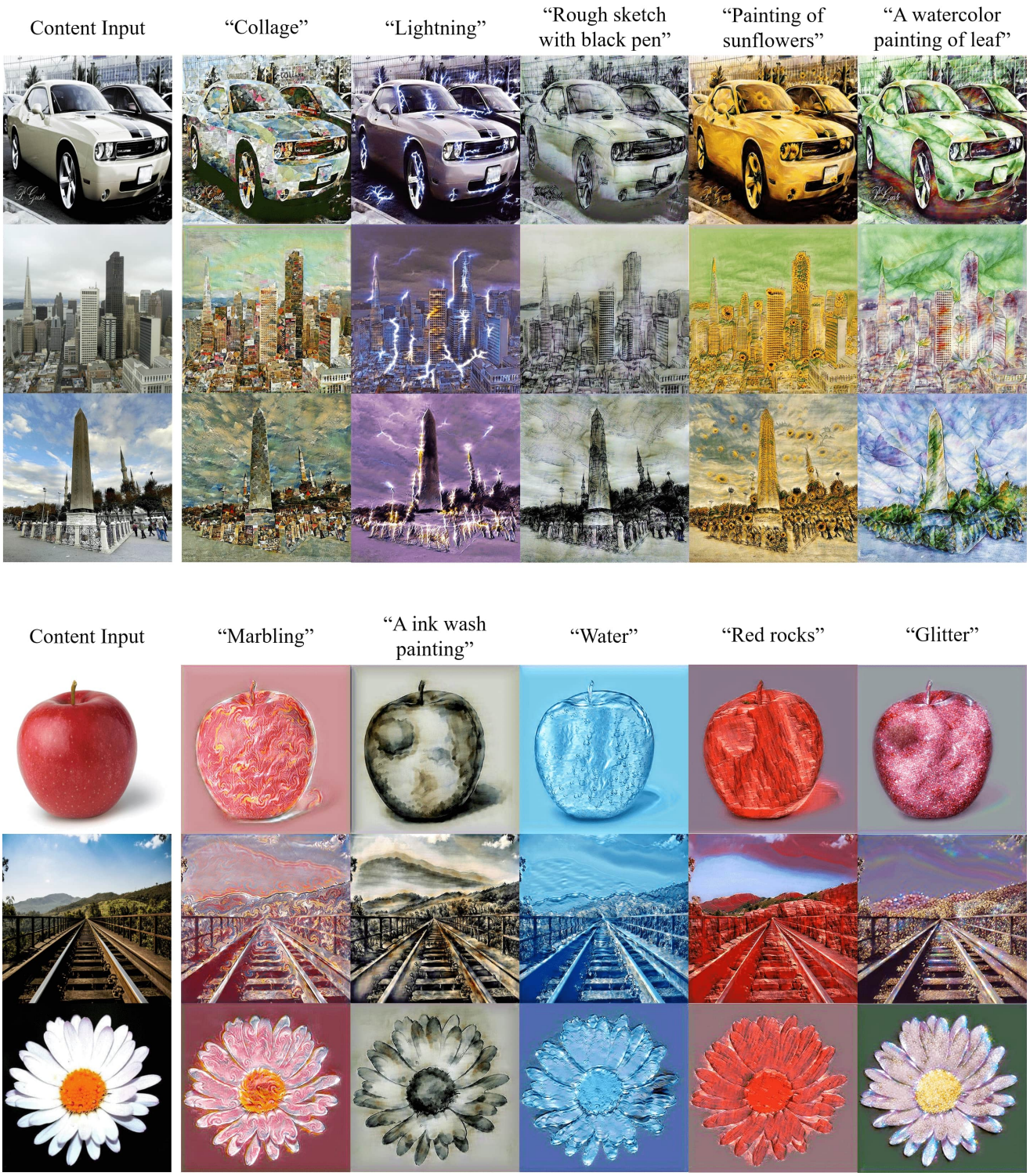

Figure 16. Additional results of our CLIPstyler.

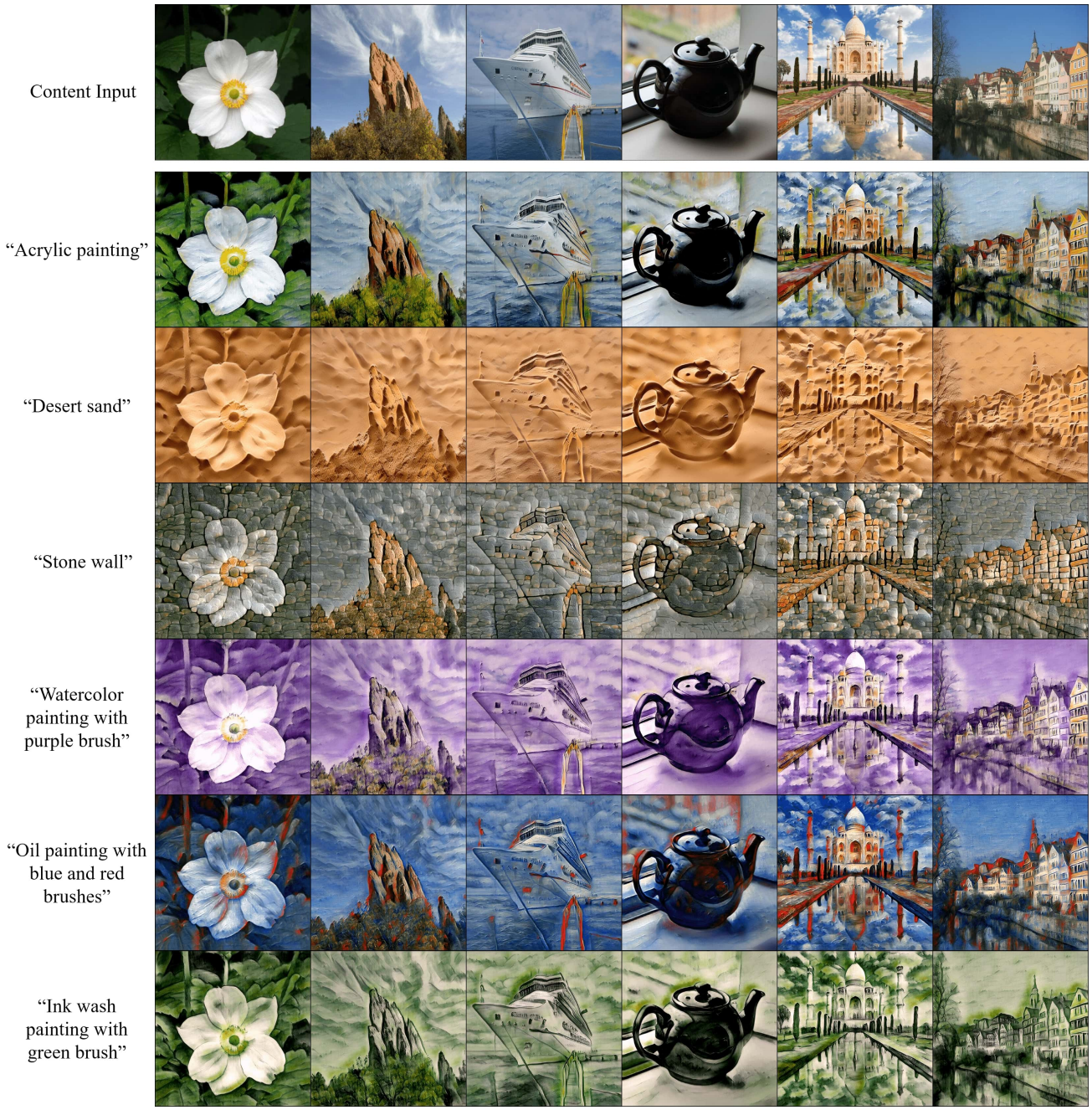

Figure 17. Additional results of our fast style transfer method.

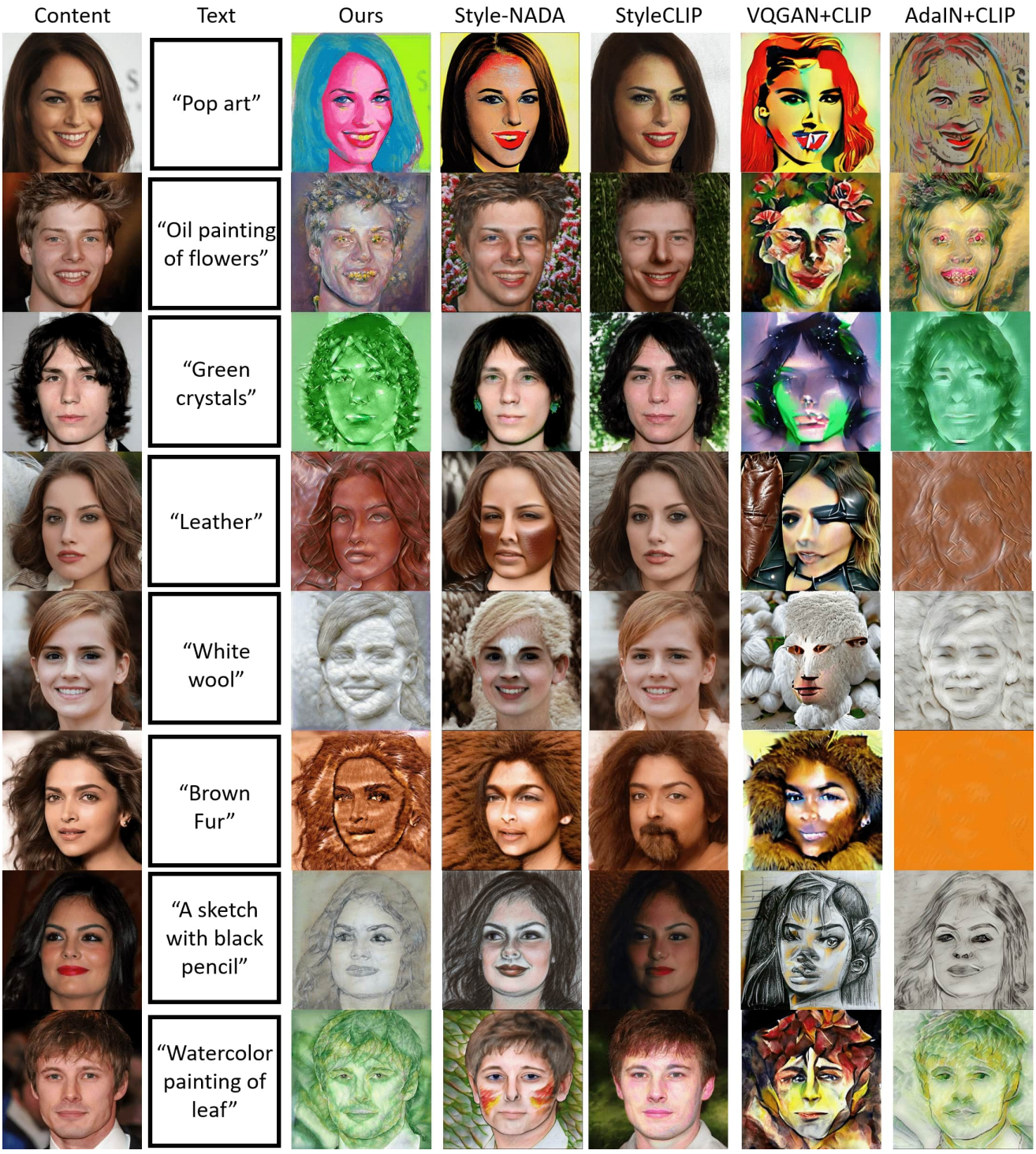

Figure 18. Additional comparison with baseline methods.

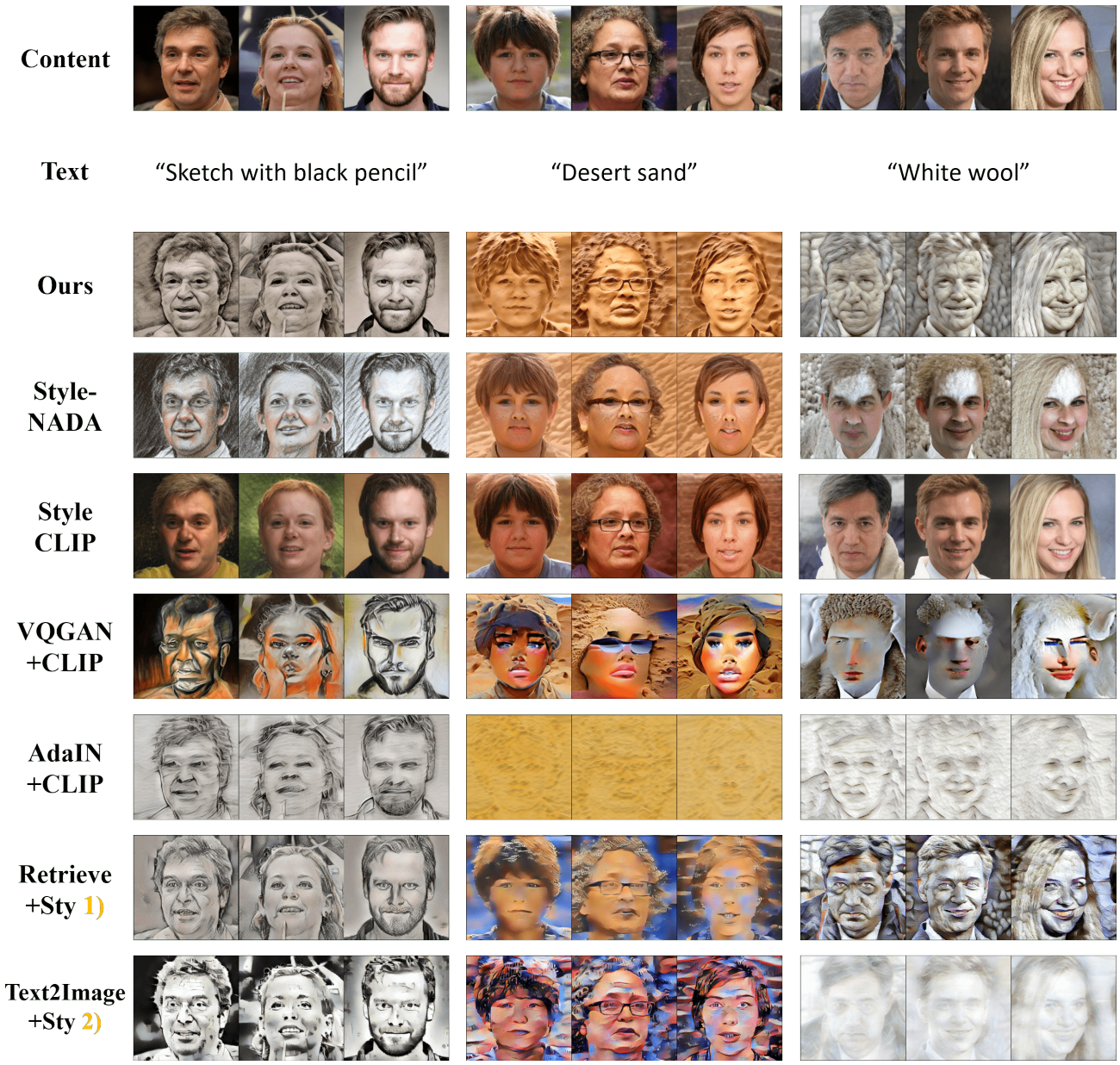

Figure 19. Additional comparison with baseline methods.

Figure 20. High resolution results from our fast style transfer method. The resolution size is 1920×2580. Up: Content image. Down: Output with text condition of "Watercolor painting".

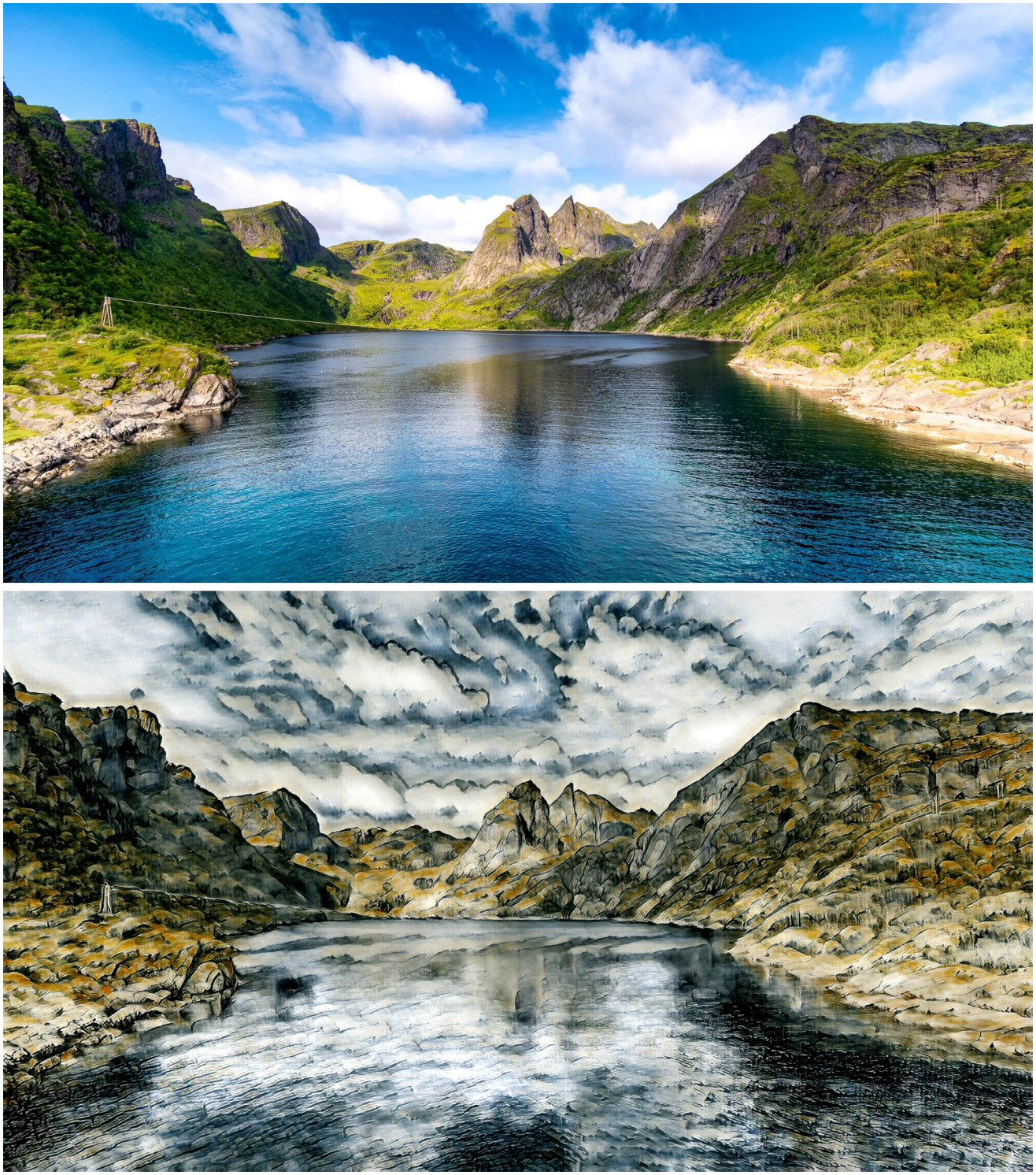

Figure 21. High resolution results from our fast style transfer method. The resolution size is 1128×2000. Up: Content image. Down: Output with text condition of “An ink wash painting”.

Figure 22. High resolution results from our fast style transfer method. The resolution size is 1600×2400. Up: Content image. Down: Output with text condition of "A fauvism style painting with bright color".

Figure 23. High resolution results from our fast style transfer method. The resolution size is 1656×2200. Up: Content image. Down: Output with text condition of "Snowy".