Computational Visual Media
https://doi.org/10.1007/s41095-020-0201-9 Vol. 7, No. 1, March 2021, 127–138
Research Article

A GAN-based temporally stable shading model for fast animation of photorealistic hair

Zhi Qiao[1] (✉), **Takashi Kanai**[1]

Abstract We introduce an unsupervised GAN-based model for shading photorealistic hair animations. Our model is much faster than previous rendering algorithms and produces fewer artifacts than other neural image translation methods. The main idea is to extend the Cycle-GAN structure to avoid semi-transparent hair appearance and to exactly reproduce the interaction of the lights with the scene. We use two constraints to ensure temporal coherence and highlight stability. Our approach outperforms and is computationally more efficient than previous methods.

Keywords hair animation; fast shading; neural networks

1 Introduction

Photorealistic hair rendering is important if virtual characters are to provide a high degree of realism. Current real-time hair rendering approaches generate unrealistic output, which looks artificial.

Human hair has many strands, forming an extremely complicated geometric structure. Each long fiber has a complicated shape, and it is difficult to represent each and every hair detail accurately using any modeling scheme. The complexity of the structure makes hair impossible to realistically render as a large surface.

Our goal is to achieve photorealistic hair animation. The luster and color of the hair depend on multiple scattering and reflection of light in the hair fibers. Our shader is based on the d'Eon model [1] for specular reflections and the Zinke model [2] for diffuse reflections. The shader approximates reflection from the surface of the hair with anisotropic specularity, as well as its refraction through fibers and its scattering by multiple strands. Figure 1(b) shows a photorealistic hair rendering, which requires the handling of many light and hair interactions in a similar manner to how they occur in the real world. To render a high-quality hairstyle, as hair is translucent, we must follow light passing through it, and bouncing around inside it, before exiting. The hair shader approximates many light paths and computes a great number of scatterings and reflections in the hair fibers. Such a process is very costly. On the other hand, due to the large number of hair strands, it is undesirable to explicitly model them individually. Therefore, in a real-time scene, a hair strand is often represented by a curve with some thickness. It is common to use very simplified rendering models that do not capture the full complexity of lighting in hair fibers. The shortcomings in the results can be especially

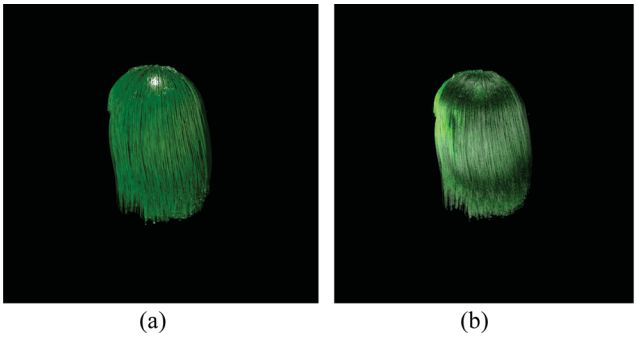


Fig. 1 Hair rendering. (a) Image rendered by a fast model. The hair material only contains roughness, highlight size, and whiteness properties. (b) Image rendered by a state-of-art model based on d'Eon's and Zinke's work. Even for the same scene, the appearance is totally different. This means that our method cannot apply supervised learning.

1 Department of General Systems Studies, Graduate School of Arts and Sciences, The University of Tokyo, Tokyo, Japan. E-mail: Z. Qiao, zhiqiao.p.r.c@gmail.com (✉); T. Kanai, kanait@acm.org.

Manuscript received: 2020-09-20; accepted: 2020-11-27

noticeable when shading lightly colored hair. In this work, we focus on energy-conserving but realistic shading of hair, providing similar results to a high-quality offline hair rendering model.

High-quality hair rendering has several challenging problems that are difficult to solve, and act as barriers to achieve realistic appearance. The first issue is how light interacts with a single fiber. This step alone involves much computation, and we must further consider how light from the first step interacts with other hair fibers. To solve these complex problems, many models have been proposed. Regrettably, no methods provide high-performance and high-quality at the same time.

To find a way to resolve the contradiction between performance and quality, we introduce a novel method for photorealistic hair animation, where high-performance hair data and high-quality hair data are both provided as references. Current machine learning research has made great progress in image translation, which can translate images from one domain to another. Our key idea is to train a generator, to produce photorealistic hair shading images from low-quality ones. Applying current research on image translation, our method uses an unsupervised model to achieve photorealistic hair shading.

The next challenge is to ensure the results are temporally coherent, as our application scenario requires a sequence of frames. This is a gap that we must bridge.

Our work is based on conditional generative adversarial networks (cGAN), and is also inspired by recent style transfer research. We propose an unsupervised learning method, which builds on the Cycle-GAN [3] architecture. In our application, directly applying Cycle-GAN to hair shading transfer leads to a temporal instability problem that results in abnormal highlight appearance. To resolve this problem, we added a temporal coherence module and a highlight correction module.

The principal contributions of this paper are

1. An energy-conserving model that converts low-quality shaded hair images to photorealistic appearance.
2. A novel method that utilizes motion vectors to ensure smooth output image sequences.
3. A new constraint to ensure highlight stability and faithful illumination appearance.

This work is the first energy-conserving photorealistic hair shading model based on neural style transfer. Figure 5 shows some of our outputs.

2 Related work

2.1 Hair rendering

As we focus on producing photorealistic hair images, it is necessary to render many high-quality and high-performance hair images for use as training datasets. In particular, the appearance of our outputs significantly depends on the quality of the inputs. So in this section, we briefly introduce previous research both for real-time hair rendering and offline hair rendering, whose results are used in our training datasets.

A typical early approach to hair rendering was proposed by Kajiya and Kay [4] and Yan et al. [5]. This model considered light scattering by a single hair fiber, using a diffuse term and a specular term. Because of the simplicity of this model, it is widely used in real-time applications such as games or interactive movies. However, this method resulted in flat hair apperance due to the inadequate prediction of the azimuthal dependence of the scattering intensity and its diffusion term. Later, Marschner et al. [6, 7] proposed a model that treats each hair fiber as a translucent cylinder. Light interacting with the hair belongs to three types of paths: R, TT, and TRT, where R indicates a reflected ray, and T indicates a transmitted or refracted ray. The state-of-art rendering model was proposed by Moon et al. [8]. It applied spherical harmonic approximation to model multiple scattered incident radiance. All of these methods have huge computational requirements, and even now cannot be applied in real time.

On the other hand, several simplified hair rendering models have been proposed for real-time use. Zinke's algorithm [2] was designed to handle simple light sources and did not directly consider light integration and transport complexities with environment lighting. Ren et al. [9] proposed an algorithm for real-time hair rendering with both single and multiple scattering effects with complex environment lighting. This method approximates the environment light by a set of spherical radial basis functions. Jansson et al. [10] presented an approximation of strand-based hair for hybrid hair rendering. All of these methods do not

sufficiently consider important phenomena such as directionality of multiple scattering, inter-reflections, blurring, and color-shifting effects.

However, the main shortcoming of most of these methods is that they utilize several simplifications and non-physical parameters, which cannot be derived from physical hair fiber properties, reducing accuracy. Although these methods can produce plausible hair appearance in specific situations, the parameters need to be specially set according to the particular scene and illumination.

2.2 Style transfer

A variety of works have addressed style transfer. Gatys et al. [11] firstly proposed a method that combines the content of one image with the style of another by matching the Gram matrix statistics of deep features using optimization. Johnson et al. [12] proposed a feed-forward network to approximately transfer a single style to multiple desired images. Based on previous research, Luan et al. [13] introduced semantic segmentation guidance and a photorealism constraint to avoid edge distortion. Others have proposed neural-based hair rendering pipelines. Wei et al. [14] presented an adversarial network for rendering photorealistic hair as an alternative to conventional computer graphics pipelines. Chai et al. [15] introduced a generic neural-based hair rendering pipeline that can synthesize photo-realistic images from virtual 3D hair models.

Inspired by recent advances in GANs [16], various advanced style transfer methods [11, 17, 18] have been proposed based on image-to-image translation. Zhu et al. [19, 20] pioneered a general solution that leverages cGANs for image-to-image translation. It has been widely applied, e.g., for semantic labeling of photos and producing photos from line drawings. To extend this method to unsupervised learning, Cycle-GAN [21] was introduced to ensure bijective consistency between the two domains, thereby providing photorealistic, diverse results.

To use Cycle-GAN for video-to-video translation, Bansal et al. [22] proposed Recycle-GAN, which applies a predictor to synthesize the next frame as shown in Fig. 2(b). This method achieves temporal coherence for video style transfer. Chen et al. [23]

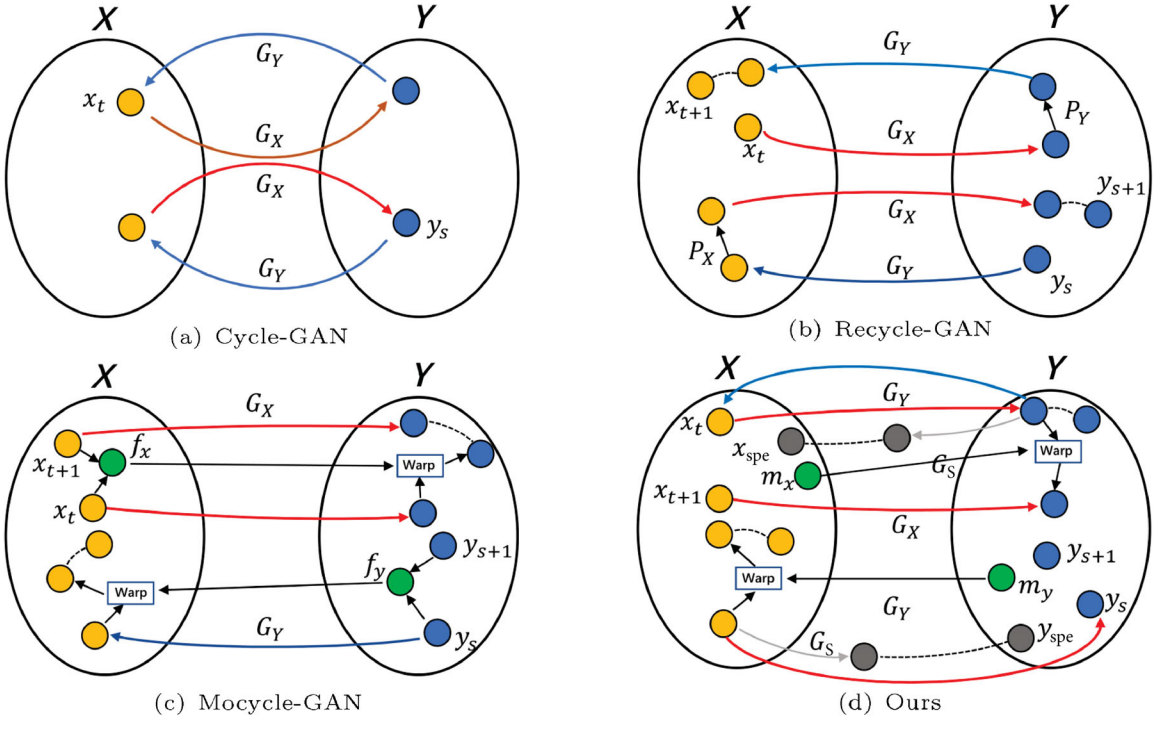


Fig. 2 Mapping approaches. (a) Cycle-GAN uses two generators (G_X and G_Y) to map between domains X and Y using adversarial loss and cycle consistency loss. The red arrow means image x_t from domain X is translated to domain Y; the output should be similar to image y_s of domain Y. In the same way, y_s is also translated to domain X via the blue arrow shown. (b) Recycle-GAN introduces predictors (P_X and P_Y) to synthesize future frames to ensure temporal coherence. (c) Mocycle-GAN utilizes generated optical flow (f_x and f_y) to predict the next frame by warping the current frame. (d) Instead of optical flow, we use the more precise motion vector (m_x and m_y) to predict the next frame, and a highlight constraint to retain faithful illumination appearance by use of the pretrained specular generator G_S.

also devised an unpaired video-to-video translation approach based on Aayush's paper: see Fig. 2(c). It uses FlowNet to generate optical flow instead of the recurrent temporal predictor in Ref. [22].

In this paper, we extend Cycle-GAN to photo-realistic hair transfer, which is faster than previous offline hair rendering. Instead of the predictor of Recycle-GAN and FlowNet of Mocycle-GAN [23], we directly apply the motion vectors generated from the rendering pipeline to predict the next frame. In addition, our model includes a highlight correction module. Our work uses reference images from the target domain to synthesize the specified desired hair appearances.

3 Formulation

3.1 Introduction

Figures 1(a) and 1(b) show images rendered using a fast shading model and a high-quality reflectance model, with the same lighting environment. The highlight appearances are very different in the two images.

Suppose we use Pix2pix-GAN to train our model and Fig. 1(b) is taken as the ground truth. The neural network will tend to learn a mapping that translates the highlight appearance from Fig. 1(a) to Fig. 1(b). However, the correct mapping in the training stage does not mean that this map is also suitable in the inference stage. Even though the highlight appearances are correct for the case with ground truth, the trained mapping may be unstable and lead to unpredictable inference results. In different scenes, the highlight positions and appearances are too complex and irregular to build a correct mapping between the hair image rendered by a fast model and a high-quality model: if the lighting or hair or viewpoint is slightly changed, the trained model could fail to produce the correct highlight position and appearance. Thus, in this application, the hair rendered by a high-quality model as in Fig. 1(b) cannot be taken as the ground truth. Thus, instead, we use Cycle-GAN as an unsupervised model.

We also apply a small number of stylistic references, which come from the target domain. We later show how the references affect the results given the same input. Our model needs to know where the appropriate highlight area is; otherwise, it can generate wrong highlight appearances during application. For this purpose, we formulate a specular generator to extract a specular map (highlight map) from RGB images. Using this generator, the highlight area can be constrained to an appropriate position close to the input highlighted area. Furthermore, temporal coherence is required for video style transfer. In contrast to conventional complicated methods, our approach directly uses the motion vector extracted from the rendering pipeline to predict the next frame. See Fig. 2.

3.2 Preliminaries

Our work is based on Cycle-GAN [16]. Cycle-GAN considers image translation from domain X to domain Y. The aim of this model is thus to learn a pair of generators. See Fig. 2(a). The generator G_X maps an image $x \in X$ to an image $y \in Y$. The generator G_Y maps an image $y \in Y$ to an image $x \in X$. The discriminators D_X and D_Y are used to discriminate real images from synthetic ones, combining adversarial loss and cycle consistency loss on the image.

3.2.1 Adversarial loss

Adversarial loss is used in GANs; it is also applied in Cycle-GANs. Its basic form is

$$L_{\mathrm{GAN}}(G_X, D_Y, X, Y) = \mathbb{E}_{y\sim P_Y}[\log D_Y(y)] + \mathbb{E}_{x\sim P_X, y\sim P_Y}[\log(1 - D_Y(G_X(x))] \quad (1)$$

The discriminator D_Y distinguishes real images from fake images by learning to maximize $L_{\mathrm{GAN}}(G_X, D_Y, X, Y)$. Meanwhile, another generator learns to minimize $L_{\mathrm{GAN}}(G_Y, D_X, X, Y)$:

$$L_{\mathrm{GAN}}(G_Y, D_X, X, Y) = \mathbb{E}_{x\sim P_X}[\log D_X(x)] + \mathbb{E}_{x\sim P_X, y\sim P_Y}[\log(1 - D_X(G_Y(y))] \quad (2)$$

Similarly, D_X distinguishes the real images from fake images in domain X. The generators G_X and G_Y aim to generate images that cannot be distinguished by the adversaries D_Y and D_X.

3.2.2 Cycle consistency loss

By minimizing the adversarial losses, G_X and G_Y learn to generate images that are photorealistic and correctly classified in the corresponding domain. However, adversarial loss can not ensure synthetic images satisfying the requirement that the learned function maps each individual input x to a desired output y. To solve this problem, Cycle-GAN applies

a consistency loss to the generators, defined as

$$L_{\text{cyc}}(G_X,G_Y) = \mathbb{E}_{x\sim P_X}[||G_Y(G_X(x)) - x||_1] + \mathbb{E}_{y\sim P_Y}[||G_X(G_Y(y)) - y||_1] \quad (3)$$

3.3 Our method

As described above, Cycle-GAN uses two generators to map between domains X and Y by considering adversarial loss. Cycle consistency loss further constrains the mapping to be a one-to-one mapping, which forces different samples in the source domain to translate correctly to the target domain: Cycle-GAN provides a bijective mapping between the two domains.

Unlike Cycle-GAN, our work builds a feedforward hair shading transformation network that can rapidly transfer style from an arbitrary high-quality reference image to an arbitrary target-style hair image.

We first provide a reference for the adversarial and cycle consistency losses. The reference may be observed in the training or inference stage, and consists of only one image obtained from the corresponding domain X or Y for one kind of hairstyle. These losses constrain the results of G_X and G_Y according to the reference r_x and r_y obtained from the corresponding domain. The adversarial losses for G_X and G_Y are

$$L_{\text{GAN}}(G_X, D_Y, X, Y) = \mathbb{E}_{y\sim P_Y}[\log D_Y(y)] + \mathbb{E}_{x\sim P_X, r_y\sim P_Y}[\log(1 - D_Y(G_X(x, r_y)))] \quad (4)$$

$$L_{\text{GAN}}(G_Y, D_X, X, Y) = \mathbb{E}_{x\sim P_X}[\log D_X(x)] + \mathbb{E}_{y\sim P_Y, r_x\sim P_X}[\log(1 - D_X(G_Y(y, r_x)))] \quad (5)$$

The references are also provided for the cycle consistency loss:

$$L_{\text{cyc}}(G_X, G_Y) = \mathbb{E}_{x\sim P_X}[||G_Y(G_X(x, r_y), r_x) - x||_1] + \mathbb{E}_{y\sim P_Y}[||G_X(G_Y(y, r_x), r_y) - y||_1] \quad (6)$$

A fundamental weakness of the Cycle-GAN model is that it learns a mapping only at the frame level. 3D animation requires a contiguous frame sequence, so the generator must learn to further achieve a temporally coherent mapping to ensure smooth visual appearance across contiguous frames. Recycle-GAN [22] and Mocycle-GAN [23] both considered this issue, and predict future frames by training a temporal predictor or an optical flow predictor. However, Recycle-GAN does not apply motion information from adjacent frames to drive video-to-video translation. Mocycle-GAN explicitly models motion across frames with optical flow, but the optical flow is predicted by another trained net, Flownet. This causes deviations in the optical flow and distortions occur during the warping step.

Instead, our method also utilizes motion information, but it is rendered by the graphics pipeline instead of coming from an additional neural network. It is provided by a motion vector. In the graphics pipeline, the renderer encodes a 2D vector representing object motion along the x-axis as green and y-axis as red. Since the motion vector is directly generated from the pipeline, it can provide very accurate optical flow information. Using this motion vector, each synthetic frame can be warped to the subsequent frame.

Consider two contiguous frames from domain X, x_t and x_{t+1}. As explained above, x_t and x_{t+1} are reconstructed as $G_Y(G_X(x_t, r_y), r_x)$ and $G_Y(G_X(x_{t+1}, r_y), r_x)$. Also, frame x_t is warped as $W(G_Y(G_X(x_t, r_y), r_x), m_x)$, which should be similar to the next reconstructed frame $G_Y(G_X(x_{t+1}, r_y), r_x)$. Correspondingly, for two contiguous frames y_t and y_{t+1} in domain Y, the warped frame $W(G_X(G_Y(y_t, r_x), r_y), m_y)$ should be similar to the next reconstructed frame $G_X(G_Y(y_{t+1}, r_x), r_y)$. We apply a temporal cycle consistency constraint is applied to the L_1 distance between the warped frame and the synthetic next frame:

$$\begin{aligned} L_{\text{tempo}}(G_X, G_Y) = \\ &\mathbb{E}_{x\sim P_X}[||W(G_Y(G_X(x_t, r_y), r_x), m_x) \\ &- G_Y(G_X(x_{t+1}, r_y), r_x)||_1] \\ &+ \mathbb{E}_{y\sim P_Y}[||W(G_X(G_Y(y_t, r_x), r_y), m_y) \\ &- G_X(G_Y(y_{t+1}, r_x), r_y)||_1] \end{aligned} \quad (7)$$

Another problem is highlight distortion, which is addressed by a specular constraint in our model. We introduce a specular map that is used to define a surface's highlight color, as shown in Fig. 4. This specular map is rendered by the graphics pipeline together with RGB images and motion vectors. Before training the primary network, we pre-train a specular generator G_S that synthesizes a specular map from an RGB image. For this pre-trained network, we use a standard Pix2pix-GAN model. The dataset consists of RGB images as the input, and the specular map as the ground truth. The architecture of the model will be detailed in the next section. During

training, G_S extracts the specular map from the synthetic RGB image: $G_X(x, r_y)$, $G_Y(y, r_x)$. The L_1 loss is used to minimize the error, which is the sum of absolute differences between the extracted specular map of the fake RGB image, $G_S(G_X(x, r_y))$, $G_S(G_Y(y, r_x))$, and the generated specular map of the real input, $G_S(x)$ and $G_S(y)$:

$$L_{spe}(G_X, G_Y) = \mathbb{E}_{x\sim P_X}[||G_S(G_X(x, r_y)) - G_S(x)||_1] + \mathbb{E}_{y\sim P_Y}[||G_S(G_Y(y, r_x)) - G_S(y)||_1] \quad (8)$$

Overall, our total loss is

$$L(G_X, G_Y, D_X, D_Y) = L_{GAN}(G_X, D_Y, X, Y) + L_{GAN}(G_Y, D_X, X, Y) + \lambda_C L_{cyc}(G_X, G_Y) + \lambda_T L_{tempo}(G_X, G_Y) + \lambda_S L_{spe}(G_X, G_Y) \quad (9)$$

where λ_C, λ_T, and λ_S are tradeoff parameters. The complete network structure is shown in Fig. 2(d).

4 Implementation

4.1 Network architecture

Our model is based on Cycle-GAN [21], which has shown impressive results for unsupervised neural style transfer. For generators G_X, G_Y, and G_S, the networks contain several Unet blocks, which are identical except in the skip connections between each layer i in the encoder and layer $n - i$ in the decoder, where n is the total number of layers. We use 70×70 PatchGAN [19] structures for the discriminator networks D_X and D_Y, to discriminate 70×70 overlapping patches in the image as real or fake.

4.2 Training details

We train the model for a total of 200 epochs. The learning rate is kept at 0.0002 for the first 50 epochs and linearly decays to 0 over the following 150 epochs. The solver is the Adam optimization algorithm initialized from a Gaussian distribution with a standard deviation of 0.02 and mean of 0.0. During training, we set tradeoff parameters $\lambda_C = \lambda_T = 10$ and $\lambda_S = 1$.

For training and running the trained networks, we used a single Nvidia 2080Ti with 11 GB GPU memory. The inference time is about 20 ms per 512×512 frame, so this model can be used for real-time scenes and interactive image editing applications. We based our implementation on the Pytorch architecture.

4.3 Pipeline

In this application, the scene consists of hair together with other objects including head and body. All are passed to the pipeline in the usual way. Firstly, the hair is rendered by a fast shading model with low quality. We only need to extract the hair RGB colours from the standard graphics pipeline before the merging step as shown in Fig. 3. Next, these data are provided as input to the pre-trained generator G_X, which generates photorealistic hair RGB values. These are finally taken back into the pipeline and the merged output is exported to the frame buffer.

4.4 Training dataset

Existing head datasets are often used for facial identification purposes and have low image resolution. It would complicate matters to segment the hair parts from these datasets. For our work, we need a training set that only includes hair instead of the entire head or body. To this end, we rendered our own training data. For the input domain, we choose a simple model to rapidly render hair images without multi-scattering, inter-reflection, subtle blurring, and color-shifting effects. For the target domain, we used the Arnold Standard Hair shader, an advanced Monte Carlo ray tracing renderer built for producing visual effects; it renders hair based on the d'Eon model for the specular component and the Zinke model for the diffuse component. Hairstyle modeling and

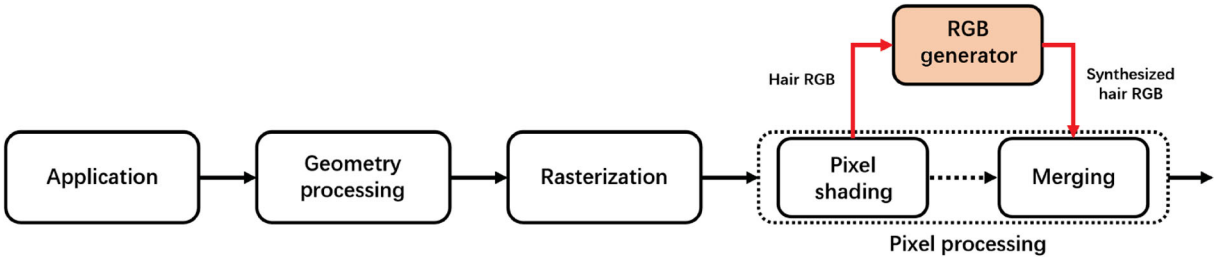
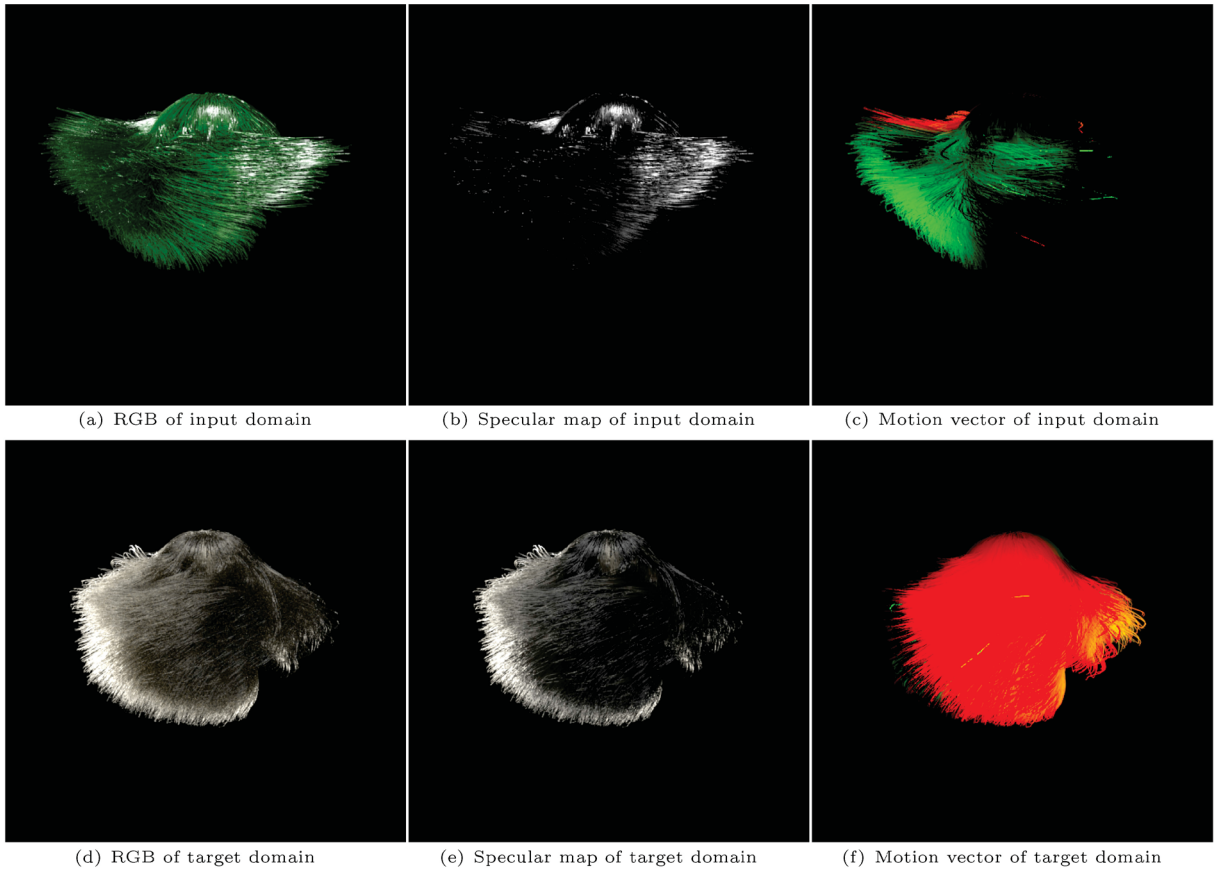


Fig. 3 Pipeline. The hair RGB colour is extracted before the merging stage and provided to the trained generator. The output is synthesized photorealistic hair RGB colour that is delivered to the merging stage.



(a) RGB of input domain (b) Specular map of input domain (c) Motion vector of input domain

(d) RGB of target domain (e) Specular map of target domain (f) Motion vector of target domain

Fig. 4 Our data includes three RGB channels for domain transfer, two channels for motion vector for temporal constraint, and three channels for specular map for highlight constraint.

other operations were performed on the 3D computer animation software Maya.

We designed various hairstyles, with diverse colors. For each hairstyle, we simulated and rendered a 400 frame sequence of size of 512×512. Each frame has three RGB channels for domain transfer, two channels of motion vector for temporal constraint, three channels of specular map for highlight constraint, one channel of depth map, and one channel of alpha map for merging. The total number of images is 6000, with 4000 for training and 2000 for testing.

4.5 Results

Figure 5 first shows results for different inputs and references. Our model can faithfully produce photorealistic hair appearance. In the right column, the generated hair not only exhibits realistic shapes but also inherits similar clusters from the reference. The results also show that our model can synthesize various hairstyles reasonably well. We can also consider whether the highlight constraint is necessary. Note that the highlight appears in the same position for input and output: our method can faithfully reproduce highlight appearances.

We next compare our results with those of previous work by excluding the influence of the reference. All the models were trained using the same dataset; see Fig. 6 for results generated by Cycle-GAN, Recycle-GAN, Mocycle-GAN, and our method. It is clear that our model achieves lower distortion and more faithful hair appearance. The results also show that previous models cannot reproduce exact highlight fields, while our method can faithfully produce highlight appearance. Without the highlight constraint, the results are less controllable, leading to unnatural-looking results.

In Fig. 8, we analyze whether the highlight constraint can be applied under different illumination conditions. We used inputs rendered for scenes with different illumination intensity and direction. The results illustrate that our method can faithfully produce highlight appearances under different illumination conditions.

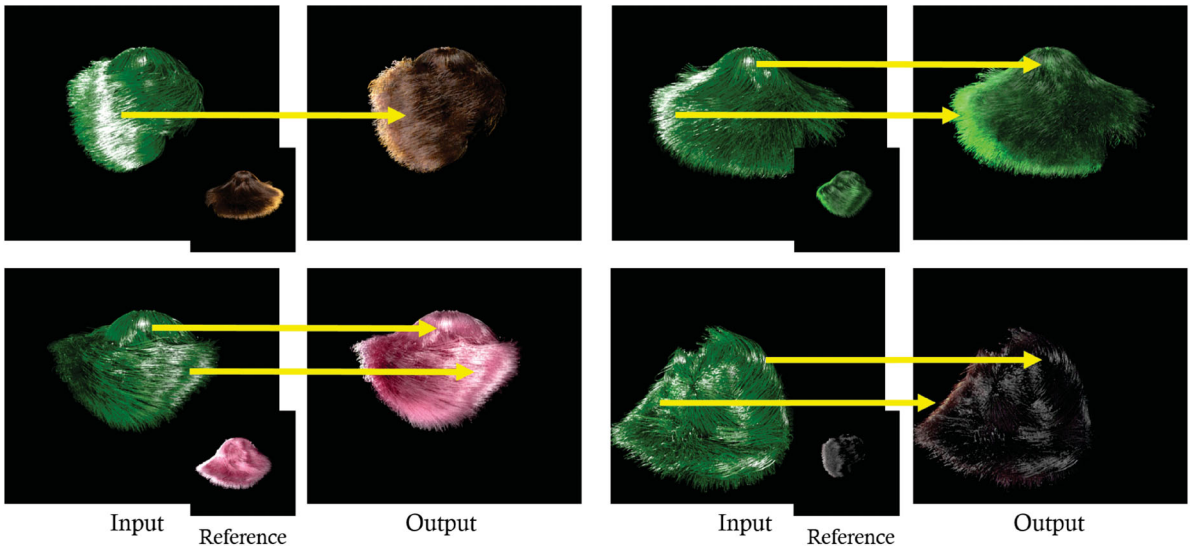


Fig. 5 Various coloring examples, showing original inputs, output results, and different kinds of references are given.

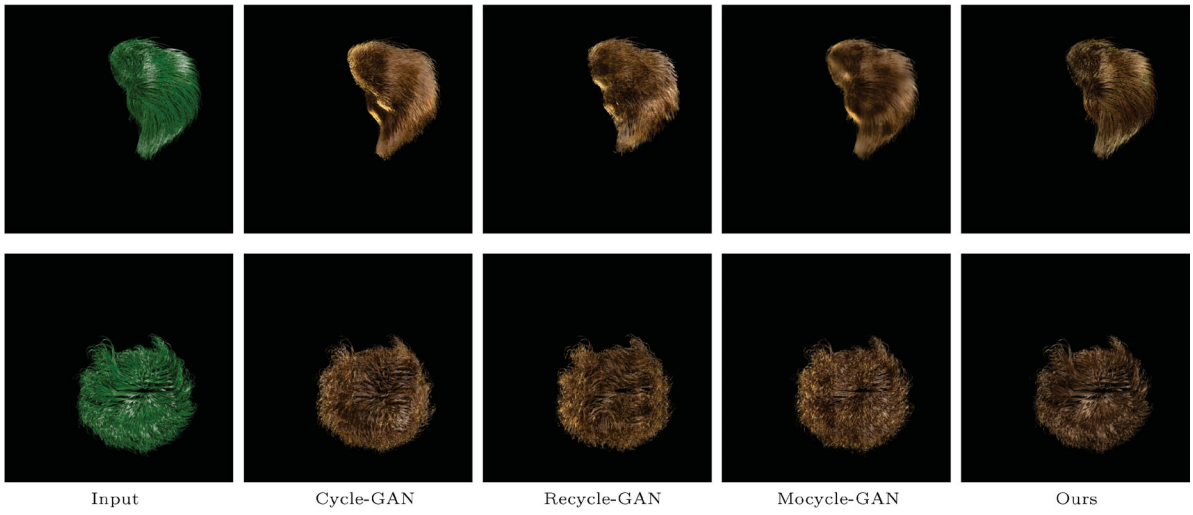


Fig. 6 Examples of previous works' results and our results.

Figure 7 shows the optical flow from Mocycle-GAN and our motion vector. Clearly, instead of synthesizing an approximate optical flow, our motion vector contains more exact information about motion direction and speed.

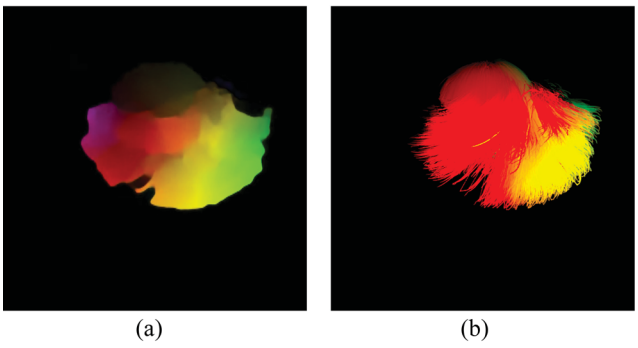


Fig. 7 Image (a) is synthesized by Flownet2 from a Mocycle-GAN model. Image (b) is directly generated by the Arnold renderer and used for predicting the future frame in our model.

Lastly, we compare the temporal coherence with previous methods. We removed the highlight constraint and reference, and trained the model using sequences of consecutive hair images. The results in Fig. 9 show the benefit of using motion vectors for video style transfer. Our results show faithful temporal coherence in consecutive frame sequences. Profiting from the spatio-temporal constraint, Recycle-GAN achieves better results than Cycle-GAN which only considers cycle consistency at the frame level. Unlike Recycle-GAN, our model uses precise motion vectors directly produced by the

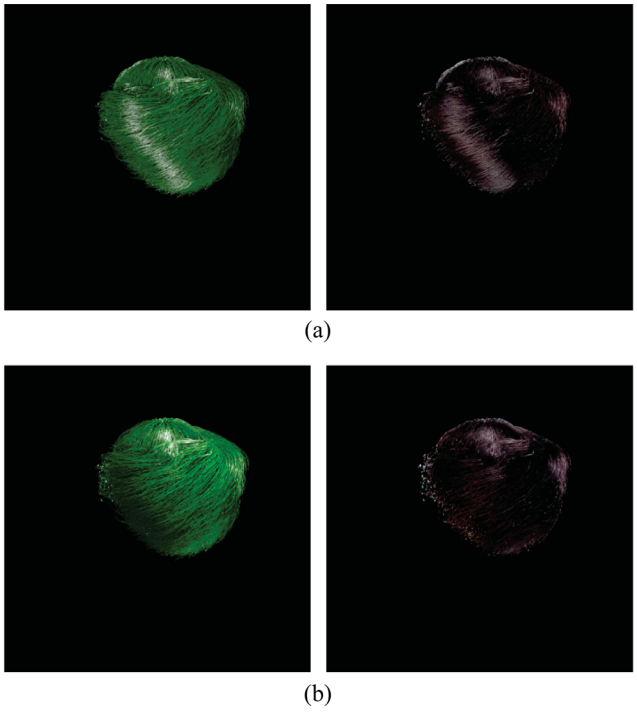


Fig. 8 Examples of illumination appearance. Using the highlight constraint, the highlight appears in a position close to the input highlight: our method can faithfully produce highlight appearances under different illumination conditions.

graphics pipeline, giving better results than previous methods. A temporal coherence comparison is given in a video in Electronic Supplementary Material. It confirms the effectiveness of our motion vector temporal constraint; a quantitative comparison is given later.

5 Performance analysis and evaluation

5.1 Speed

We need a state-of-the-art rendering method to provide the target domain dataset; we used hair images provided by Maya Arnold. Each image took more than 50 s to render. For the source domain, we used a fast rendering method that took less than 0.7 s to compute each frame.

The limited memory of the GPU prevented us from applying the work to high-resolution hair: the intermediate layers with their feature maps can take up huge amounts of memory, so we only implemented this model at 512×512 resolution.

Table 1 shows that our model is dozens of times faster than directly photorealistically rendering hair. Rendering time varies somewhat for differing hairstyles and motions.

5.2 Quality

We adopt Fréchet inception distance (FID) [24] for RGB channel evaluation. FID is widely applied in evaluation of generated images and is a metric

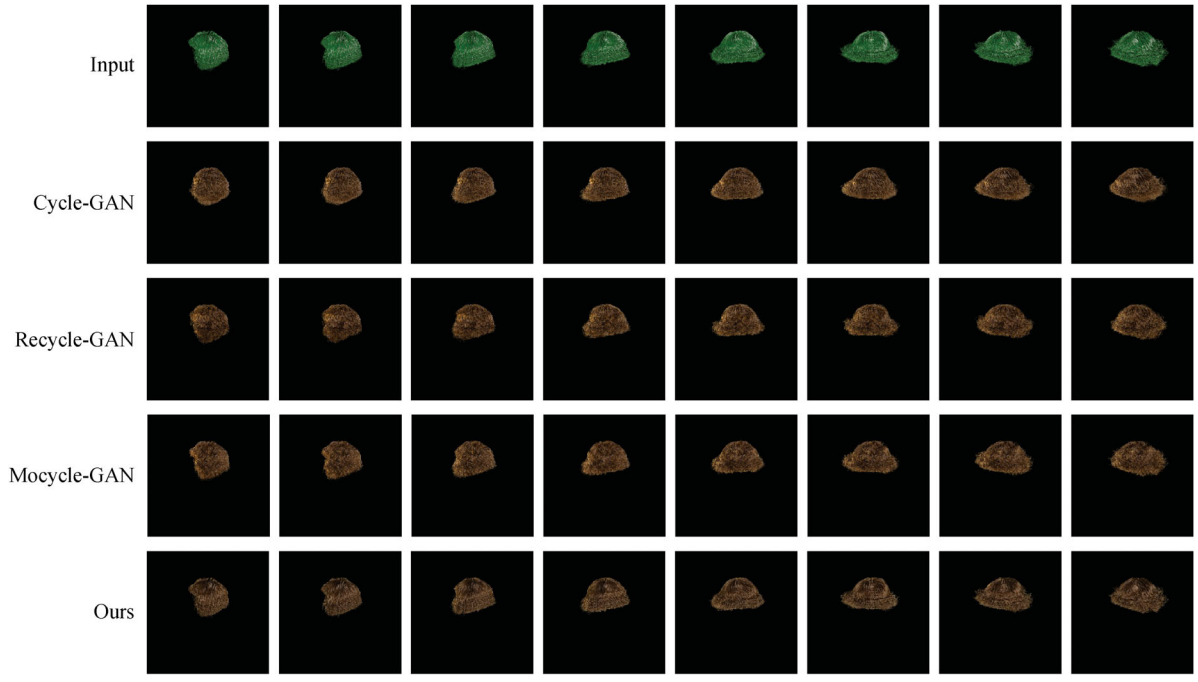


Fig. 9 Examples of video-to-video results by Cycle-GAN, Recycle-GAN, Mocycle-GAN, and our model. The original inputs and the output results by different models are given. Each row denotes one sequence of frames.

Table 1 Performance. We rendered the target domain dataset using the Arnold renderer, and the input domain using a fast rendering model. Our model generates one frame in 20 ms on average. It is dozens of times faster than Arnold

	Rendering	Synthesis	Total
Arnold	> 50 s	N/A	> 50 s
Ours	$<$ **0.70 s**	**20 ms**	$<$ **0.72 s**

for calculating the feature distance between real and generated images using a pre-trained inception network. Features are extracted from the RGB images to compute the FID score; a lower score indicates that the result is closer to the target domain.

Table 2 shows FID scores for Cycle-GAN, Recycle-GAN, and our method. Recycle-GAN performs better than Cycle-GAN by considering the consistency of the domain and time cycles through spatio-temporal constraints. Furthermore, Mocycle-GAN promotes pixel-wise temporal consistency by warping the synthetic frame with optical flow, so is also better than Cycle-GAN. However, our method performs best, using motion vectors that are more accurate than Mocycle-GAN's.

Table 2 Fréchet inception distance (FID) translation quality scores for hair-to-hair synthesis of RGB channels

	FID
Cycle-GAN	3.97
Recycle-GAN	3.72
Mocycle-GAN	3.84
Ours	**3.70**

We further evaluate the specular map for realistic hair illumination using peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) as image quality metrics. PSNR is widely applied in evaluating image quality; SSIM assesses perceived quality of images in terms of structure, luminance, and contrast.

Here, given a pair of RGB images from the output and corresponding input domains, the specular generator synthesizes specular maps for both output and input. As shown in Table 3, by encouraging the specular constraint, our model performs consistently better than other methods in terms of the two metrics. This confirms the effectiveness of our specular constraint, which can retain more highlight information in hair synthesis.

Table 3 Peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) assessment of specular map translation quality for hair illumination appearance

	PSNR	SSIM
Cycle-GAN	24.99 dB	0.91
Recycle-GAN	25.11 dB	0.91
Mocycle-GAN	25.84 dB	0.93
Ours	**27.07 dB**	**0.98**

6 Conclusions

We have presented an energy-conserving model that synthesizes photorealistic hair images from low-quality hair images. Our method transfers hair images to those with desired appearances according to reference hair. We provide continuity for the translation of a hair image sequence by temporal constraints, which for the first time apply motion vectors to improve the structure and temporal continuity of hair. Our method also introduces specular constraints to ensure faithful highlight appearance.

Compared to previous image translation methods, our model generates more convincing results and improves preservation of illumination from the input. Our model is dozens of times faster than traditional state-of-art hair rendering models. Our work suggests that unsupervised image translation can faithfully reproduce photorealistic hair animation and significantly reduce computational expense. We also hope that our novel framework can be applied to the shading of semi-transparent objects like sunset glows and colored glass.

Acknowledgements

Zhi Qiao acknowledges receipt of Japanese government scholarships (MEXT). This work was partially supported by JSPS KAKENHI, Grant Number JP19K11990.

Electronic Supplementary Material Supplementary material is available in the online version of this article at `https://doi.org/10.1007/s41095-020-0201-9`.

References

[1] D'Eon, E.; Francois, G.; Hill, M.; Letteri, J.; Aubry, J.-M. An energy-conserving hair reectance model. In: Proceedings of the 22nd Eurographics Conference on Rendering, 1181–1187, 2011.

[2] Zinke, A. Photo-realistic rendering of fiber assemblies. In: Ausgezeichnete Informatikdissertationen, 2007.

[3] Zhu, J. Y.; Park, T.; Isola, P.; Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, 2242–2251, 2017.

[4] Kajiya, J. T.; Kay, T. L. Rendering fur with three dimensional textures. *ACM SIGGRAPH Computer Graphics* Vol. 23, No. 3, 271–280, 1989.

[5] Yan, L.-Q.; Tseng, C.-W.; Jensen, H. W.; Ramamoorthi, R. Physically-accurate fur reflectance: Modeling, measurement and rendering. *ACM Transactions on Graphics* Vol. 34, No. 6, Article No. 185, 2015.

[6] Marschner, S. R.; Jensen, H. W.; Cammarano, M.; Worley, S.; Hanrahan, P. Light scattering from human hair fibers. *ACM Transactions on Graphics* Vol. 22, No. 3, 780–791, 2003.

[7] Ward, K.; Bertails, F.; Kim, T. Y.; Marschner, S. R.; Cani, M. P.; Lin, M. C. A survey on hair modeling: Styling, simulation, and rendering. *IEEE Transactions on Visualization and Computer Graphics* Vol. 13, No. 2, 213–234, 2007.

[8] Moon, J. T.; Walter, B.; Marschner, S. Efficient multiple scattering in hair using spherical harmonics. *ACM Transactions on Graphics* Vol. 27, No. 3, 1–7, 2008.

[9] Ren, Z.; Zhou, K.; Li, T. F.; Hua, W.; Guo, B. N. Interactive hair rendering under environment lighting. In: Proceedings of the ACM SIGGRAPH 2010 Papers, Article No. 55, 2010.

[10] Jansson, E. S. V.; Chajdas, M. G.; Lacroix, J.; Ragnemalm, I. Real-time hybrid hair rendering. In: Proceedings of the Eurographics Symposium on Rendering, 1–8, 2019.

[11] Gatys, L. A.; Ecker, A. S.; Bethge, M. Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2414–2423, 2016.

[12] Johnson, J.; Alahi, A.; Li, F. F. Perceptual losses for real-time style transfer and super-resolution. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9906*. Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 694–711, 2016.

[13] Luan, F. J.; Paris, S.; Shechtman, E.; Bala, K. Deep photo style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 6997–7005, 2017.

[14] Wei, L. Y.; Hu, L. W.; Kim, V.; Yumer, E.; Li, H. Real-time hair rendering using sequential adversarial networks. In: *Computer Vision – ECCV 2018. Lecture Notes in Computer Science, Vol. 11208*. Ferrari, V.; Hebert, M.; Sminchisescu, C.; Weiss, Y. Eds. Springer Cham, 105–122, 2018.

[15] Chai, M.; Ren, J.; Tulyakov, S. Neural hair rendering. In: *Computer Vision – ECCV 2020. Lecture Notes in Computer Science, Vol. 12363*. Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 371–388, 2020.

[16] Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, 2672–2680, 2014.

[17] Gatys, L.; Ecker, A.; Bethge, M. A neural algorithm of artistic style. *Journal of Vision* Vol. 16, No. 12, 326, 2016.

[18] Jing, Y. C.; Yang, Y. Z.; Feng, Z. L.; Ye, J. W.; Yu, Y. Z.; Song, M. L. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics* Vol. 26, No. 11, 3365–3385, 2020.

[19] Isola, P.; Zhu, J. Y.; Zhou, T. H.; Efros, A. A. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5967–5976, 2017.

[20] Wang, T. C.; Liu, M. Y.; Zhu, J. Y.; Tao, A.; Kautz, J.; Catanzaro, B. High-resolution image synthesis and semantic manipulation with conditional GANs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8798–8807, 2018.

[21] Zhu, J. Y.; Park, T.; Isola, P.; Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, 2242–2251, 2017.

[22] Bansal, A.; Ma, S. G.; Ramanan, D.; Sheikh, Y. Recycle-GAN: Unsupervised video retargeting. In: *Computer Vision – ECCV 2018. Lecture Notes in Computer Science, Vol. 11209*. Ferrari, V.; Hebert, M.; Sminchisescu, C.; Weiss, Y. Eds. Springer Cham, 122–138, 2018.

[23] Chen, Y.; Pan, Y. W.; Yao, T.; Tian, X. M.; Mei, T. Mocycle-GAN: Unpaired video-to-video translation. In: Proceedings of the 27th ACM International Conference on Multimedia, 647–655, 2019.

[24] Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 6629–6640, 2017.

Zhi Qiao is a Ph.D. candidate at the University of Tokyo, Japan. His research interests center on computer graphics and computer vision. He is currently receiving a fellowship from the Ministry of Education, Culture, Sports, Science and Technology (MEXT).

Takashi Kanai is an associate professor in the Graduate School of Arts and Sciences, the University of Tokyo. His research interests include geometry processing and physics-based animation in computer graphics. He received his doctoral degree in engineering from the University of Tokyo in 1998. He is a member of ACM, ACM SIGGRAPH, IIEEJ (the Institute of Image Electronics Engineers of Japan), and IPSJ (Information Processing Society of Japan).