HairStyle Editing via Parametric Controllable Strokes

Xinhui Song, Chen Liu, Youyi Zheng, Zunlei Feng, Lincheng Li, Kun Zhou, *Fellow, IEEE*, and Xin Yu

***Abstract*—In this work, we propose a stroke-based hairstyle editing network, dubbed HairstyleNet, allowing users to conveniently change the hairstyles of an image in an interactive fashion. Different from previous works, we simplify the hairstyle editing process where users can manipulate local or entire hairstyles by adjusting the parameterized hair regions. Our HairstyleNet consists of two stages: a stroke parameterization stage and a stroke-to-hair generation stage. In the stroke parameterization stage, we first introduce parametric strokes to approximate the hair wisps, where the stroke shape is controlled by a quadratic Bézier curve and a thickness parameter. Since rendering strokes with thickness to an image is not differentiable, we opt to leverage a neural renderer to construct the mapping from stroke parameters to a stroke image. Thus, the stroke parameters can be directly estimated from hair regions in a differentiable way, enabling us to flexibly edit the hairstyles of input images. In the stroke-to-hair generation stage, we design a hairstyle refinement network that first encodes coarsely composed images of hair strokes, face, and background into latent representations and then generates high-fidelity face images with desirable new hairstyles from the latent codes. Extensive experiments demonstrate that our HairstyleNet achieves state-of-the-art performance and allows flexible hairstyle manipulation.**

***Index Terms*—Hairstyle editing, hairstyle transfer, parameterized hair strokes, stroke-controllable.**

I. Introduction

HUMAN hairstyles play an important role in depicting personal characteristics and can improve the image of a person. Editing the hairstyle of a portrait image is highly desirable in many scenarios. It not only helps hair salon customers to preview whether a hairstyle is suitable or not, but also offers users to customize a hairstyle for virtual conferences or social networking softwares. However, due to the intricate hair peculiarity (such as light scattering and semi-transparent occlusions) and various hairstyles, it is often very challenging to edit portrait hairstyles in 2D images flexibly.

Early hairstyle generation methods rely on 3D hair models to generate hairstyle images. For example, neural hair rendering methods [1], [2] synthesize portrait images with various hairstyles by leveraging 3D hair models and the hair appearance of a reference image. However, constructing strand-based 3D hair models is usually time-consuming and difficult for non-experts.

Alternatively, attribute-based methods [3], [4], [5], [6], [7] have been proposed to edit hairstyles based on the gender and color attributes, and thus they cannot change hairstyles freely. Recently, MichiGAN [8], LOHO [9] and Barbershop [10] produce new hairstyles by transferring hairstyles' structure, appearances and shapes via generative adversarial networks [11]. Since these methods focus on global hairstyle transferring, they cannot control the generation of local hair details. DeepFaceEditing [12] and SketchHairSalon [13] present a flexible way to generate hairstyles by employing sketches. Since they do not constrain the appearance consistency between the facial and hair regions, the generated images may suffer unnatural boundaries or artifacts.

In this paper, we introduce a novel parametric controllable stroke-based hairstyle editing network, namely HairstyleNet, to flexibly revise local structure, colors, and global styles of hair in a portrait image. In a nutshell, HairstyleNet first parameterizes a hair region by parametric strokes (i.e., quadratic Bézier curves), then adjusts the stroke parameters according to users' preference, and finally renders the edited parameterized hair and facial region into a photorealistic portrait image, as illustrated in Fig. 1. Our proposed HairstyleNet consists of a stroke parameterization stage and a stroke-to-hair generation stage. In the stroke parameterization stage, we utilize strokes to simulate hair wisps, where a quadratic Bézier curve and a thickness parameter are adopted to control stroke shapes, and stroke colors are controlled by the color and transparent parameters. Then, we formulate the stroke parameterization as a parameter searching process that aims to maximize the similarity between the hair region and the rendered strokes. Since rendering strokes with thickness is not differentiable, minimizing the differences between the hair region and the rendered strokes to achieve the stroke parameters becomes non-differentiable.

To solve this problem, we instead leverage a neural renderer to imitate the non-differentiable rendering process from stroke parameters to a stroke image. More specifically, we first generate image pairs of stroke images from stroke parameters (a

Manuscript received 18 July 2022; revised 10 January 2023; accepted 30 January 2023. Date of publication 3 February 2023; date of current version 26 June 2024. This work was supported in part by the ARC-Discovery under Grant DP220100800 and in part by the ARC-DECRA under Grant DE230100477. Recommended for acceptance by T. Ju. *(Xinhui Song and Chen Liu authors contributed equally to this work.) (Corresponding author: Lincheng Li.)*

Xinhui Song and Lincheng Li are with the Fuxi AI Lab of Netease, Inc., Hangzhou, Zhejiang 310052, China (e-mail: songxinhui@corp.netease.com; lilincheng@corp.netease.com).

Youyi Zheng, Zunlei Feng, and Kun Zhou are with the Zhejiang University, Hangzhou, Zhejiang 310027, China (e-mail: youyizheng@zju.edu.cn; zunleifeng@zju.edu.cn; kunzhou@acm.org).

Chen Liu and Xin Yu are with the University of Queensland, Brisbane, QLD 4072, Australia (e-mail: yenanliu36@gmail.com; xin.yu@uts.edu.au).

This article has supplementary downloadable material available at https://doi.org/10.1109/TVCG.2023.3241894, provided by the authors.

Digital Object Identifier 10.1109/TVCG.2023.3241894

1077-2626 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

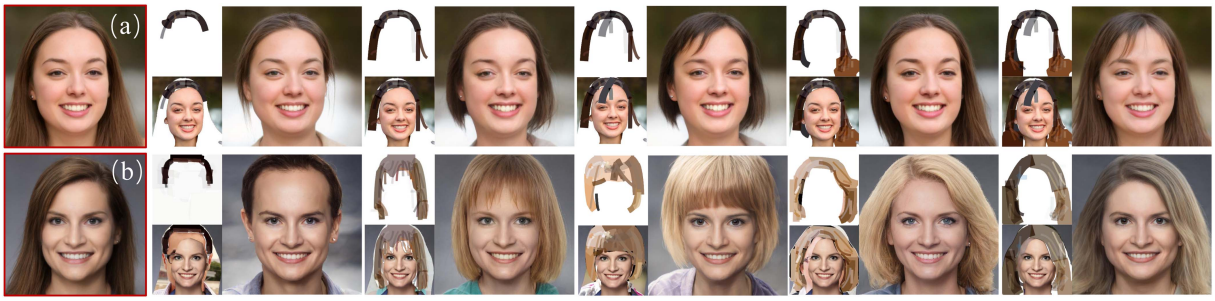


Fig. 1. Given an input image (a) or (b), our method HairstyleNet manipulates hairstyles by controlling the stroke parameters. Specifically, a user can leverage our hairstyle editing tool to manipulate stroke-based hair images (top-left). Then, the stroke-based hair and face images are composed together (bottom-left). Finally, synthesized portrait images with new hairstyles are obtained from the composed versions (right).

non-differentiable rendering pipeline), and then train the neural renderer with the image pairs. After the neural renderer has been fully trained, we incorporate this differentable renderer into our stroke parameterization process. In this manner, our proposed stroke parameterization pipeline is fully differentiable. As a result, we can obtain stroke parameters to represent hair region in images, and then users can flexibly edit the stroke shapes, thicknesses and colors to change the hairstyles with our interactive hairstyle editing tool.

In the stroke-to-hair generation stage, we design a hairstyle refinement network for generating a natural and high-fidelity face image from the composed image including the stroke-based hair region, face region, and background. We observed that some facial and hair regions in the composed images may look inharmonious after users' editing in previous works. Thus, we impose a global appearance consistency constraint on the composed portraits. Specifically, we train a refinement encoder to project a composed image to the style space of a pre-trained StyleGAN2 and then synthesize a high-fidelity and realistic portrait image by utilizing the powerful reconstruction ability of the StyleGAN2 generator. Quantitative and qualitative comparisons demonstrate the superiority of our method on hairstyle editing and transferring.

Overall, the main contributions are summerized as follows:

- We introduce the first parametric controllable hairstyle editing pipeline, HairstyleNet, to flexibly change hairstyles, including hair shapes, colors and structure, by manipulating the parameterized hair strokes.
- We achieve a fully differentiable hairstyle editing network by adopting a neural renderer to imitate the rendering process from stroke parameters to a stroke image, thus significantly facilitating the network training and leading to superior hair editing effects.
- We provide an interactive user-friendly hair manipulation editing tool, which can be easily adopted in other multimedia applications.

II. Related Works

A. Deep Image Generation

Generative adversarial networks (GANs) [11] have shown remarkable results in image generation tasks. GANs are widely used in highly realistic image generation [11], [14], [15], [16], [17], [18], facial attribution image generation [3], [4], [19], [20], [21], image conversion [22], [23], [24], [25], [26], [27], [28], and super-resolution [29], [30], [31], [32], [33], [34], [35], [36], [37].

The advent of StyleGAN [38] and StyleGAN2 [39] significantly improves image generation quality. StyleGAN implicitly incorporates hierarchical latent information into its shallow and deep layers to synthesize high-fidelity human faces. Meanwhile, many methods [40], [41], [42], [43], [44] project images into the StyleGAN latent space for generating high-quality and diverse outputs. These StyleGAN inversion methods can be roughly grouped into two categories: learning-based and optimization-based. The learning-based methods leverage a neural network to learn the correspondences between latent codes and images, and require the whole process to be differentiable. The optimization based algorithms back-propagate the gradients of reconstruction errors to update the latent codes.

B. Facial Attribute Manipulation

Attribute-based facial image manipulation methods [3], [4], [5], [6], [45], [46], [47], [48], [49], [50], [51] change face appearances by controlling specific facial attributes, such as gender, age, expression, eyeglasses, and hair. These methods usually adopt the generative adversarial networks or encoder-decoder architectures. IcGAN [19] introduces an encoder to learn an inverse mapping of a conditional GAN. An image is first encoded into a latent representation, and then decoded to a target image conditioning on the desired attribute labels. StarGAN [3] performs multi-domain attribute image translation by changing the attribute labels of input images. SMILE [47] achieves multiple facial attribute manipulation with the help of a guided exemplar. SC-EFGAN [52] generates high-quality portraits by user-friendly input in terms of the colors and shapes of sketches. Deep Plastic Surgery [53] presents a coarse-to-fine sketch refinement strategy to achieve portrait editing by sketches. Although existing facial attribute manipulation methods can edit hairstyles, they can only edit simple hair attributes to a limited extent, such as changing hair colors or bangs.

C. Hair Model and Image-Based Hair Editing

Hairstyle is a critical component of a character. Hair acquisition pipelines capture hair models from various inputs, including multi-view images [54], [55], a single-view image [56], [57], [58], [59], point clouds [60], [61], [62] and RGB-D images [55], [63]. Several hair editing methods have been proposed using a

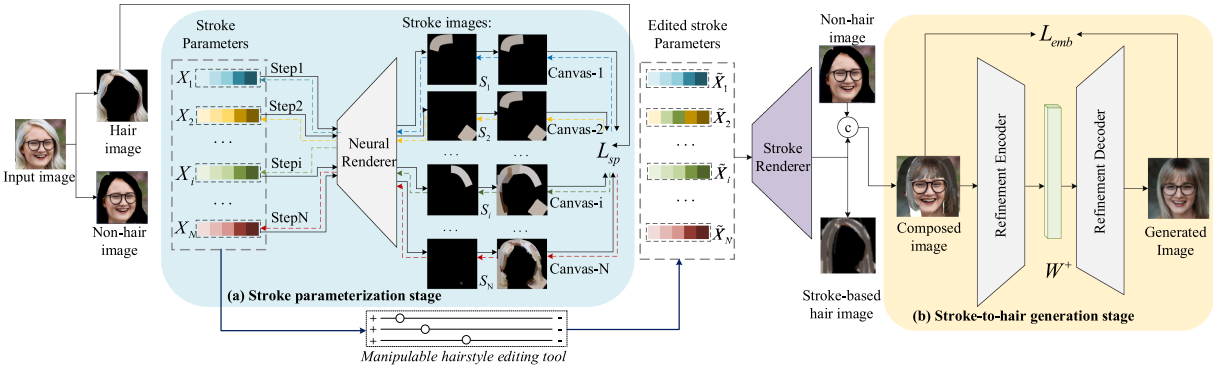


Fig. 2. The pipeline of our framework consists of two stages. In part (a), the hair region of a reference image is rendered into a stroke image, where X_i is a set of stroke parameters (including quadratic Bézier curve points, color, transparency, and thickness), N is the total number of strokes used to draw the stroke-based hair image, S_i is the stroke image rendered with the set of stroke parameters, and L_{sp} is the loss between the hair image and the canvas. Users can edit the stroke parameters X_i by the manipulable hairstyle editing tool, and the edited stroke parameters $\hat{X}_i$ will be rendered as the stroke-based hair image. In part (b), the refinement encoder module is used to map the composed image into the latent codes W^+, and the refinement decoder generates the individual image with a new hairstyle according to the latent codes of the composed image.

3D hair model [56], [64]. Nevertheless, the 3D modeling and rendering progress is expensive.

Existing methods exploit hair editing to synthesize realistic 2D hair images directly. H-GAN [65] consists of an encoder-decoding sub-network, a GAN sub-network, and a recognition sub-network. H-GAN edits hair by adding the hairstyle vector to a face latent representation and then decoding the modified latent representation. HairMapper [66] designs a pipeline to remove hair by finding a specific path in the latent space of StyleGAN. HairClip [67] proposes a hair editing interaction mode by leveraging the powerful image text representation capability of the Contrastive Language-Image Pre-Training (CLIP) model. While HairClip performs high-quality global hairstyle editing, it can not edit specific local hair regions intuitively and flexibly. LOHO [9] uses a two-stage optimization and gradient orthogonalization for reference hairstyle transfer, including hair structure and appearance. Barbershop [10] suggests a new latent space for image blending when transferring hair shapes, appearances, and coarse structure to a target image. Although these works achieved promising progresses on hair generation conditioning specific reference inputs, they often do not favor flexible editing.

Olszewski et al. [68] synthesize realistic facial hair from an input image with a user-assigned region and sparse sketches. MichiGAN [8] achieves hair manipulation by disentangling hair into four orthogonal attributes, including shapes, appearances, structure, and background. SketchHairSalon [13] and DeepFaceEditing [12] generate realistic hair images from freehand sketches. These sketch-based methods often suffer unnatural boundaries. In this paper, we propose a parametric controllable stroke-based hairstyle editing method, allowing users to manipulate various attributes of hair freely in an interactive manner.

III. Proposed Method

Given an input image I_{in}, HairstyleNet aims at generating the individual image with an edited hairstyle according to user operations. As shown in Fig. 2, our method consists of two stages: (1) a hair stroke parameterization stage, in which hair will be represented by parametric strokes; (2) a stroke-to-hair generation stage, in which the stroke-based hair image will be blended with the non-hair region to synthesize a realistic portrait image.

A. Stroke Parameterization Stage

A hairstyle is made up of tens of thousands of hair strands. Since controlling each hair strand individually is impractical, we model the hairstyle as multiple overlapping hair wisps. Inspired by stylized neural painting [69], we represent each hair wisp as a quadratic Bézier curve with thickness, color and transparency, dubbed stroke S. Specifically, the stroke parameters $X \in \mathbb{R}^{11}$ is defined as $X = \{P_0, P_1, P_2, d, c, a\}$, where the $d \in \mathbb{R}$, $c \in \mathbb{R}^3$, and a represent the thickness, color, and transparency of the quadratic Bézier curves, and the $P_i, i \in [0, 2]$ are three 2D control points in the pixel space:

$$B(k) = (1-k)^2 P_0 + 2(1-k)kP_1 + k^2 P_2,\ k \in [0, 1]. \quad (1)$$

Since modeling hair by strokes is a one-to-many mapping, we incrementally search the parameters of each stroke in an iterative manner, as illustrated in Fig. 2(a). The searching process mimics the drawing procedure on an empty canvas. We gradually add new strokes to the blank canvas and update the stroke parameters progressively at each step by maximizing the similarity between the input and the canvas.

The existing stroke renderer draws a stroke image by applying multiple continuous colored rectangle segments along the quadratic Bézier curve's direction. As a result, the stroke renderer is often non-differentiable, increasing the difficulty of the similarity maximization process between the hair image and the stroke-based images. To solve this problem, a neural renderer is employed to imitate the stroke renderer, motivated by [69]. Specifically, we obtain pairs of stroke parameters X and stroke images S generated by a stroke renderer. Then, a neural renderer, which takes stroke parameters as input and outputs strokes, is trained with a pixel-wise similarity loss between a stroke image and its ground-truth S.

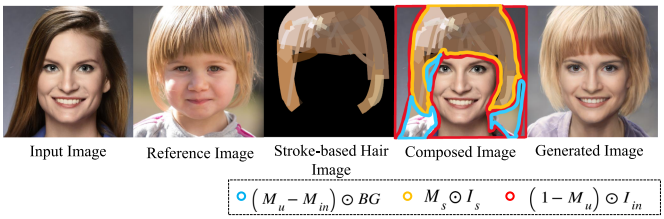


Fig. 3. Illustration of components and masks in the composed image.

With the help of the neural renderer, we design a novel stroke parameterization framework to parameterize the hair image to a sequence of strokes, which are represented by stroke parameters. Specifically, for each step, a neural renderer first predicts a stroke image S_i from a randomly sampled stroke parameter X_i. Then, X_i will be updated by T iterations to maximize the similarity between the hair image and the current canvas. After N steps, we obtain stroke parameters $\{X_i\}_{i=1}^{N}$ that would fit the hair image. The stroke parameters are orderly rendered on the canvas, and we name the final canvas as a stroke-based hair image. Afterwards, users can flexibly edit the hairstyles of input images, including shapes, structure, and colors, via controlling the stroke parameters with our interactive hairstyle editing tool, as seen in Fig. 4.

B. Stroke-to-Hair Generation Stage

As shown in Fig 2, the edited stroke parameters are rendered into the stroke-based hair image I_s by the stroke neural renderer, and then it is combined with the non-hair image as a composed image I_c. In the stroke-to-hair generation stage, ensuring the edited hair matches the face and unifying the appearance style of the generated image should be emphasized and are our focus.

In this work, we provide two ways of hairstyle editing in the editing system for users: (i) stroke-based local hairstyle editing and (ii) reference-based hairstyle editing. For reference-based hairstyle editing, users can transfer the hairstyles of reference images to input images. As illustrated in the second row of Fig. 1, the stroke-based hair regions of the reference images cannot fully cover the non-hair regions of the input images. Due to the shape differences between the reference hair regions and the input hair regions, blank areas would occur around the face regions in composed images. Hence, we leverage the regions obtained by $(M_s - M_{in}) \odot BG$ to fill these blank areas. Specifically, the composed image I_c is defined as:

$$I_c = (1 - M_u) \odot I_{in} + M_s \odot I_s + (M_u - M_{in}) \odot BG, \quad (2)$$

where M_{in} and the M_s represent the hair region masks of the input image and the stroke-based image, and the $M_u = M_s \bigvee M_{in}$ is the union hair mask. Here, the operator $-$ indicates the element-wise subtraction of two matrices, $\bigvee$ represents the element-wise OR operation, and $\odot$ is the element-wise multiplication operation, as illustrated in Fig. 3.

For stroke-based hairstyle editing, our system allows users to edit the local hairstyle of input images. In other words, strokes in the composed images are rendered from the hair regions of input images. In this fashion, the blank areas will not corrupt the non-hair regions in composed images. In general, blank areas rarely appear when users add or edit strokes on a composed image. For the stroke removal operation, blank areas usually appear in the hair regions. As seen in Fig. 1, since face regions are not corrupted in the composed images, the identity information in generated images is well maintained even though there are blank areas in hair regions. Therefore, choosing a BG image is not mandatory in stroke-based local hairstyle editing operations.

In practice, the style and quality of the generated hair region and the input non-hair region may not be compatible. To guarantee the overall style appearance harmonious of the generated image, we first project the composed image into a latent representation by the refinement encoder E, i.e., $\mathcal{W}+ \in \mathbb{R}^{18\times512}$, and then the refinement decoder synthesizes the high-fidelity portrait image according to the latent code of the composed image, as illustrated in Fig. 2(b). More specifically, the refinement encoder builds upon the pixel2style2pixel encoder [40], [41] that enables the encoded composed image to be mapped to a realistic high-quality image in the latent space of the refinement decoder. Here, we adopt a pre-trained StyleGAN2 as our refinement decoder due to its powerful image generation ability. In this fashion, we can constrain a consistent style between hair and face regions as well as achieve fine-grained hair details in the final results.

C. Training Details and Objective Function

1) Training Objectives for Stroke Parameterization: In the stroke parameterization stage, we aim at converting the whole hair region of the input image into a stroke-based image in order to edit local hair. This would require that the stroke-based image has the same appearance and structure as the input hair region. Therefore, we apply the following loss functions to constrain the appearance and structure similarity.

The $\mathcal{L}_1$ Loss and the LPIPS Loss: We first adopt the pixel-wise $\mathcal{L}_1$ loss to measure the low-level intensity differences between the hair images and generated strokes, as follow,

$$\mathcal{L}_1(I_s, I_h) = ||I_s - I_h||_1, \quad (3)$$

where I_s, I_h are the stroke-based hair image and the input hair image, respectively. In addition, the perceptual loss $\mathcal{L}_{LPIPS}$ [70] is introduced to constrain the overall appearance and structure similarity in the hair region, expressed by,

$$\mathcal{L}_{LPIPS}(I_s, I_h) = \sum_{j=1}^{5} \frac{1}{N_j} ||F_j(I_s) - F_j(I_h)||_2, \quad (4)$$

where F_j indicates the feature maps extracted from the layers of *pool1, pool2, pool3, pool4, pool5* of VGG-16, respectively, and N_j is the pixel number in the jth layer output.

Optimal Transport Loss: Stylized neural painting [69] proves the optimal transport loss can effectively minimize the transportation cost of moving strokes from one location to another. We thus employ the optimal transport to regulate the stroke locations in a stroke-based image. To be specific, the optimal

transportation loss $\mathcal{L}_{ot}$ is defined as:

$$\mathcal{L}_{ot}(v_s, v) := \langle P', C\rangle, s.t. P' = \arg\min_{P\in\mathcal{U}} \langle P, C\rangle - \frac{1}{\lambda}h(P). \tag{5}$$

The entropy is defined as $h(P) = -\sum_{i,j=1}^{d} p_{ij}\log p_{ij}$, where p_{ij} is a joint probability of P. $P \in \mathbb{R}_{+}^{n\times n}$ is the joint probability matrix, whose (i, j)-th element denotes the joint probability of the ith pixel in I_s and jth pixel in I_h, and n is the number of pixels in an image. C is the cost matrix, indicating the costs of moving a "unit pixel mass" from one position in I_s to another one in I_h, and its (i, j)-th element is measured by the euclidean distance between the ith pixel's location in I_s and jth pixel's location in I_h.

Stroke Width Regularization Loss: To prevent strokes from excessive overlapping, we adopt the width regularization loss $\mathcal{L}_d$ to constrain the widths of strokes. The width regularization loss is defined as:

$$\mathcal{L}_d = \sum_{i\in N} ||d_i||_1, \tag{6}$$

where the N is the number of strokes, and the d_i is the width of the ith stroke.

Without the width constraint, all strokes tend to cover the hair region with a large width, thereby failing to depicting local hairstyles. In other words, severe stroke overlapping would exert negative influences on achieving the local hairstyle editing for users. Moreover, extensive overlapped strokes will make stroke selection in the editing phase difficult. With the width constraint, strokes are generated in a coarse-to-fine rendering process thanks to the progressive learning strategy. Strokes rendered first in order depict the shapes and the structure of the global hair regions, and thus they usually have large widths. Strokes rendered later in order describe the fine structure and shapes of the local hair regions, usually with small widths.

Hair Orientation Loss: We introduce a *hair orientation loss* $\mathcal{L}_{ori}$ in the hair region for maintaining the hair internal structure between the hair image I_h and stroke-based hair image I_s.

We utilize dense 2D orientation fields to depict the hair structure for each pixel of the hair region using a Gabor filter [71]. Specially, we first use 32 oriented filter kernels $\{K_\theta, \theta \in [0, \pi)\}$ to estimate the trigonometric orientation map θ^* and its associated confidence map γ of the hair region in the image I. The orientation and the confidence maps of the pixel position p are defined as follows,

$$\theta_p^* = \arg\max_{\theta} |(K_\theta \otimes I_h)_p|, \tag{7}$$

$$\gamma_p = \max_{\theta} |(K_\theta \otimes I_h)_p|, \tag{8}$$

where $\otimes$ represents the convolution operator. Then, we convert the trigonometric orientation map θ^* into a two-channel continuous orientation map O' and a one-channel continuous orientation map ${o_p}'$ for each pixel p, written by,

$${O_p}' = \left[\cos\left(2\cdot\theta_p^*\right) * \gamma_p, \sin(2\cdot\theta_p^*) * \gamma_p\right], \tag{9}$$

$${o_p}' = \arctan\left({O_p}'\right)/2. \tag{10}$$

Then, we calculate per-pixel MAE orientation distance in the hair region as the hair orientation loss, defined as:

$$\mathcal{L}_{ori} = ||(O(I_s) - O(I_h))||_1, \tag{11}$$

where $O(I_s)$ and $O(I_h)$ are the orientation maps of the stroke image and the hair region of the input image, respectively.

Total Loss for Stroke Parameterization: The overall loss function in the stroke parameterization stage is defined by,

$$\mathcal{L}_{sp} = \alpha_1\mathcal{L}_1 + \alpha_2\mathcal{L}_{LPIPS} + \alpha_3\mathcal{L}_d + \alpha_4\mathcal{L}_{ot} + \alpha_5\mathcal{L}_{ori}, \tag{12}$$

where α_1, α_2, α_3, α_4, α_5 are empirically set to 1, 1, 0.005, 0.1, 1, respectively.

2) Training Objectives for the Refinement Autoencoder: In this stage, we do not need to train the decoder part since we adopt a pre-trained StyleGAN2 as our refinement decoder. Thus, we only train the refinement encoder to effectively map a composed image to the latent space of the pre-trained StyleGAN2. In other words, the composed image that contains coarse hair strokes will be matched to its nearest StyleGAN2 latent code, which not only represents a photo-realistic identity-preserving face portrait but also exhibits the edited hairstyles.

To this end, we impose several loss functions on the generated face images. Since we do not ground-truth images after hairstyle editing, we first parameterize the hair region of an input image and then employ the input image as ground-truth and the stroke-parameterized image as input to train our refinement encoder E. Here, we utilize the pixel-wise $\mathcal{L}_1$ and $\mathcal{L}_{LPIPS}$ losses to enforce image intensity and feature similarities. To preserve identity information during hairstyle editing, an identity-preserving loss is used to constrain the difference between the facial features of the composed image I_c and the generated image I, defined by,

$$\mathcal{L}_{id} = 1 - \cos(F_{fr}(I), F_{fr}(I_c)), \tag{13}$$

where F_{fr} represents the facial recognition network LightCNN-29, which extracts a 256-dimensional facial embedding as the facial features, and $\cos(\cdot,\cdot)$ indicates the cosine distance.

We noticed that if we directly project the composed image to the latent space of StyleGAN2, the generated images usually suffer blurred effects. This is mainly because the latent codes generated by our refinement encoder are not close to the manifold of StyleGAN2. Instead, we opt to learn the residual between the latent code of a composed image and the center of the StyleGAN2 latent codes. In this way, we found that the generated images are more photo-realistic, and this implies that learning the residuals implicitly constrains the generated latent codes to be on the manifold of StyleGAN2.

In summary, the overall objective to train our refinement encoder is defined by:

$$\begin{aligned}\mathcal{L}_{emb} = {} & \lambda_1 * \mathcal{L}_1 + \lambda_2 * \mathcal{L}_{LPIPS} + \lambda_3 * \mathcal{L}_{id} \\ & + \lambda_4 * \mathcal{L}_{ori}(I * M_h, I' * M_h'),\end{aligned} \tag{14}$$

where M_h and M_h' represent the predicted hair masks of the I and I', respectively. In training of the image refinement encoder, λ_1, λ_2, λ_3, and λ_4 are set to $1, 0.8, 1, 1$, respectively. The resolutions of the composed images and the generated images are of size 256×256 pixels and 1024×1024 pixels, respectively.

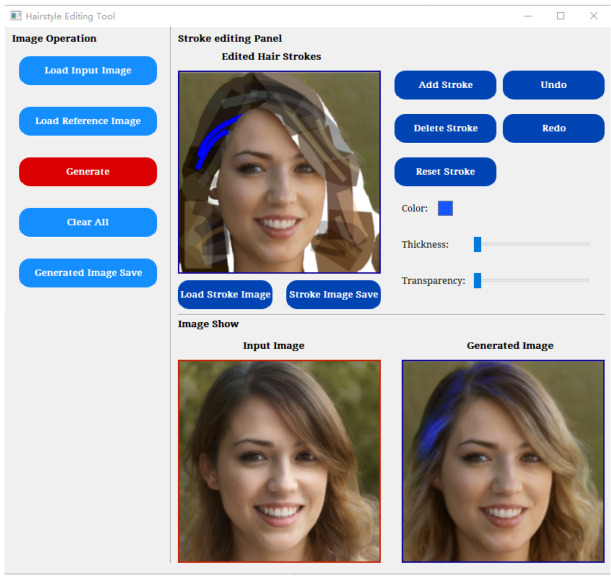


Fig. 4. Our hairstyle editing system. We can manipulate the hairstyles by controlling the strokes parameters, and each stroke is selected by mouse clicking. This figure provides an example of changing the color of a stroke from the original one to blue.

IV. Interactive Hairstyle Editing System

An interactive hairstyle editing tool is provided for users based on the proposed HairstyleNet, as shown in Fig. 4. Convenient editing operations can be achieved in the following ways.

Stroke-Based Hairstyle Editing: Local hairstyle edition is an important function in many multimedia applications. How to enable users to edit local hair flexibly is one of our key concerns. To achieve a convenient operation for users, we provide a stroke image visualization panel that allows users to add, delete and edit a hair strand just by clicking a stroke on the image. Besides, users can edit stroke parameters by adjusting the parameter bars provided in the stroke editing panel. After clicking the generation button, the high-fidelity portrait image with a new hairstyle will be presented in the panel.

Reference Hairstyle Editing: A direct way to manipulate the hairstyle is transferring the hairstyle of a reference image to the input image. In our hairstyle editing tool, we provide a simple operation to achieve this. Users can change hairstyles by clicking the "Load Reference Image" button. After clicking that, the composed image and the generated image will be loaded into the panel.

V. Experiments and Analysis

Qualitative and quantitative experiments are implemented to verify hairstyle editing performance of our proposed method.

A. Dataset and Metrics

Datasets: We employ the large-scale portrait dataset Flickr-Face-HQ (FFHQ) [38] to conduct our experiments. We randomly select 10 K images from FFHQ as the training dataset to train the stroke parameterization network and the refinement encoder, and 5 K images from the remaining images as the test dataset. The resolution of images is resized to 512×512 pixels.

Metrics: We mainly employ three metrics to evaluate the quantitative results produced by different methods. Frecet Inception Distance (FID) measures the visual quality of generated images. Peak Signal-to-Noise Ratio (PSNR) evaluates the image intensity differences between the input image and the generated image. Structural Similarity Index (SSIM) measures the visible structure difference between two images.

Implementation Details: Randomly sampled vectorial parameters and their corresponding strokes generated by the stroke renderer are used to construct the training dataset for the neural renderer. This dataset contains 10 K training pairs. The size of each stroke image is 128×128 pixels. During optimization, the learning rate is set to $2e-4$ and betas to $(0.9, 0.99)$. The learning rate is reduced to its $\frac{1}{10}$ every 100 epochs and stopped after 500 epochs, following [69].

In the stroke parameterization stage, we follow the model architecture and training protocols in [69]. As shown in Fig. 2, we add new hair strokes to a canvas step-by-step. In each step, the parameters of strokes on the $Canvas{-}i$ are updated according to the appearance and structure similarity between the $Canvas{-}i$ and the hair image. Different from [69], we do not divide the canvas into several sub-blocks. Although using sub-blocks facilitates generating detailed hair structure on stroke-based hair images, the multi-scale rendering way breaks the continuity of hair wisps. Another distinct difference is that we introduce a hair orientation loss to maintain the internal structure of the hair.

In the stroke-to-hair generation stage, we first use BiSeNet [72] to segment hair regions from images in FFHQ. Then, the segmented hair images are rendered into hair stroke images by the neural renderer once the neural renderer has been trained. The hair stroke image and the non-hair region from the same image are combined together to form a composed image. In this fashion, the original image and the composed image will be exploited to train the refinement encoder. In this stage, Adam [73] optimizer is used, with β_1 and β_2 set to 0.5 and 0.999, respectively. The learning rate is set to 0.0001 with the batch size of 2. The total iterations of training the refinement network is 50 K. All the experiments are implemented on GeForce RTX 2080Ti. In comparisons, we employ the pre-trained models and codes released by the authors. Our training codes and pre-trained model will be released soon.

B. Quantitative and Qualitative Comparison

Quantitative Comparison: We compare HairstyleNet with the state-of-the-art hair image generation methods: LOHO [9], MichiGAN [8], DeepFace Editing [12] and SketchHairSalon [13]. Because the inpainting module in MichiGAN is not publicly available, we instead employ the face inpainting method LaFin [74] to replace the inpainting module. We employ the released model of SketchHairSalon to generate the comparison results in Fig. 5. Since the matte generation code in SketchHairSalon is not publicly available, we utilize the Gabor filter and sketch simplification method to generate the Salon mattes [75], [76].



Fig. 5. Visual comparison with the state-of-the-art hairstyle transfer methods MichiGAN [8], LOHO [9], DeepFaceEditing [12], and SketchHairSalon [13]. For reference hairstyle editing, our approach achieves better visual results.

TABLE I
QUANTITATIVE COMPARISONS OF DIFFERENT METHODS

Methods	FID↓	PSNR↑	SSIM↑
LOHO [9]	97.82	17.69	0.70
MichiGAN [8]	26.62	27.58	0.88
DeepFaceEditing [12]	123.73	18.53	0.72
SketchHairSalon [13]	185.60	14.31	0.75
HairstyleNet (Ours)	21.68	21.99	0.73

↑ Means the higher, the better. ↓ Means the lower, the better.

Here, our method aims to reconstruct images from stroke-based images. The other competing methods that focus on hairstyle transferring use the input images as reference images but remove the face regions in the reference images to avoid direct copy. In this way, we can compare the quantitative results among different methods. Results are reported in Table I. HairstyleNet achieves superior performance to other methods on most of the metrics. Note that, although MichiGAN obtains the best result on PSNR and SSIM, its visual results suffer obvious artifacts, especially around boundaries between facial and hair regions. This is because MichiGAN simply copies the hair region from a reference image to the target one. Compared with other methods, Hairstyle has the lowest value on the FID, demonstrating its superior ability on achieving holistic appearance consistency after hair editing. In terms of running time, HairstyleNet could provide real-time interactions for users. In the stroke parameterization stage, the stroke generation time depends on the complexity of the hair and the number of strokes. For example, the stroke generation time of the input image with 35 strokes in Fig. 2 costs 2.35 s. Hence, the stroke generation time is acceptable to users. In the stroke-to-hair generation stage, our system spends 0.1 s on generating a portrait with a new hairstyle after users click the generation button.

Qualitative Comparison: In Fig. 5, we demonstrate the visual results of hair transferring from a reference image to a target image. As shown in Fig. 5, when the hair regions of the reference images cannot cover the target hair regions, the competing methods suffer obvious artifacts on the uncovered regions, or fail to transfer the hairstyles of reference images to the target ones. MichiGAN, LOHO, and SketchHairSalon exhibit obvious



Fig. 6. Visual results of hairstyle manipulation. The first and third rows illustrate edited hair strokes, and the second and fourth rows show the edited results.

artifacts near the boundaries between facial and hair regions, while the face regions suffer obvious artifacts in the results of DeepFaceEditing. Note that, the hair structure and shape information in Salon-mattes can be well preserved in generated images. However, the SketchHairSalon only synthesizes the hair region from the input sketch and the hair matte, rather than generating the whole image. Then, the generated hair regions are directly pasted to the input images. This is the main reason for style inconsistencies between the hair regions and the non-hair regions in the generated images.

In contrast, our method transfers the hairstyles effectively and achieves photo-realistic results, where the hair and facial regions are blended seamlessly. In addition, we further illustrate the stroke-based hairstyle editing by manipulating strokes. Since existing methods do not provide such effects, we only demonstrate our results in Fig. 6. More visual results on local hairstyle are shown in Fig. 7.

C. Ablation Study

In the stroke parameterization stage, to verify the importance of each loss function, we generate stroke-based hair images with different loss functions and show the visual results in the left side of Fig. 8. Although employing $\mathcal{L}_1 + \mathcal{L}_{ot}$ can represent the overall hair shapes and colors, it fails to capture the detailed hair structure. Without employing $\mathcal{L}_{ori}$, we fail to produce correct hair wisp orientations, while without using $\mathcal{L}_d$ the width of thin hair wisps cannot be fully captured. Without $\mathcal{L}_{LPIPS}$, the results contain some noisy strokes and miss some correct ones. On the right side of Fig. 8, we plot the MSE between the input hair images and the generated stroke-based hair images according to the various number of strokes. The full losses lead to lower MSE than all other ablated loss functions except $\mathcal{L}_1 + \mathcal{L}_{ot}$. Since $\mathcal{L}_1$+ $\mathcal{L}_{ot}$ focuses on minimizing the pixel color differences, it obtains

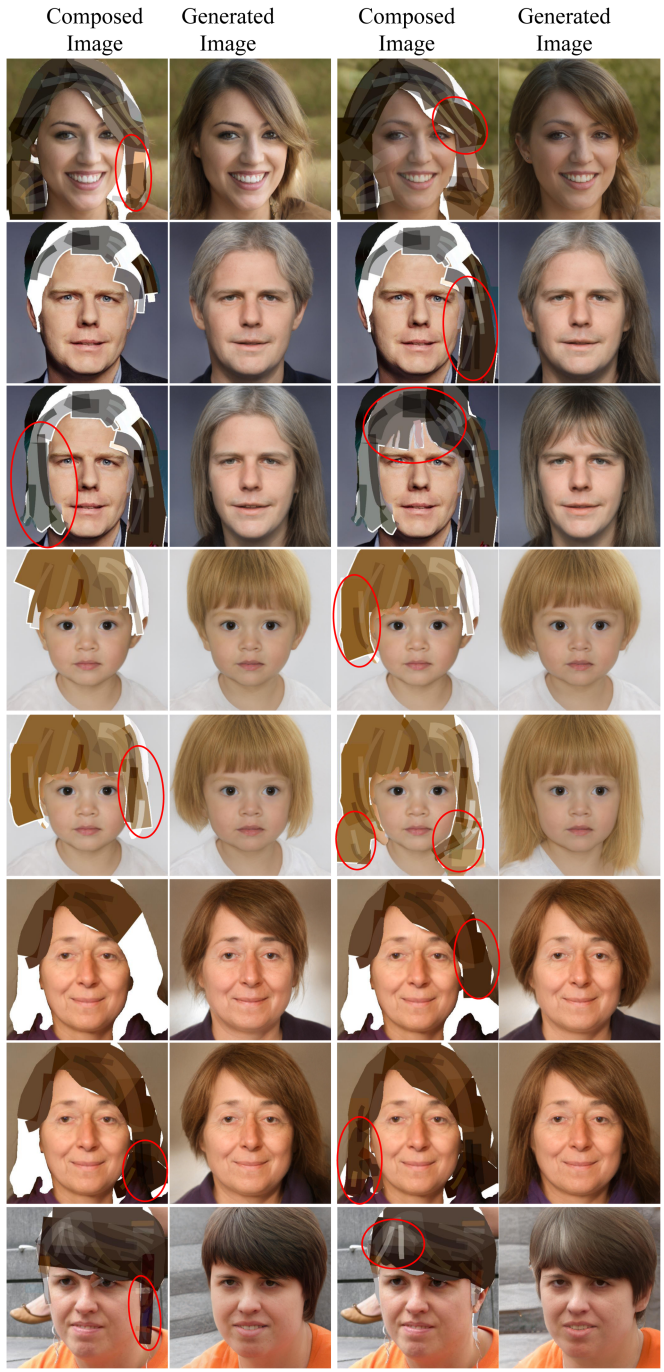


Fig. 7. Visual Results of local hairstyle editing.

the minimum MSE but fails to capture hair wisp structure as expected. The above results validate that $\mathcal{L}_{ori}$, $\mathcal{L}_d$, and $\mathcal{L}_{LPIPS}$ all contribute to the modeling of detailed hair wisp structure generation.

To demonstrate the impact of each loss function on the stroke-to-hair generation stage, we show the generated results trained with different losses in Fig. 9. Note that after incorporating the identity-preserving loss $\mathcal{L}_{id}$, our refinement encoder makes the generated identities very similar to the input portraits. Besides, detailed hair information, especially the direction of hair strands,



Fig. 8. Impacts of different losses on stroke parameterization. Right: visual results of employing different losses. Left: loss values decrease as the number of stroke increases.



Fig. 9. Impacts of different losses on the final visual results in our HairstyleNet. Left: input (top) and reference (bottom) images; Right: results of employing different losses.

TABLE 2
IMPACTS OF DIFFERENT LOSSES ON HAIRSTYLENET

Methods	w/o$\mathcal{L}_{id}$	w/o$\mathcal{L}_{ori}$	w/o$\mathcal{L}_{LPIPS}$	w/o$\mathcal{L}_1$	Ours
FID ↓	27.39	21.51	32.22	32.22	21.68
Cosine ↓	0.29	-	0.16	0.49	0.13

TABLE III
DIFFERENT WAYS OF LATENT CODE REGRESSION

Methods	Directly mapping	Ours
FID	29.87	21.68
Cosine	0.27	0.13

can be preserved with the $\mathcal{L}_{ori}$. Without $\mathcal{L}_1$ or $\mathcal{L}_{LPIPS}$, the results are inconsistent with the reference images in term of hair shape.

We notice that the stroke parameterization in general leads to some information loss, and thus the reconstructed photo-realistic portraits from composed images may not fully comply with original input images but are still visually appealing. Therefore, we argue that the contributions of losses may not truly reflect in terms of the quantitative values on PSNR and SSIM. Similar to the evaluation on generative networks, we employ FID to evaluate the contributions of each loss, as seen in Table II. Without hair orientation loss $\mathcal{L}_{ori}$, the FID improves slightly but it will lead to unwanted hairstyles visually as seen in Fig. 9. Thus, the hair orientation loss is necessary in our method. In order to further show the identity-preserving ability of each loss function, we report the impacts of different losses on the cosine metric in Table II. By employing the identity-preserving and perceptual losses, our results are more realistic, while the impacts of other losses mainly reflect on the visual results. Without $\mathcal{L}_1$, we fail to generate the high-fidelity image because it is difficult to constrain the pixel similarity between the generated images and the input images. As expected, without employing $\mathcal{L}_{id}$, the identity information is not maintained. The visual results in Fig. 9 show that incorporating $\mathcal{L}_{LPIPS}$ can preserve some detailed facial and hair features. Thus, without using $\mathcal{L}_{LPIPS}$, the final results suffer slight performance degradation as indicated in Table II. Although we employ $\mathcal{L}_{ori}$ in the refinement stage, it mainly imposes on the hair regions. Thus, we do not conduct its impacts on identity preservation to avoid confusion.

In the stroke-to-hair generation stage, we generate a high-fidelity image by learning the residual between the ground-truth latent code of a composed image and the center of the StyleGAN2 latent codes rather than directly regressing the ground-truth latent code. To further demonstrate the superiority of our method, we compare our method with directly mapping the composed image to the latent space. As indicated in Table III, our

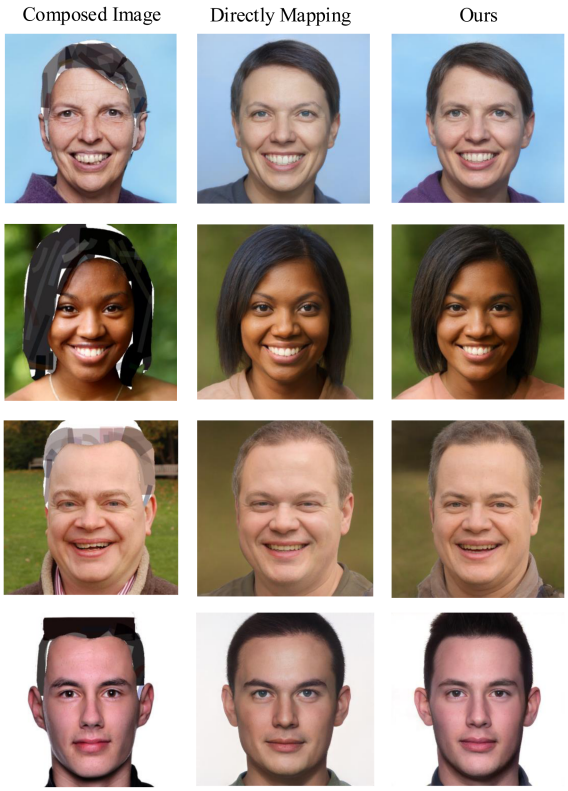


Fig. 10. Visual comparison between direct mapping of styleGAN latent codes and our method.



Fig. 11. Visual results of rendered strokes in the progressive strategy and the parallel strategy. The progressive strategy produces strokes that carry more detailed hair information, including structure, shapes, and colors.

method achieves better identity-preserving results and generates high-quality images. The visual results are shown in Fig. 10.

D. User Study

To further analyze the visual quality of the edited results generated by different methods, we recruit 10 volunteers to participate in the user study. 20 groups of results obtained by LOHO, MichiGAN, DeepFaceEditing, SketchHairSalon, and HairstyleNet are provided to participants. Participants are asked

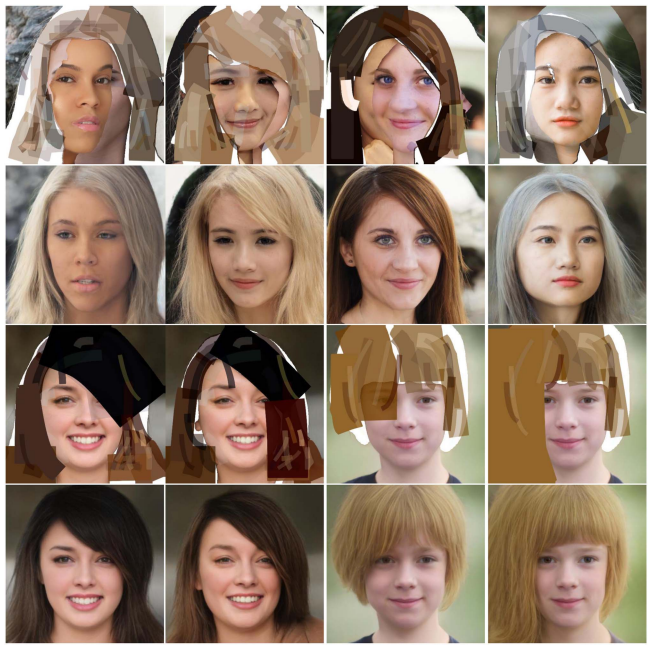


Fig. 12. Visual results of our method under some extreme cases. First Row: Composed images with different face orientations. Second Row: Results generated from images in the first row. Third Row: Composed images that the stroke-based hair covers most of the face region. Fourth Row: Results generated from images in the third row. The above results show that the identity information is well maintained in generated images although the composed images have various face orientations or severe face occlusions.

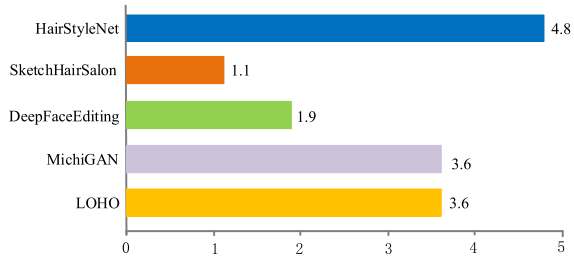


Fig. 13. User study on hairstyle editing methods. The scores for all methods are averaged by 10 volunteers. 5 means the highest quality of the generated image, and 1 represents the lowest quality image.

to rank the generated image quality of each method from 1 to 5 as five methods are compared. 5 indicates the best visual result while 1 represents the worst result. During the user study, the reference and input images are provided to participants for comparison. The final statistical results show that images generated by our HairstyleNet are more favored by participants, with an average rating of 4.8, which surpasses the second best method by an average score of 1.2.

VI. Discussion

Advantages of Strokes. One strength of our framework is generating high-realistic portraits without unnatural boundaries. To support our claim, we show some extreme examples and discuss them in two cases. In the first case, we edit hairstyles of portraits with different face orientations. As seen in Fig. 12, the identity information is well maintained in generated



Fig. 14. Visual results of hair color editing.

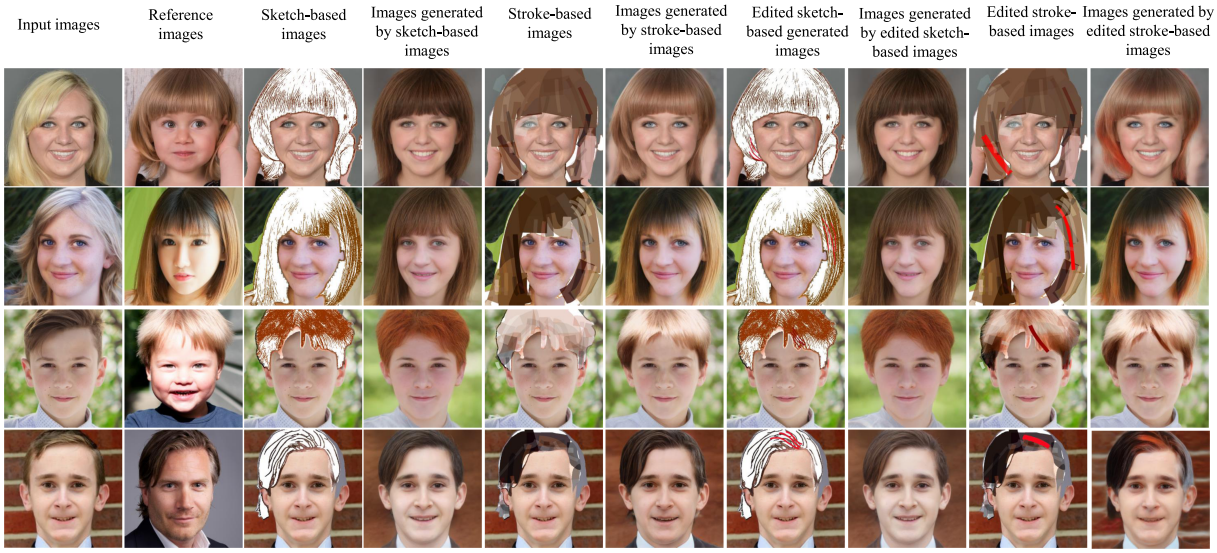


Fig. 15. Visual results of sketch-based and stroke-based hair transferring and editing. Our stroke representation method performs better on local hair editing. Compared with the sketch representation method, strokes are able to preserve the shapes, colors, and structure of local hair, and portraits generated from sketch-based images achieve better visual quality.

images, even though the face orientations are inconsistent in these input images. Another case is that the stroke-based hair covers most of the face regions in the composed images. As shown in the last column of Fig. 12, our framework is able to generate photo-realistic images in the case of severe face occlusions. The above results show that our refinement encoder is able to map the composed images to the latent space of the pre-trained StyleGAN2. In this fashion, the identity information is well preserved in generated images even for these extreme cases.

Another superiority is that the stroke-based hair representation is preferable to other hair representations for local and global hairstyle editing. To demonstrate the superiority of the stroke-based hair representation, we utilize the method in DeepFaceEditing to generate sketch-based images and fine-tune our refinement encoder on the sketch-based images. As indicated in Fig. 15, images generated from sketch-based images fail to preserve the detailed hair structure and colors. The editing results also indicate that sketch-based hair representations are unable to achieve local hairstyle editing. In contrast, portraits generated by the stroke-based hairstyle editing method achieve better visual quality.

Learning Strategy of Strokes. We explore the learning strategy for our renderer network among the progressive learning strategy and the parallel learning strategy. In the stroke parameterization stage, we apply a progressive learning strategy to train our renderer network. As shown in Fig. 2, we add new hair strokes to a canvas step-by-step. In each step, the parameters of strokes on the $Canvas{-}i$ are updated by calculating the appearance and structure similarity between the $Canvas{-}i$ and the hair image. For the parallel learning strategy, we render all strokes in parallel over a number of iterations. However, in this way, the model

would be difficult to converge to the correct strokes. We set 35 strokes to approximate a hair region, and each stroke consists of 11 parameters. Searching for the optimal gradient direction in such a huge parameter space is a formidable task. Experiments are conducted to support our claim. We run gradient descents for 50×35 steps to generate strokes in the progressive strategy. For the parallel updating manner, we run the same number of steps for gradient descents. As shown in Fig. 11, for the parallel updating fashion, the optimizer is struggling to mimic the hair structure, colors, and shapes.

Since optimizing the stroke parameters in parallel tends to produce suboptimal smaller strokes, it converges slower than in the progressive strategy. Thus, within the same iterations, the stroke colors of the parallel strategy fail to approximate the real hair regions, as shown in the third row of of Fig. 11. Furthermore, we conduct experiments on increasing the iteration number of stroke parameters optimization in the parallel strategy. As indicated in Fig. 11, in the parallel updating fashion, when stroke parameters have been updated 100 iterations, the stroke colors become closer to the real hair colors, and when the iteration number reaches 150, stroke colors will resemble the hair colors better. For hairstyle editing tasks, fewer iterations lead to more pleasant interaction experience for users, and thus we opt to our proposed progressive strategy.

Limitations. One disadvantage of our framework is that there exists color bias between the strokes and the generated hair regions, as shown in Fig. 14. In our framework, we train our refinement encoder on composed images and adopt a pre-trained StyleGAN2 as our refinement decoder. In composed images, hair is described by strokes and has differences from hair in real images. Hence, a composed image tends to match its nearest latent code in the latent space of the pre-trained StyleGAN2. In this fashion, the approximately latent code could be decoded into a realistic portrait. This is the reason that there is a hair color deviation between the composed images and real images. Another disadvantage is that there is a color diffusion phenomenon while changing the local hair color, as shown in Fig. 4. The color diffusion phenomenon is induced by the decoding process of the pretrained StyleGAN2 in the stroke-to-hair generation stage. Since the training dataset has fewer images with locally dyed hair, StyleGAN2 tends to generate consistent hair colors to a large extent. In future work, we will explore how to confine colors within the local and global hair regions.

VII. Conclusion

In this paper, we presented a novel stroke controllable hairstyle editing method, dubbed HairstyleNet. To the best of our knowledge, our method is the first attempt to represent input portrait hair regions by controllable parametric strokes in a fully differentiable way. Benefiting from the parametric strokes, we can edit hair by simply changing the parameters of strokes and thus achieve new hairstyle effects. Furthermore, we leverage a pre-trained face generator to constrain the harmony between the edited hair and face regions, thus achieving visually appealing results. Moreover, HairstyleNet can also transfer a hairstyle of a reference image to the input thanks to the design of our controllable parametric strokes. Extensive experiments demonstrate that our proposed HairstyleNet achieves superior hairstyle editing results compared to existing methods and our developed hairstyle editing tool will significantly contribute to many multimedia applications.

References

[1] L. Wei, L. Hu, V. Kim, E. Yumer, and H. Li, "Real-time hair rendering using sequential adversarial networks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 99–116.

[2] M. Chai, J. Ren, and S. Tulyakov, "Neural hair rendering," 2020, *arXiv: 2004.13297*.

[3] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8789–8797.

[4] M. Liu et al., "STGAN: A unified selective transfer network for arbitrary image attribute editing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3673–3682.

[5] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "AttGAN: Facial attribute editing by only changing what you want," *Trans. Image Process.*, vol. 28, no. 11, pp. 5464–5478, 2019.

[6] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "StarGAN v2: Diverse image synthesis for multiple domains," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8188–8197.

[7] Y. Shen, C. Yang, X. Tang, and B. Zhou, "InterfaceGAN: Interpreting the disentangled face representation learned by GANs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 2004–2018, Apr. 2022.

[8] Z. Tan et al., "MichiGAN: Multi-input-conditioned hair image generation for portrait editing," 2020, *arXiv: 2010.16417*.

[9] R. Saha, B. Duke, F. Shkurti, G. W. Taylor, and P. Aarabi, "LOHO: Latent optimization of hairstyles via orthogonalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1984–1993.

[10] P. Zhu, R. Abdal, J. Femiani, and P. Wonka, "Barbershop: GAN-based image compositing using segmentation masks," 2021, *arXiv:2106.01505*.

[11] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[12] S.-Y. Chen et al., "DeepFaceEditing: Deep face generation and editing with disentangled geometry and appearance control," 2021, *arXiv:2105.08935*.

[13] C. Xiao, D. Yu, X. Han, Y. Zheng, and H. Fu, "SketchHairSalon: Deep sketch-based hair image synthesis," *ACM Trans. Graph.*, vol. 40, no. 6, pp. 1–16, 2021.

[14] H.-W. Dong and Y.-H. Yang, "Convolutional generative adversarial networks with binary neurons for polyphonic music generation," 2018, *arXiv:1804.09399*.

[15] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," 2018, *arXiv:1809.11096*.

[16] H. Zhang et al., "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 5907–5915.

[17] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, vol. 29, pp. 469–477, 2016.

[18] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," 2017, *arXiv:1710.10196*.

[19] G. Perarnau, J. Van De Weijer, B. Raducanu, and J. M. Álvarez, "Invertible conditional GANs for image editing," 2016, *arXiv:1611.06355*.

[20] X. Yu, B. Fernando, R. Hartley, and F. Porikli, "Super-resolving very low-resolution face images with supplementary attributes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 908–917.

[21] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7354–7363.

[22] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2223–2232.

[23] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1125–1134.

[24] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1857–1865.

[25] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8798–8807.

[26] I. Anokhin et al., "High-resolution daytime translation without domain labels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7488–7497.

[27] Y.-C. Chen, X. Xu, and J. Jia, "Domain adaptive image-to-image translation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5274–5283.

[28] F. Shiri, X. Yu, F. Porikli, R. Hartley, and P. Koniusz, "Identity-preserving face recovery from stylized portraits," *Int. J. Comput. Vis.*, vol. 127, no. 6, pp. 863–883, 2019.

[29] X. Yu and F. Porikli, "Ultra-resolving face images by discriminative generative networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 318–333.

[30] Y. Guo et al., "Closed-loop matters: Dual regression networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5407–5416.

[31] C. Ledig et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4681–4690.

[32] H. Bin, C. Weihai, W. Xingming, and L. Chun-Liang, "High-quality face image SR using conditional generative adversarial networks," 2017, *arXiv:1707.00737*.

[33] S. Vasu, N. Thekke Madam, and A. Rajagopalan, "Analyzing perception-distortion tradeoff using enhanced perceptual super-resolution network," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 114–131.

[34] X. Wang et al., "ESRGAN: Enhanced super-resolution generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 63–79.

[35] X. Yu and F. Porikli, "Hallucinating very low-resolution unaligned and noisy face images by transformative discriminative autoencoders," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3760–3768.

[36] X. Yu, B. Fernando, B. Ghanem, F. Porikli, and R. Hartley, "Face super-resolution guided by facial component heatmaps," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 217–233.

[37] X. Yu, F. Shiri, B. Ghanem, and F. Porikli, "Can we see more? Joint frontalization and hallucination of unaligned tiny faces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 9, pp. 2148–2164, Sep. 2020.

[38] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4401–4410.

[39] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of styleGAN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8110–8119.

[40] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for styleGAN image manipulation," *Trans. Graph.*, vol. 40, no. 4, pp. 1–14, 2021.

[41] E. Richardson et al., "Encoding in style: A styleGAN encoder for image-to-image translation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2287–2296.

[42] R. Abdal, Y. Qin, and P. Wonka, "Image2StyleGAN: How to embed images into the styleGAN latent space?," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 4432–4441.

[43] R. Abdal, Y. Qin, and P. Wonka, "Image2StyleGAN: How to edit the embedded images?," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8296–8305.

[44] S. Guan, Y. Tai, B. Ni, F. Zhu, F. Huang, and X. Yang, "Collaborative learning for faster styleGAN embedding," 2020, *arXiv:2007.01758*.

[45] A. Pumarola, A. Agudo, A. M. Martinez, A. Sanfeliu, and F. Moreno-Noguer, "GANimation: Anatomically-aware facial animation from a single image," in *Proc. IEEE Conf. Eur. Conf. Comput. Vis.*, 2018, pp. 818–833.

[46] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "MaskGAN: Towards diverse and interactive facial image manipulation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5549–5558.

[47] A. Romero, L. Van Gool, and R. Timofte, "Smile: Semantically-guided multi-attribute image and layout editing," 2020, *arXiv:2010.02315*.

[48] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of GANs for semantic face editing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9243–9252.

[49] H. Chang, J. Lu, F. Yu, and A. Finkelstein, "PairedCycleGAN: Asymmetric style transfer for applying and removing makeup," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 40–48.

[50] S. Zhou, T. Xiao, Y. Yang, D. Feng, Q. He, and W. He, "GeneGAN: Learning object transfiguration and attribute subspace from unpaired data," 2017, *arXiv:1705.04932*.

[51] R. Abdal, P. Zhu, N. J. Mitra, and P. Wonka, "StyleFlow: Attribute-conditioned exploration of styleGAN-generated images using conditional continuous normalizing flows," *Trans. Graph.*, vol. 40, no. 3, pp. 1–21, 2021.

[52] Y. Jo and J. Park, "SC-FEGAN: Face editing generative adversarial network with user's sketch and color," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1745–1753.

[53] S. Yang, Z. Wang, J. Liu, and Z. Guo, "Deep plastic surgery: Robust and controllable image editing with human-drawn sketches," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 601–617.

[54] S. Paris, H. M. Briceño, and F. X. Sillion, "Capture of hair geometry from multiple images," *Trans. Graph.*, vol. 23, no. 3, pp. 712–719, 2004.

[55] M. Zhang, P. Wu, H. Wu, Y. Weng, Y. Zheng, and K. Zhou, "Modeling hair from an RGB-D camera," *Trans. Graph.*, vol. 37, no. 6, pp. 1–10, 2018.

[56] M. Chai, L. Wang, Y. Weng, Y. Yu, B. Guo, and K. Zhou, "Single-view hair modeling for portrait manipulation," *Trans. Graph.*, vol. 31, no. 4, pp. 1–8, 2012.

[57] M. Chai, L. Luo, K. Sunkavalli, N. Carr, S. Hadap, and K. Zhou, "High-quality hair modeling from a single portrait photo," *Trans. Graph.*, vol. 34, no. 6, pp. 1–10, 2015.

[58] L. Hu, C. Ma, L. Luo, and H. Li, "Single-view hair modeling using a hairstyle database," *Trans. Graph.*, vol. 34, no. 4, pp. 1–9, 2015.

[59] Y. Zhou et al., "HairNet: Single-view hair reconstruction using convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 235–251.

[60] L. Luo, H. Li, S. Paris, T. Weise, M. Pauly, and S. Rusinkiewicz, "Multi-view hair capture using orientation fields," in *Proc. IEEE Conf Comput. Vis. Pattern Recognit.*, 2012, pp. 1490–1497.

[61] L. Luo, C. Zhang, Z. Zhang, and S. Rusinkiewicz, "Wide-baseline hair capture using strand-based refinement," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 265–272.

[62] L. Hu, C. Ma, L. Luo, and H. Li, "Robust hair capture using simulated examples," *Trans. Graph.*, vol. 33, no. 4, pp. 1–10, 2014.

[63] L. Hu, C. Ma, L. Luo, L.-Y. Wei, and H. Li, "Capturing braided hairstyles," *Trans. Graph.*, vol. 33, no. 6, 2014.

[64] M. Chai, L. Wang, Y. Weng, X. Jin, and K. Zhou, "Dynamic hair manipulation in images and videos," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 1–8, 2013.

[65] W. Yin, Y. Fu, Y. Ma, Y.-G. Jiang, T. Xiang, and X. Xue, "Learning to generate and edit hairstyles," in *Proc. 25th ACM Int. Conf. Multimedia*, 2017, pp. 1627–1635.

[66] Y. Wu, Y.-L. Yang, and X. Jin, "HairMapper: Removing hair from portraits using GANs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 4227–4236.

[67] T. Wei et al., "HairCLIP: Design your hair by text and reference image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 18 072–18 081.

[68] K. Olszewski et al., "Intuitive, interactive beard and hair synthesis with generative models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7446–7456.

[69] Z. Zou, T. Shi, S. Qiu, Y. Yuan, and Z. Shi, "Stylized neural painting," 2020, *arXiv:2011.08114*.

[70] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.

[71] L. Luo, H. Li, and S. Rusinkiewicz, "Structure-aware hair capture," *Trans. Graph.*, vol. 32, no. 4, pp. 1–12, 2013.

[72] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 325–341.

[73] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[74] Y. Yang, X. Guo, J. Ma, L. Ma, and H. Ling, "LaFIn: Generative landmark guided face inpainting," 2019, *arXiv:1911.11394*.

[75] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to simplify: Fully convolutional networks for rough sketch cleanup," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 1–11, 2016.

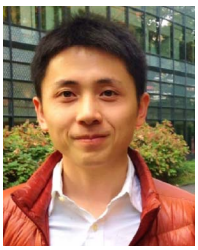
[76] E. Simo-Serra, S. Iizuka, and H. Ishikawa, "Mastering sketching: Adversarial augmentation for structured prediction," *ACM Trans. Graph.*, vol. 37, no. 1, pp. 1–13, 2018.



Xinhui Song received the MS degree in computer science from Zhejiang University, in 2017. She is currently a computer vision researcher with Fuxi AI Lab, Netease. Her research interests include facial expression estimating, deep learning, and hair reconstruct in games.



Chen Liu received the MS degree from the Northwest Agriculture and Forestry University of Computer Science and Technology of China, Shanxi, in 2022. She is currently working toward the PhD degree with the University of Queensland, and works as an intern in Netease.



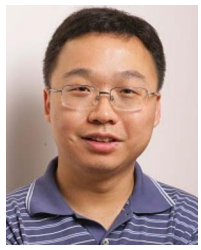
Youyi Zheng received the BS and MS degrees in mathematics from Zhejiang University, China, in 2005 and 2007, respectively, and the PhD degree in computer science from the Hong Kong University of Science & Technology, in 2011. He is a researcher with the State Key Lab of CAD&CG, Zhejiang University. His research interests include geometric modeling, imaging, and human computer interaction. He has served as an associate editor of the *Visual Computer* and *Frontiers of Computer Science*.



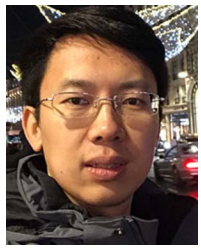
Zunlei Feng received the PhD degree in computer science from Zhejiang University, in 2019. He is currently an assistant research fellow with the College of Software Technology, Zhejiang University, and Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies. His research interests include representation learning, image classification, image segmentation, and pattern recognition.



Lincheng Li received the BS and PhD degrees in electronic engineering from Tsinghua University, Beijing, China, in 2011 and 2017, respectively. He is currently a researcher with Netease Fuxi AI Lab, Hangzhou, China. His research interests include computer vision, pattern recognition, and image and video processing.



Kun Zhou (Fellow, IEEE) received the BS and PhD degrees in computer science from Zhejiang University, in 1997 and 2002, respectively. He is a Cheung Kong professor with the Computer Science Department, Zhejiang University. His research interests include in visual computing, parallel computing, human computer interaction, and virtual reality. He currently serves on the editorial advisory boards of *ACM Transactions on Graphics* and *IEEE Spectrum*.



Xin Yu received the BS degree in electronic engineering from the University of Electronic Science and Technology of China, in 2009, the PhD degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2015, and the PhD degree from the College of Engineering and Computer Science, Australian National University, Australia, in 2019. He is currently a senior lecturer with the University of Queensland. His interests include computer vision and image processing.