The Visual Computer (2021) 37:2341–2353
https://doi.org/10.1007/s00371-020-01990-7

ORIGINAL ARTICLE

Learning color space adaptation from synthetic to real images of cirrus clouds

Qing Lyu[1] · Minghao Chen[1] · Xiang Chen[1]

Accepted: 2 October 2020 / Published online: 14 October 2020

Abstract
Cloud segmentation plays a crucial role in image analysis for climate modeling. Manually labeling the training data for cloud segmentation is time-consuming and error-prone. We explore to train segmentation networks with synthetic data due to the natural acquisition of pixel-level labels. Nevertheless, the domain gap between synthetic and real images significantly degrades the performance of the trained model. We propose a color space adaptation method to bridge the gap, by training a color-sensitive generator and discriminator to adapt synthetic data to real images in color space. Instead of transforming images by general convolutional kernels, we adopt a set of closed-form operations to make color-space adjustments while preserving the labels. We also construct a synthetic-to-real cirrus cloud dataset SynCloud and demonstrate the adaptation efficacy on the semantic segmentation task of cirrus clouds. With our adapted synthetic data for training the semantic segmentation, we achieve an improvement of 6.59% when applied to real images, superior to alternative methods.

Keywords Color space · Synthetic-to-real · Domain adaptation · Cirrus clouds · Segmentation · Style transfer

1 Introduction

Cloud images are widely used in climate modeling, weather prediction, renewable energy generation, and satellite communications [1–4]. Digital analysis of clouds and their features is necessary for these subjects. One of the first steps in cloud image analysis is cloud segmentation. Previous learning-based cloud segmentation methods are supervised and require a large number of training images with manually labeled ground-truth [5–8]. Since manual labeling is time-consuming and error-prone, we explore to train cloud segmentation networks with synthetic images.

Training on synthetic images has become increasingly popular in vision tasks, such as object detection [9–12], viewpoints estimation [13–16], and semantic segmentation [17]. For example, photo-realistic rendering on 3D models can provide an accurate pixel-level annotation for each object in the scene, which eliminates the labeling cost for image segmentation learning. Under such circumstances, several synthetic datasets like Virtual KITTI [11] and SYNTHIA [18] have been generated for training and evaluating vision models. When directly applying the synthetic dataset for real-world image tasks, performance degradation arises from the inherent domain gap. Such gaps may result from many reasons, e.g., differences in geometric details, textures, backgrounds, and lightings. Some versatile domain transfer models are proposed to transfer synthetic images to real images [19,20]. However, the methods often break the preservation of original labels, especially for the pixel-level prediction tasks like semantic segmentation. Though there is a dedicated loss formulation for the preservation of images contents, segmentation labels cannot be preserved entirely due to the loss terms balancing.

We propose another strategy to solve the problem. Instead of loss design, we constrain the adaptation in color space. It would never destroy the segmentation label, which is crucial for tasks requiring accurate label-preserving. The reasons for choosing to adapt synthetic data in color space are twofolds. First, we think that dividing the factors like color, shape, and textures provides a better opportunity for optimizing each subtask. Through division, we prevent the shape from

✉ Xiang Chen
xchen.cs@gmail.com

Qing Lyu
lyuqing@zju.edu.cn

Minghao Chen
minghaochen01@gmail.com

[1] The State Key Lab of CAD&CG, Zijingang Campus, Zhejiang University, Hangzhou 310058, Zhejiang, China

(a) rendering images (b) real images (c) transferred images

Fig. 1 Color Space Adaptation. Compared with the original photo-realistic rendering images (**a**), the transferred images (**c**) after the color space adaption are visually more consistent with the real images (**b**). Here we only present rendering images of similar colors. The full set of rendering images is diverse with a wide color range. Please refer to the "Appendix" for more results

changing and preserve original segmentation labels. Second, among other factors, the difference in color space plays a vital role. Since the synthetic data are produced from a photo-realistic 3D renderer, their textures are similar to real images. However, the synthetic data are raw images, while the real images are often post-processed either by digital cameras or photography software, which leads to a large discrepancy in their color space. Therefore, we focus on the adaptation in color space.

In this paper, we propose a color space adaptation framework to transfer a synthetic dataset to a given target of real images. Specifically, we adopt a set of basic operations for color space adjustments. These closed-form operations merely affect color representation without incurring any perceptual difference and label damage. We leverage such properties to generate sufficient adversarial examples for pretraining an effective classifier that is only sensitive to colors, but not to shapes and textures. Using the pre-trained color-sensitive classifier to evaluate the difference of color representations, we can train a generative network to adapt the synthetic dataset to the real images. Furthermore, we build a synthetic-to-real cirrus cloud dataset SynCloud to quantitatively evaluate the efficacy of the proposed color space adaptation framework for cloud image segmentation. The SynCloud dataset consists of synthetic cloud images and unlabeled real images for adaptation training. The segmentation labels for synthetic images are automatically generated, while the labels of real images are manually annotated. All synthetic images are produced by physically based rendering and are thus photo-realistic. Figure 1a displays some exemplars of our synthetic images. Our experiments show that the transferred dataset has significantly improved the segmentation performance compared with the original one. We also demonstrate some other applications of the cirrus cloud segmentation results, including 3D reconstruction, matting, and composition, for some interesting extensions.

To summarize, the main contributions of our work are:

1. We propose a simple yet robust framework for color space adaptation from synthetic data to real images while preserving the pixel-level labels.
2. We construct a synthetic-to-real cirrus cloud dataset SynCloud to validate the efficacy of our adaptation method.
3. We apply our method to tasks like style transfer and photograph post-processing to show its potential.

2 Related work

Generative adversarial networks Since proposed by Goodfellow et al. [21] in 2014, generative adversarial networks (GANs) have attracted many research interests and achieved success in various image generation tasks such as the image super-resolution [22,23]. DCGANs [24] embed convolutional networks (CNNs) in the GANs framework for image representation learning. Recently, researches have been exploring the possibility of applying GANs to the domain transfer problems like the image-to-image translation [25–29]. Zhu et al. [26] propose CycleGAN with a novel cycle-consistent constraint for unpaired image-to-image translation. CoGAN [28] trains two coupled GANs to synthesize pairs of corresponding images. With the development of GANs, domain adaptation from text to the image has also been explored [30–32].

Synthetic-to-real domain adaptation Previously, visual domain adaptation has been focused on the transformation of latent distributions in feature space [9,33]. With the remarkably generative capability of GANs, new methods begin to convert synthetic images in pixel-level. Bousmalis et al. [19] combine a content-similarity loss with GANs to generate contexts around synthetic subjects mimicking the target domain. CyCADA [20] combines cycle-consistency constraints with GANs to transfer the synthetic images at both pixel-level and feature-level. This line of work usually requires a dedicated loss formulation to balance the preservation of images contents. We avoid the loss design by using exactly content-preserving operations to generate color space variants of real images for training the classifier. RenderGAN [34] inserts a parameterized 3D model with a cascade of image augmentations into the GANs to imitate the tag images rendering process. We focus on the color space adaptation of a synthetic dataset of diverse shapes for semantic segmentation, rather than rendering a template model from scratch for tag recognition.

Color transformation Learning color transformations, e.g., enhancement, stylization and colorization, from given image exemplars is a challenging problem [35–38]. There are mainly two types of automatic methods for color enhancement: example-based and learning-based. Example-based methods directly transfer the color of an example image to another one by optimization [39–41]. Learning-based color enhancement is another dominant stream which learns a mapping function from the source images to the target images with training data.

Learning-based color enhancement can be accomplished in two ways: by local transformations and by globally parameterized transformations. For local transformations, images with the desired pixel colors and styles are directly produced by a deep convolutional neural network [42–45]. These methods usually work best under a fixed image resolution and do not generalize well to images with arbitrary high resolutions, which lead to deteriorated results or unaffordable memory consumption. Recently, globally parameterized transformations have been proposed [46–49], in which the deep neural networks determine a series of parametric operations to enhance the input image. These methods can be further divided according to the usage of paired data [48] or unpaired data [46,47,49], where the requirement of paired data often limits its usage. Hu et al. [47] generates a retouching sequence using reinforcement learning (RL) with an adversarial reward. Park et al. [46] proposes a distort and recovery training scheme for unpaired learning using RL. While RL-based methods are powerful for learning complex behaviors, they are sensitive to the initial values and often requires a careful fine-tuning of the hyper-parameters to stabilize the training process. In contrast, our method does not rely on RL and is both robust and easy-to-use. Moreover, a direct application of these methods to our dataset would lead to poor performance due to a different kind of domain gap. The rendering images and the real images in our dataset does not share a common set of underlying scenes, which makes the differences in shapes and textures, rather than colors, easily dominate the classification. See the comparison in Sect. 5.1 for a more detailed discussion.

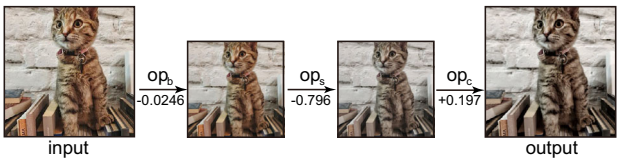


Fig. 2 Adjustment process. A predicted sequence of color adjustment operations for transferring artist style A to artist style B on the Pexels dataset

Image segmentation As a fundamental task in computer vision, pixel-level labeling ushers in a new development in semantic segmentation [50–54] and instance segmentation [55]. Synthetic data have also been generated for training segmentation models [17]. We adapt the synthetic data to real-world images to bridge their domain gap in color space and consider this an effective way to improve segmentation performance.

3 Method

The color space adaptation is a learning approach that is trained on unpaired images. Given the synthetic image dataset X_s as source and the real image dataset X_r as target, we expect to learn a function $\mathsf{g}_{s\to r}$ to adapt X_s toward X_r in color space, while preserving their textures and shapes. Figure 1 shows the color space adaptation from synthetic cirrus cloud images to real images.

To limit the adaptation only in color space, we transfer the images merely through a set of basic image processing operators, *i.e.,* adjusting the brightness, saturation, and contrast of the images, instead of through general convolutional kernels. We define the composition of the set of basic image processing operators as the adaptation operator $\mathsf{g}_{s\to r}(x;\alpha) := \mathsf{ops}(x;\alpha)$, where x is the input image, α is the set of parameters for image processing, and ops is the combination of basic color adjustment operators. In our case,

$$\begin{aligned}\mathsf{g}_{s\to r}(x;\alpha) &= \mathsf{ops}(x;\alpha)\\ &= \mathsf{op}_c(\mathsf{op}_s(\mathsf{op}_b(x;\alpha_b);\alpha_s);\alpha_c),\end{aligned} \tag{1}$$

where $\alpha = \{\alpha_b, \alpha_s, \alpha_c\}$. Figure 2 shows an example of the adjustment process with ops. The closed-form expressions of op_b, op_s and op_c can be found in "Appendix A".

Formally, we have an optimization problem defined as

$$\min_{\alpha} \sum_{x_s \in X_s} \sum_{x_r \in X_r} \mathsf{d}(\mathsf{g}_{s\to r}(x_s; \alpha), x_r) \tag{2}$$

where x_s and x_r is an image in X_s and X_r, respectively, and d is a function measuring the distance between two images in their color space representations.

Here we pretrain a discriminator to measure the distance d and minimize a generator to produce adjustment parameters α for every image.

3.1 Generative adversarial network

GANs are proposed for estimating generative models via an adversarial process [21]. In the general framework of GANs, there are two models trained alternatively: A generative model G that captures the data distribution and a discriminator model D that estimates the probability that a sample comes from the training data rather than G. Both models are realized as convolutional neural networks in our case. The training procedure for G is to maximize the probability of D making a wrong prediction, while D is trained to minimize that probability. As a result, D and G play the following two-player minimax game with value function $V(G, D)$:

$$\min_{\theta_g} \max_{\theta_d} V(G, D) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x; \theta_d)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z; \theta_g); \theta_d))] \tag{3}$$

where x is training data and z is noise. $\mathbb{E}$ denotes the expectation. Here we use the notion $x \sim p_{data}(x)$ for samples x drawn from a corresponding probability data distribution p_{data}. The same also applies for $z \sim p_z(z)$. $G(z; \theta_g)$ represents a function with parameters θ_g that maps from noise $z \sim p_z(z)$ to data space $x \sim p_{data}(x)$. For G represented as convolutional neural networks, θ_g are parameters of convolutional kernels in the network. $D(x; \theta_d)$ with parameters θ_d estimates the probability that x came from the data rather than being produced by G. Again, θ_d are the parameters of a convolutional neural networks D. We will omit $x \sim p_{data}(x)$ and $z \sim p_z(z)$, as well as θ_g and θ_d for simplicity below.

Color sensitivity The standard GANs train the discriminator D by classifying the transferred synthetic images $\mathsf{g}_{s\to r}(X_s)$ and the real images X_r. However, this quickly results in a color-insensitive discriminator due to the dominance of perceptual content differences, (e.g., shapes, textures) between

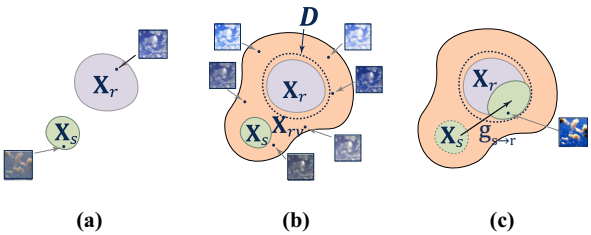


Fig. 3 Method. The synthetic images X_s and real images represent distinct subsets X_r in the color space (**a**). Color adjustment operations are used to generate random variants X_{rv} of the real images X_r. A discriminator D is trained to classify X_{rv} and X_r well in color space (**b**). Next, the generator is trained to adapt the synthetic images X_s under the guidance of the pretrained discriminator D. The transferred synthetic images $\mathsf{g}_{s\to r}(X_s)$ after the adaptation is much more close to the real images X_r in the color space (**c**)

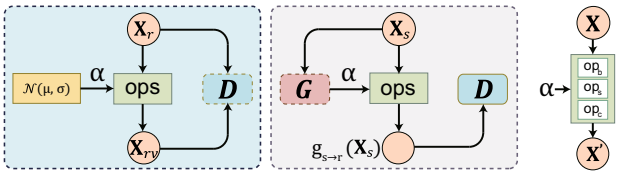


Fig. 4 Pipeline. We learn the discriminator D (left) by pretraining on X_r and X_{rv} and then train the generator G (middle) under the guidance of the pretrained D. The module of color adjustment operations ops (right) is shared between the two stages. It takes in Gaussian random samples as parameters to generate adversarial data for training the discriminator and takes in the generator output as parameters to generate adapted synthetic images for training the generator

X_s and X_r. We eliminate the undesired effects from content differences by purposely constructing a training dataset whose images are different only in the color space. Specifically, we generate the color variants of real images X_r as X_{rv} by using the image processing operators ops. When training the discriminator D, color variants X_{rv} are seen as *False* and the original real images X_r are seen as *True*. By classifying X_{rv} and X_r, we succeed to learn a color space discriminator D with its classification boundary found in color space (Fig. 3b). While we prevent D from directly observing the synthetic data X_s during training, the color space discriminator D still works for synthetic data X_s since the color variants X_{rv} have covered color space of X_s (Fig. 3b). Under the guidance of the pretrained D, we can learn the generator G to adapt the color space information of X_s correctly (Fig. 3c).

3.2 Color space discriminator

As shown in Fig. 4 left, to embody the discriminative function d, we choose to sample sufficient variants of X_r in color space as X_{rv} and consequently train a classifier to distinguish between X_r and the adversarial dataset X_{rv}. Therefore, we adopt the composition of a set of basic image processing

operators ops (Fig. 4 right), to randomly adjust the brightness, saturation and contrast of images in X_r. These low-level operators ensure that the learned classifier is only effective in distinguishing the color representations of the images, but insensitive to their high-level perceptual difference. We use D as the discriminative function $\mathsf{d}(x_{rv}, x_r)$ and determine its parameters by optimizing the loss function:

$$L_D = \mathbb{E}\left[-\log(D(x_r))\right] + \mathbb{E}\left[-\log\left(1 - D(x_{rv})\right)\right] \tag{4}$$

where $x_{rv} = \mathsf{ops}(x_r; \alpha_{\mathsf{randn}})$ is an image in dataset X_{rv} and $\alpha_{\mathsf{randn}} \sim \mathcal{N}(\mu, \sigma)$. We compute the mean μ and standard deviation σ of $\mathcal{N}$ as

$$\mu = 0, \quad \int_{-1}^{1} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(\alpha-\mu)^2}{2\sigma^2}} = p \tag{5}$$

where p is the probability that α falls in the range $[-1, 1]$. We set $p = 0.99$ to ensure that α is in the usual range of color adjustment in most cases.

3.3 Color adjustment parameter generator

As shown in Fig. 4 middle, the color adjustment operations ops are embedded into $\mathsf{g}_{s\to r}$, by taking the output of the generator G as the adjustment parameters α. In some sense, this mimics the post-processing steps taken by photographers in the real world to retouch raw images. Specifically, the generator is composed of three components with the same network structure, i.e., $G = \{G_b, G_s, G_c\}$. Each component takes an input image and produces an adjustment parameter for an operator in ops respectively, $\alpha = \{\alpha_b = G_b(x), \alpha_s = G_s(x), \alpha_c = G_c(x)\}$. Finally, the ops together with the produced α are applied on the input image to generate an adapted one. We train the generator G by maximizing the score of the adapted images $\mathsf{g}_{s\to r}(X_s)$ obtained on the pretrained discriminator D (fixed in this stage). The loss function is:

$$L_G = \mathbb{E}[\log(1 - D(\mathsf{g}_{s\to r}(x_s; G(x_s))))] \tag{6}$$

3.4 Training

Our training process has two stages. In the first stage (Fig. 4 left), we feed the adversarial data X_{rv} and real images X_r into the D to train its parameters by following Eq. 4. In the second stage (Fig. 4 middle), we fix the pretrained D and feed the synthetic images X_s into the G to train its parameters by following Eq. 1. However, with our color-sensitive discriminator, we need not follow the standard GANs training process to repeat the two-stages alternatively, which simplifies the GAN training and makes our method robust.

4 Architecture and training data

4.1 Datasets

The SynCloud dataset We construct the SynCloud dataset for quantitative evaluation on the semantic segmentation task of the cirrus cloud images. It contains synthetic cloud images naturally attached with semantic segmentation labels (tagged as *Synthetic Part*) and real cloud images (tagged as *Real Part*).

- *Synthetic part* We construct 11, 654 synthetic images, 7, 969 images of which for training, and the left 3, 685 images for testing, based on the shape modeling and photo-realistic 3D rendering methods. First, we computationally generate 624 3D volume data with heterogeneous grid densities, through fluid simulation and image-based reconstruction. Then, we use a physically based renderer Mitsuba [56], to generate photo-realistic images, as well as the segmentation labels, from those volume data of cirrus clouds. We randomly sample the parameters of the volume rendering, e.g., scene lighting, reflections, lens focal length, and camera viewpoint, to extend the coverage range. The generated synthetic images all have 512×340 resolution.[1]
- *Real part* This part of the data contains 1381 unlabeled real images for adaptation training and 427 real images with manually annotated labels for segmentation training. Among annotated images, 273 of them are in the training set, and the left 154 images are in the test set. We manually annotate the images using the Magnetic Lasso tool in Adobe Photoshop. Due to the complex boundaries of cirrus, manual labeling is exceptionally time-consuming. It takes around 10-20 min per image.

Photograph datasets for style transfer For the style transfer and photograph post-processing task, we utilize two sources of training data:

- *The MIT-Adobe FiveK Dataset* It is a photograph dataset consisting of 5000 RAW images and their retouched versions created by five professional experts [57]. In this work, we choose 4000 input RAW images and 4000 retouched images of expert A as training data and use another 1000 input RAW images as test data.
- *The Pexels Dataset* We crawled professionally retouched photos from two artists on www.pexels.com. The dataset consists of 284 images from artist A and 425 images from artist B. We use 90% of them for training and 10% for testing. The two sets of data have relatively consistent styles. One of them is bright and vivid, and the other

[1] We will release the cirrus clouds dataset, including all the volume data, rendering settings and rendering results.

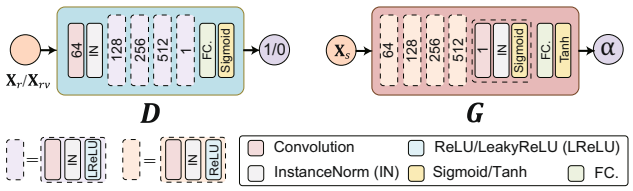


Fig. 5 Architecture. The network architectures of the D and G. All convolution layers have filters of kernel size 4 and strides 2. The number of output filters is written in the block

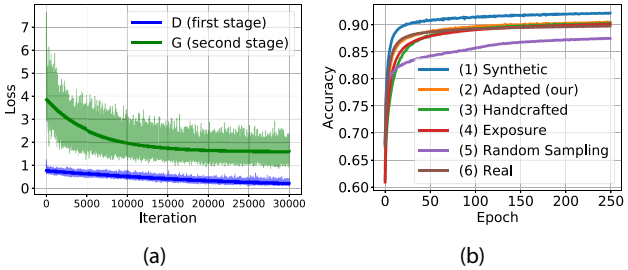


Fig. 6 Training process. **a** Losses of the discriminator and the generator, during the first and second stage of the color space adaptation training. **b** The FCN accuracies during the semantic segmentation training

is more cold and tranquil. We show some exemplars in Fig. 13.

All images in the MIT-Adobe FiveK dataset and the Pexels dataset are resized with a fixed width of 256 and randomly cropped to 256 × 256 resolution.

4.2 Network architecture

Discriminator We design the network structure of the discriminator D (Fig. 5 left) by following the guidelines of DCGANs [24]. Specifically, we use strided convolutions to replace pooling layers and choose Leaky ReLUs (with slope coefficient 0.2) as the activation function. We also use instance normalization instead of batch normalization. The convolution group, including a convolution layer, an instance normalization layer, and a leaky ReLU layer, are repeated four times. At last, a fully connected layer is connected with a sigmoid function as the output.

Generator We design the network structure of the generator G (Fig. 5 right) by following the guidelines of DCGANs [24]. The convolution group, including a convolution layer, an instance normalization layer, and an activation function layer, are repeated five times. At last, a fully connected layer is connected with a tanh function as the output.

4.3 Training details

All our implementations are based on the Keras library [58] with the TensorFlow [59] backend. A desktop machine with Ubuntu 16.04 and a Geforce 1080Ti GPU is used for training the neural networks.

Adaptation training For all the experiments, we choose 24 as the batch size. The input images are zero-centered and rescaled to $[-1, 1]$. The network weights are initialized with He initialization [60]. We adopt the Adam optimizer [61] and train all the networks from scratch with $lr = 1e{-}3$ and $\beta_1 = 0.5$. First, we train the discriminator for 4 epochs. Then the generator is trained for 4 epochs with the pretrained discriminator frozen. The loss change during the training process is shown in Fig. 6a. When training D, we get a loss value around 0.32 for training data and 0.25 for test data. When training G, we get a loss value of around 1.39 for training data and 1.83 for test data.

Partial results of the synthetic images before and after color space adaptation are shown in Fig. 1. Visually, the adapted images are much more similar to the color style of real images. The sliced-Wasserstein distance[2] (SWD) metric [63] between the synthetic images and the real images has reduced about 55% after the adaptation.

Segmentation training The FCN-8s architecture [50] is used to perform the semantic segmentation task. We fine-tune network with the VGG-16 model and adopt the SGD optimizer with the learning rate of 1e−5 and the momentum of 0.9. Mean squared error is used as the loss function. We iterate the training process for 250 epochs and achieve a stable training accuracy of around 0.9. The accuracy change during the training process is shown in Fig. 6b.

5 Experiments

5.1 Evaluation

Baseline and comparison We quantitatively evaluate the color space adaptation method through the performance of semantic segmentation tasks. Specifically, we prepare six datasets for training the segmentation network:

(1) *Synthetic images (from Synthetic Part).* 7969 *synthetic images with automatic generated segmentation labels through physically based rendering.*
(2) *Adapted synthetic images (our).* 7969 synthetic images adapted with our method.

[2] SWD owns similar properties to the Wasserstein distance but simpler to compute. It is widely used in various applications, including generative modeling and general supervised/unsupervised learning, to measure the quality of generative images [62].

(3) *Augmented synthetic images by a handcrafted approach.* 7969 synthetic images augmented by fitting a Gaussian model and shifting the features. We show details in "Appendix B".
(4) *Enhanced synthetic images by Exposure* [47]. 7969 synthetic images retouched by the method of Exposure.
(5) *Augmented synthetic images by random sampling.* 21252 augmented images enhanced from 7969 synthetic images by randomly sampling the color adjustment operation parameters.
(6) *Real images* (from *Real Part*). 273 real images with manually annotated segmentation labels.

Finally, we use 154 unique test real images (from *Real Part*) to evaluate the segmentation performance of the FCNs trained on the above six datasets. We use mean IoU scores to measure the segmentation performance. Mean IoU is the intersection over the union of the prediction and ground truth in the segmentation results. A higher score means better performance.

Table 1 shows the evaluation results. Compared with the original synthetic images, the randomly augmented synthetic images have only less than 2% relative improvement. On the other hand, using the adapted synthetic images as training data has improved the segmentation performance from 0.68 to 0.75 (about 10% relative improvement), proving the efficacy of our color space adaptation method. The performance of adapted images is very close to that of the real images. It demonstrates that replacing real images with synthetic images to eliminate manual labeling efforts is promising, at least partially. Using the combined dataset with the adapted images and real images, we obtain a segmentation performance of 0.7765, which is higher than merely using the real images.

We also execute a comparison with a state-of-the-art method for color enhancement [47]. Table 1 shows that directly applying the method to our dataset does not perform well on the segmentation task. The method also adopts an adversarial loss based on a discriminator to guide the inference learning of the color adjustment process. Their source and target images for training the discriminator share a common set of underlying scenes, which naturally form a color-sensitive dataset and result in a color-sensitive discriminator. However, in our dataset the synthetic and real images have disjoint sets of underlying scenes, whose differences in shapes and textures, but not colors, would easily dominate the classification. This inherent gap in our synthetic-to-real dataset together with the complexity of the reinforcement learning makes it challenging to obtain a color-sensitive discriminator, which explains the failure. The segmentation performance of 0.6176 is close to our ablation study result of 0.6247 with a color-insensitive discriminator. This also justifies the necessity of our decoupled construction of the color variants X_{rv} for training the discriminator.

Table 1 mean IoU for all datasets

Training dataset	Mean IoU (test)
(1) Synthetic	0.6829
(2) Adapted (our)	**0.7488**
(3) Handcrafted	0.7118
(4) Exposure [47]	0.6176
(5) Random sampling	0.6996
(6) Real	**0.7668**

Bold indicates the best score and the ground-truth

Table 2 Mean IoU for different adaptation options

Choice of p in Eq. 5			Color-sensitive D		mean IoU
0.9	0.99	0.999	With	Without	
✓			✓		0.7359
	✓		✓		**0.7488**
		✓	✓		0.7458
	✓			✓	0.6247
		✓		✓	0.6753

Bold indicates the best score and the ground-truth

Ablations We adopt the following two ablations:

(1) *Choice of p in* Eq. 5. We choose different p of 0.9, 0.99, and 0.999 to control the probability of the adjustment parameter α in the range $[-1, 1]$. Segmentation performance in Table 2 (the first three lines) shows that the change of p brings slight variations to the mean IoU scores. In practice, we have found that $p = 0.99$ is a reasonably good choice for all the experiments.
(2) *With/without the color-sensitive D*. We train a color-sensitive D by following the design in Sect. 3.1, and a color-insensitive one by classifying between mapped source images and target images and putting it into the back-propagation. Table 2 (the last two lines) shows that with a color-insensitive D, the segmentation performance quickly degrades, to be even worse than using non-adapted images. As discussed in Sect. 3.1, a possible reason is that once the discriminator starts to observe the synthetic images, the perceptual information like shapes and textures could gradually dominate the classification.

5.2 Applications

Cirrus segmentation We present the method mainly for the image segmentation of cirrus clouds. Figure 7 compares some segmentation results of different methods. The interactive segmentation methods (Fig. 7b, c) are often tedious and

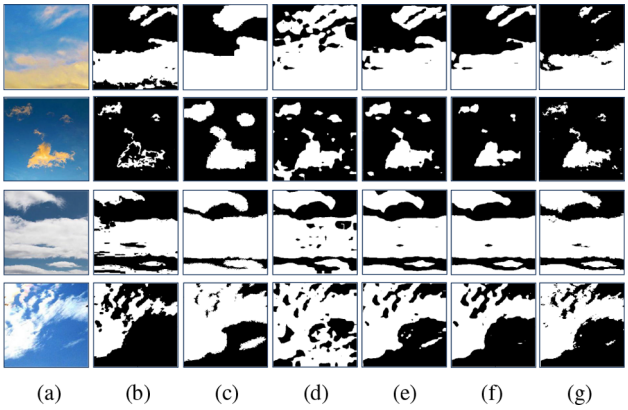


Fig. 7 Segmentation results. (**a**) The original images. (**b**) The chromatic segmentation [64]. (**c**) The paint selection segmentation [65]. (**d**) The segmentation trained on the original synthetic images. (**e**) The segmentation trained on the synthetic images after color space adaptation. (**f**) The segmentation trained on the manual labeled real images. (**g**) The ground truth segmentation labels

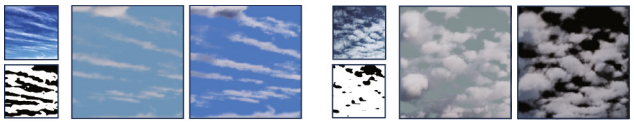


Fig. 8 Further applications. The single image-based 3D volume data reconstruction of cirrus clouds

time-consuming to yield a reasonable annotation result. The segmentation results using the original synthetic images for training have obvious artifacts (Fig. 7d). On the other hand, using the adapted synthetic images for training significantly improves the segmentation accuracy (Fig. 7e), whose results are visually quite similar with that of using manually labeled images for training (Fig. 7f) and the ground truth (Fig. 7g). Finally, we can make some further applications based on the segmentation results.

3D reconstruction By leveraging the trained segmentation network, we generate the segmentation label as input to reconstruct the 3D volume data from a cirrus cloud image [64]. We re-render it under new lighting conditions to produce new images (Fig. 8).

Matting and composition We execute image matting [66] based on a tri-map computed from the inflated segmentation label, to first separate the cirrus cloud foreground from the source image (Fig. 9 left) and then paste it onto other target images (Fig. 9 middle) to recompose novel cirrus clouds images (Fig. 9 right).

Style transfer and photograph post-processing Our method has the potential for tasks like style transfer and photograph post-processing. We execute style transfer task on the MIT-Adobe [57] dataset and the Pexels dataset and compare our method with CycleGAN. We put experiment details in "Appendix C". Figure 10 shows the transferred styles. Our method can output results with comparable quality to CycleGAN on these datasets. While CycleGAN generates images directly and owns a broader space to change styles, our method predicts parameters for human-understandable operations. Figure 2 demonstrates a sequence of color adjustment operations for transferring between two artists' styles. Moreover, CycleGAN leads to blurry images and edge distortion (see Fig. 11), while our method is inherently resolution-independent due to the content preserving operations. Therefore, we can train our model on low-res images and apply it to arbitrary high-res ones, making the method more suitable for photograph processing and graphics applications.

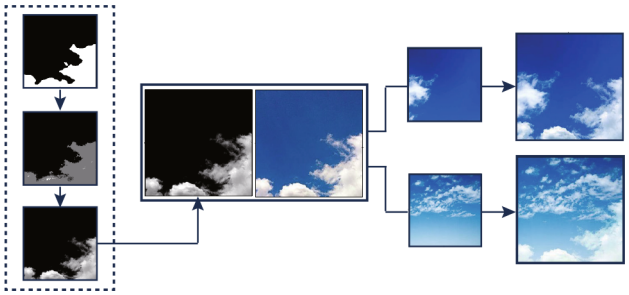


Fig. 9 Further applications. The cirrus clouds image matting and composition

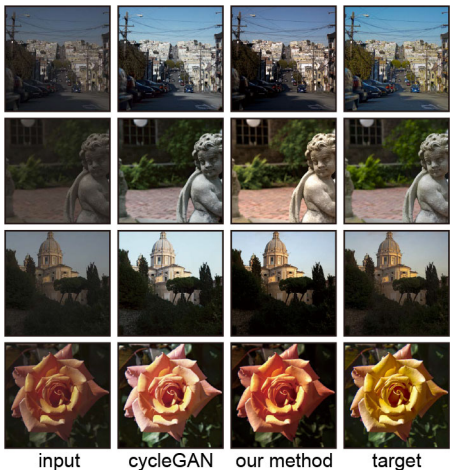


Fig. 10 Comparison with CycleGAN. Results generated on the MIT-Adobe FiveK dataset

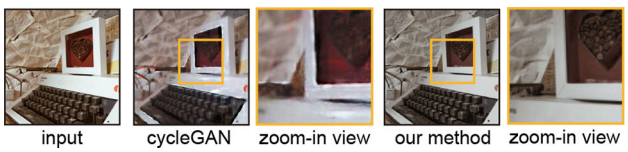


Fig. 11 Comparison with CycleGAN. Our method can process arbitrary high-resolution images on which CycleGAN leads to blurry results

6 Conclusions

We have introduced a color space adaptation method for bridging the gap between synthetic and real cirrus cloud images. The adaptation method is a two-stage learning approach. A sequence of label-preserving operations is adopted in both two stages to make variants in the color space. We demonstrate that training on the adapted synthetic cirrus cloud images has significantly improved the semantic segmentation performance compared with training on the original ones.

Our method is easy to implement and deploy. It can play a primary preprocessing role in the adaptation toolset and be complementary to those convolution-based approaches. The framework is extension-friendly: other advanced color adjustment operations can be readily introduced, e.g., the spatially variant adjustment based on curves. These operations are human-understandable and easy to control, which could improve the variability of the adversarial data for training and extend the adaptation scope of the generator. This enhanced adaptation ability is essential to adapt a more complex dataset like Virtual KITTI [11]. With targeted designs of label-preserving operations, our method of color space domain adaptation owns a robust performance on different tasks. Moreover, the performance improves along with the rise of the details produced by photo-realistic rendering. It is promising to apply our method to general synthetic datasets in color space and acquire proper performance improvement with the development of photo-realistic rendering technologies.

Funding This study was funded by National Natural Science Foundation of China (Grant Nos. 61772024, 61732016).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix

A: Color adjustment operations

Brightness The brightness adjustment operation is defined as

$$\mathsf{op}_b(x;\alpha_b) = \begin{cases} x\cdot(1-\alpha_b)+\alpha_b, & \text{if } \alpha_b >= 0 \\ x + x\cdot\alpha_b, & \text{otherwise} \end{cases} \tag{7}$$

where x is the input image, and α_b is a scalar parameter that controls the extent of the adjustment. We clip α_b into the range $[-1, 1]$.

Saturation The saturation adjustment operation is defined as

$$\mathsf{op}_s(x;\alpha_s) = \begin{cases} x + (x-\mathsf{L}(x))\cdot\mathsf{s}(x,\alpha_s), & \text{if } s > 0 \\ \mathsf{L}(x) + (x-\mathsf{L}(x))\cdot(1+\mathsf{s}(x,\alpha_s)), & \text{otherwise} \end{cases} \tag{8}$$

where x is the input image and α_s is a scalar parameter that controls the extent of the adjustment. We clip α_s into the range $[-1, 1]$.

$\mathsf{L}(x)$ is the per-pixel average of the three channels $\frac{1}{2}\cdot[\mathsf{rgb\_max}(x)+\mathsf{rgb\_min}(x)]$, and $\mathsf{s}(x,\alpha_s)$ is defined as

$$\mathsf{s}(x;\alpha_s) = \begin{cases} 1/\mathsf{S}(x) - 1, & \text{if } \alpha_s + \mathsf{S}(x) >= 1 \\ 1/(1-\alpha_s) - 1, & \text{otherwise} \end{cases}$$

$\mathsf{S}(x)$ is defined as a per-pixel ratio

$$\mathsf{S}(x) = \begin{cases} \mathsf{delta}(x)/(2\cdot\mathsf{L}(x)), & \text{if } \mathsf{L} < 0.5 \\ \mathsf{delta}(x)/(2-2\cdot\mathsf{L}(x)), & \text{otherwise} \end{cases}$$

where $\mathsf{delta}(x) = \mathsf{rgb\_max}(x) - \mathsf{rgb\_min}(x)$.

Contrast The contrast adjustment operation is defined as

$$\mathsf{op}_c(x;\alpha_c) = \begin{cases} \bar{x} + (x-\bar{x})/(1-\alpha_c), & \text{if } \alpha_c >= 0 \\ \bar{x} + (x-\bar{x})\cdot(1+\alpha_c), & \text{otherwise} \end{cases} \tag{9}$$

where x is the input image, $\bar{x}$ is the average of all pixel values of x, and α_c is a scalar parameter that controls the extent of the adjustment. We clip α_c into the range $[-1, 1]$ (Fig. 12).

B: Handcrafted approach for image augmentation

We execute an adaptation approach using handcrafted features in the experiment. First, we transfer the images to HSV space to extract features of saturation and brightness. Then, we fit a Gaussian distribution model to the feature points of the real images. Next, for each synthetic image, we shift its features toward a target point sampled from the Gaussian distribution model. Finally, we reconstruct augmented images from the shifted features. Compared with the handcrafted feature, our generator learns more powerful features in higher dimensions and leverages that to decide the best way to shift each synthetic image (Fig. 13).

C: Details of style transfer and image post-processing

All the training images are zero-centered and rescaled to $[-1, 1]$. We set the batch size to 8. We adopt the Adam optimizer with $lr = 2\text{e}{-4}$ and $\beta_1 = 0.5$ and train both

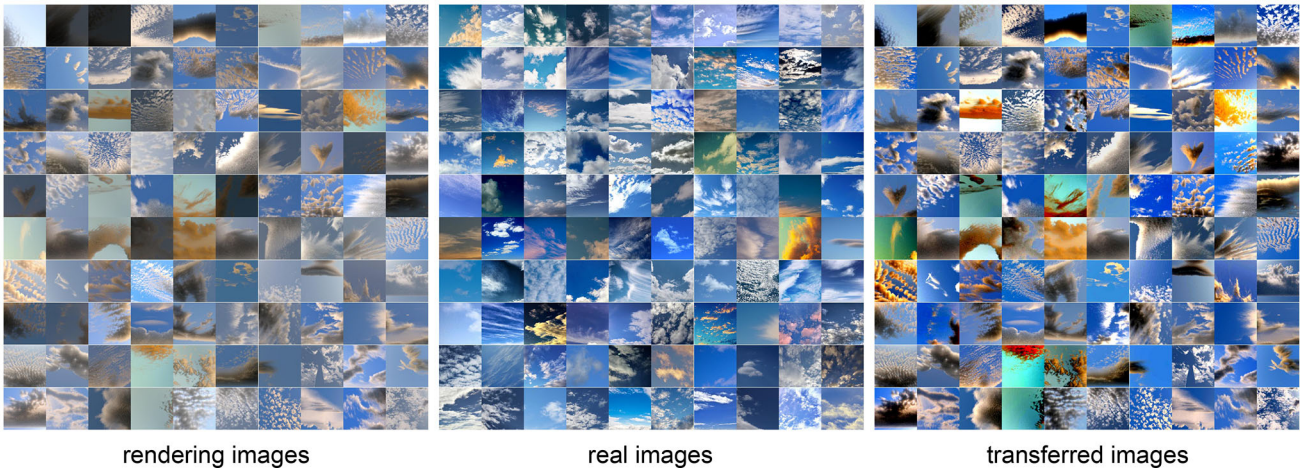


Fig. 12 More color adaptation results

Fig. 13 Styles overview. Retouched images of artist A (left) and artist B (right) from www.pexels.com

Fig. 14 Adaptation on Pexels. Our method adapts the photos of artist A (top row) to the style of artist B (bottom row)

Fig. 15 Cross-dataset generalization. The model trained on the Pexels dataset is applied to raw images of the MIT-Adobe FiveK Dataset (top row) to obtain an artist style (bottom row)

the discriminator and the generator for 100 epochs. We show more color adaptation results on the Pexels dataset in Fig. 14. We also apply the model trained on the Pexels dataset to images in the MIT-FiveK dataset to show the ability of cross-dataset generalization (Fig. 15). While similar effects can be produced by [47], our method does not require reinforcement learning.

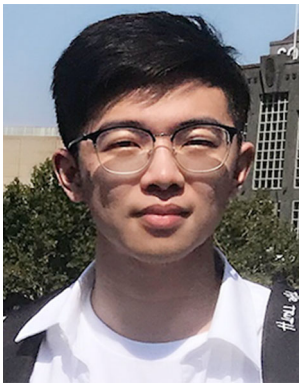
References

1. Yuan, F., Lee, Y.H., Meng, Y.S.: Comparison of cloud models for propagation studies in ka-band satellite applications. In: Proceedings of IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (AP-S/URSI), pp. 383–384 (2014)
2. Christodoulou, C., Michaelides, S., Pattichis, C.: Multifeature texture analysis for the classification of clouds in satellite imagery. IEEE Trans. Geosci. Remote Sens. **41**(11), 2662–2668 (2003)
3. Yuan, F., Lee, Y.H., Meng, Y.S.: Comparison of radio-sounding profiles for cloud attenuation analysis in the tropical region. In: Proceedings of IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (AP-S/URSI), pp. 259–260 (2014)
4. Mahrooghy, M., Younan, N.H., Anantharaj, V.G., Aanstoos, J., Yarahmadian, S.: On the use of a cluster ensemble cloud classification technique in satellite precipitation estimation. IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens. **5**(5), 1356–1363 (2012)
5. Dev, S., Lee, Y.H., Winkler, S.: Color-based segmentation of sky/cloud images from ground-based cameras. IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens. **10**(1), 231–242 (2017)
6. Dianne, G., Wiliem, A., Lovell, B.C.: Deep-learning from mistakes: automating cloud class refinement for sky image segmentation. In:

Proceedings of Digital Image Computing: Techniques and Applications (DICTA), pp. 1–8 (2019)
7. Dev, S., Manandhar, S., Lee, Y.H., Winkler, S.: Multi-label cloud segmentation using a deep network. In: Proceedings of IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (AP-S/URSI) (2019)
8. Dev, S., Lee, Y.H., Winkler, S.: Multi-level semantic labeling of sky/cloud images. In: Proceedings of IEEE International Conference on Image Processing (ICIP), pp. 636–640 (2015)
9. Sun, B., Saenko, K.: From virtual to reality: fast adaptation of virtual object detectors to real domains. In: Proceedings of British Machine Vision Conference (BMVC) (2014)
10. Massa, F., Russell, B.C., Aubry, M.: Deep exemplar 2d–3d detection by adapting from real to rendered views. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6024–6033 (2016)
11. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtualworlds as proxy for multi-object tracking analysis. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4340–4349 (2016)
12. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: Proceedings of IEEE/RJS International Conference on Intelligent Robots and Systems (IROS), pp. 23–30 (2017)
13. Liebelt, J., Schmid, C.: Multi-view object class detection with a 3d geometric model. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1688–1695 (2010)
14. Stark, M., Goesele, M., Schiele, B.: Back to the future: learning shape models from 3d cad data. In: Proceedings of British Machine Vision Conference (BMVC), pp. 1–11 (2010)
15. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: viewpoint estimation in images using CNNS trained with rendered 3D model views. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 2686–2694 (2015)
16. Grabner, A., Roth, P.M., Lepetit, V.: 3D pose estimation and 3D model retrieval for objects in the wild. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3022–3031 (2018)
17. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: ground truth from computer games. In: Proceedings of European Conference on Computer Vision (ECCV), pp. 102–118 (2016)
18. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3234–3243 (2016)
19. Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., Krishnan, D.: Unsupervised pixel-level domain adaptation with generative adversarial networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 95–104 (2017)
20. Hoffman, J., Tzeng, E., Park, T., Zhu, J.Y., Isola, P., Saenko, K., Efros, A., Darrell, T.: Cycada: cycle-consistent adversarial domain adaptation. In: Proceedings of International Conference on Machine Learning (ICML), pp. 1989–1998 (2018)
21. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proceedings of Neural Information Processing Systems (NeurIPS), pp. 2672–2680 (2014)
22. Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W.: Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 105–114 (2017)
23. Snderby, C.K., Caballero, J., Theis, L., Shi, W., Husz'r, F.: Amortised map inference for image super-resolution. In: Proceedings of International Conference on Learning Representations (ICLR) (2017)
24. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: Proceedings of International Conference on Learning Representations (ICLR) (2016)
25. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5967–5976 (2017)
26. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 2242–2251 (2017)
27. Taigman, Y., Polyak, A., Wolf, L.: Unsupervised cross-domain image generation. In: Proceedings of International Conference on Learning Representations (ICLR) (2017)
28. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. In: Proceedings of Neural Information Processing Systems (NeurIPS), pp. 469–477 (2016)
29. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2337–2346 (2019)
30. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. In: Proceedings of International Conference on Machine Learning (ICML), vol. 48, pp. 1060–1069 (2016)
31. Zhang, H., Xu, T., Li, H.: Stackgan: text to photo-realistic image synthesis with stacked generative adversarial networks. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 5908–5916 (2017)
32. Hong, S., Yang, D., Choi, J., Lee, H.: Inferring semantic layout for hierarchical text-to-image synthesis. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7986–7994 (2018)
33. Long, M., Cao, Y., Wang, J., Jordan, M.: Learning transferable features with deep adaptation networks. In: Proceedings of International Conference on Machine Learning (ICML), pp. 97–105 (2015)
34. Sixt, L., Wild, B., Landgraf, T.: Rendergan: generating realistic labeled data. In: Proceedings of International Conference on Learning Representations (ICLR) (2017)
35. Wang, B., Yu, Y., Xu, Y.Q.: Example-based image color and tone style enhancement. ACM Trans. Graph. (TOG) **30**(4), 64 (2011)
36. Kuanar, S., Rao, K.R., Mahapatra, D., Bilas, M.: Night time haze and glow removal using deep dilated convolutional network. arXiv preprint arXiv:1902.00855 (2019)
37. Kuanar, S., Conly, C., Rao, K.R.: Deep learning based HEVC in-loop filtering for decoder quality enhancement. In: Proceedings of Picture Coding Symposium (PCS), pp. 164–168 (2018)
38. Kuanar, S., Athitsos, V., Mahapatra, D., Rao, K., Akhtar, Z., Dasgupta, D.: Low dose abdominal ct image reconstruction: an unsupervised learning based approach. In: Proceedings of IEEE International Conference on Image Processing (ICIP), pp. 1351–1355 (2019)
39. Reinhard, E., Adhikhmin, M., Gooch, B., Shirley, P.: Color transfer between images. IEEE Comput. Graph. Appl. **21**(5), 34–41 (2001)
40. Huang, H., Zang, Y., Li, C.F.: Example-based painting guided by color features. Vis. Comput. **26**(6), 933–942 (2010)
41. Huang, H., Xiao, X.: Example-based contrast enhancement by gradient mapping. Vis. Comput. **26**(6), 731–738 (2010)

42. Yan, Z., Zhang, H., Wang, B., Paris, S., Yu, Y.: Automatic photo adjustment using deep neural networks. ACM Trans. Graph. (TOG) **35**(2), 11 (2016)
43. Gharbi, M., Chen, J., Barron, J.T., Hasinoff, S.W., Durand, F.: Deep bilateral learning for real-time image enhancement. ACM Trans. Graph. (TOG) **36**(4), 118 (2017)
44. Chen, Y.S., Wang, Y.C., Kao, M.H., Chuang, Y.Y.: Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6306–6314 (2018)
45. Limmer, M., Lensch, H.P.A.: Infrared colorization using deep convolutional neural networks. In: Proceedings of International Conference on Machine Learning and Applications (ICMLA), pp. 61–68 (2016)
46. Park, J., Lee, J.Y., Yoo, D., Kweon, I.S.: Distort-and-recover: color enhancement using deep reinforcement learning. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5928–5936 (2018)
47. Hu, Y., He, H., Xu, C., Wang, B., Lin, S.: Exposure: a white-box photo post-processing framework. ACM Trans. Graph. (TOG) **37**(2), 26 (2018)
48. Bianco, S., Cusano, C., Piccoli, F., Schettini, R.: Learning parametric functions for color image enhancement. In: Proceedings of International Workshop on Computational Color Imaging (CCIW), vol. 11418, pp. 209–220 (2019)
49. Chai, Y., Giryes, R., Wolf, L.: Supervised and unsupervised learning of parameterized color enhancement. In: Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 992–1000 (2020)
50. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440 (2015)
51. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFS. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **40**(4), 834–848 (2018)
52. Bi, L., Feng, D.D., Kim, J.: Dual-path adversarial learning for fully convolutional network (FCN)-based medical image segmentation. Vis. Comput. **34**(6), 1043–1052 (2018)
53. Bi, L., Kim, J., Kumar, A., Fulham, M., Feng, D.: Stacked fully convolutional networks with multi-channel learning: application to medical image segmentation. Vis. Comput. **33**(6), 1061–1071 (2017)
54. Wang, J., Zheng, C., Chen, W., Wu, X.: Learning aggregated features and optimizing model for semantic labeling. Vis. Comput. **33**(12), 1587–1600 (2017)
55. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask r-CNN. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **42**(2), 386–397 (2020)
56. Jakob, W.: Mitsuba renderer (2010). http://www.mitsuba-renderer.org
57. Bychkovsky, V., Paris, S., Chan, E., Durand, F.: Learning photographic global tonal adjustment with a database of input / output image pairs. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 97–104 (2011)
58. Chollet, F., et al.: Keras. https://keras.io (2015)
59. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.: Tensorflow: a system for large-scale machine learning. In: Proceedings of USENIX Symposium on Operating Systems Design and Implementation (OSDI), pp. 265–283 (2016)
60. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 1026–1034 (2015)
61. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. In: Proceedings of International Conference on Learning Representations (ICLR) (2015)
62. Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., Rohde, G.: Generalized sliced Wasserstein distances. In: Proceedings of Neural Information Processing Systems (NeurIPS), pp. 261–272 (2019)
63. Bonneel, N., Rabin, J., Peyr, G., Pfister, H.: Sliced and radon Wasserstein barycenters of measures. J. Math. Imaging Vis. **51**(1), 22–45 (2015)
64. Dobashi, Y., Shinzo, Y., Yamamoto, T.: Modeling of clouds from a single photograph. Comput. Graph. Forum (CGF) **29**(7), 2083–2090 (2010)
65. Liu, J., Sun, J., Shum, H.Y.: Paint selection. p. 69 (2009)
66. Chen, Q., Li, D., Tang, C.K.: KNN matting. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **35**(9), 2175–2188 (2013)

Qing Lyu received the bachelor's degree in Digital Media Technology from Zhejiang University in 2016. Currently, she is working toward the Ph.D. degree at the State Key Lab of CAD&CG, Zhejiang University. Her research interests include visual computing and computer graphics.



Minghao Chen received the B.S. degree from the Zhejiang University, Hangzhou, Zhejiang, China, in 2018, where he is currently pursuing the Ph.D. degree. His main research interests include computer vision and domain adaptation.

Xiang Chen is an Associate Professor in the State Key Lab of CAD&CG, Zhejiang University. He received his Ph.D. in Computer Science from Zhejiang University in 2012. His current research interests mainly include fabrication-aware design, physics-based simulation, image analysis, shape modeling/retrieval and computer-aided design.