GAN-Based Face Attribute Editing

SHUANG LI, DAN LI, TIANCHI CAO, YUKE SUN, YINGSONG HU, AND JUNWEN JI

School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

Corresponding author: Dan Li (lidanhust@hust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61502185, and in part by the Fundamental Research Funds for the Central Universities under Grant 2017KFYXJ071.

ABSTRACT Recently, a variety of methods using the Generative Adversarial Network (GAN) for face editing have been proposed. However, the existing methods cannot control the editing content of the face elements according to the user-specified attributes or need to train a conditional GAN for editing tasks, which means it is difficult to add new attributes in the future. In this paper, a method to edit face attributes by editing the latent variable with the help of a pre-trained unconditional GAN and a linear classification model is proposed. In particular, face attribute editing is divided into two separate stages: Firstly, based on the optimization function, the generative model does a latent variable search to generate a high-quality face image that is similar to the input image. Secondly, by editing the latent variable of the GAN, the attribute of the generated face image can be modified indirectly, so it is almost unaffected by the training process and network structure of GAN, which means it is a flexible method for nearly all GAN network. Images of the FFHQ dataset are edited by attribute labels defined in Celeba dataset for experiments. These experiments prove that our method can edit a variety of face images that vary with race, gender, age, and camera shooting angle. The overall quality of the edited image is not inferior to the other face attribute editing method, and attribute classification for edited image shows a 92.6% attribute editing success rate of the proposed method.

INDEX TERMS Face attribute editing, latent variable search, optimization function, generative adversarial network.

I. INTRODUCTION

Face editing is to edit and modify human face images to make them suitable for personal aesthetics or medical applications. The most widely used application is to post-process the photos of the people, such as opening the eyes of the people in the photos, closing the mouth, etc. In the past, it was professionals to use image processing software such as PhotoShop to perform manual pixel-by-pixel processing, which was a huge workload, whereas the result was not natural. In recent years, with the rapid development of deep learning, the intelligent face editing method based on deep learning has attracted wide attention of researchers [1]–[4]. However, face editing is still a challenging job, because the details of the face vary widely depending on race, gender, age, and even camera shooting angle. In this case, it is even more difficult for face editing adaptively.

Recently, various methods for editing faces using the generative adversarial network have been proposed. For example, Iizuka uses local and global discriminators to judge the realism of the edited image [5], and Yeh uses the pre-trained unconditional GAN to fill the unknown area of an image by searching for latent variable [6]. These methods cannot control the content used to fill the face when editing the face, so they can’t do editing based on user-defined attributes. Although some work can edit the face image according to the attributes specified by the user, there are requirements for the structure and training process of the GAN. The conditional GAN is applied for the methods. For conditional GAN, conditional constraints defined by specialists and expected editing attributes required by the users are bound during the whole training process [7], [8]. However, conditional GAN brings the drawback that it is difficult to add the control of new attributes after training because it is necessary to retrain the GAN with the new attributes.

Based on the idea of [6], a method of face attribute editing that satisfies directionality and global consistency is proposed in this paper. Directionality means that the target face attribute can be controlled to appear or disappear according to the choice of the user, and global consistency means that the extra face elements do not change during editing and the edited area remain consistent with the unedited area. Analogously,
the image can be searched in GAN generative space to match the input image by the defined search optimization function and simultaneously obtain the corresponding latent variable. The difference is that the linear classification model is also applied to get the mapping between the latent variable of GAN and the face attribute, and then the latent variable is interpolated through normal vector movement to obtain the attributed-attached latent variable. Finally, the searched attribute image and the input image can be fused with a user-defined mask to synthesis a face image with the target attribute appearing.

To evaluate the effectiveness of the proposed method, we compared the performance of the latent variable search between directly using the user-defined mask and pre-processing the mask. Then we verify how to choose the optimization function from theory and experiment. We also show that the method of this paper can be applied to face images of different races, ages, genders, camera shooting angles, and compare our edited image quality with the work of Suzuki et al. [8]. For quantitative analysis, an attribute classification network trained by the Celeba facial dataset [9] is used to select images that do not have a specified facial attribute in the FFHQ[10] dataset, and then use this attribute as the target for face attribute editing. The edited images are again classified by the attribute classification network to get the success rate of attribute editing.

In short, we have made the following three contributions in this paper: (1). The image editing method initially used for low resolution (64 × 64 × 3) images is now used for high resolution (1024 × 1024 × 3) images. (2). The previous method has condition constraint in the training process of GAN: Before training the GAN, must define all editing attributes and bind it to some network units (such as the input latent variable) in the training process. The ability to generate high-quality images and the ability to control the attributes of the generated images are coupled, usually modifying one of them means retraining the GAN model. Instead, we assume that there is a well-trained unconditional GAN, which means that for a large number of well-trained GANs, regardless of network structure, after adding some general components, it can generate high-quality images according to the specified attributes without retraining. The ability to generate high-quality images and the ability to control the attributes of the generated images are decoupled and can be optimized separately. (3). We analyze the effectiveness of our method from a qualitative and quantitative perspective on the FFHQ dataset.

The rest of this paper is structured as follows: Section II reviews the work related to image editing. Then we introduce the main ideas of our algorithm in Section III, including how to deal with the user-defined mask, the definition of optimization function, and the attribute editing method. Section IV shows four kinds of experiments: mask experiment, optimization function verification experiment, the quantitative experimental results on the FFHQ dataset, and the qualitative experiment compared to other methods. Finally, Section V summarizes the full paper.

II. RELATED WORK

Image editing is a method of modifying the content of an image to obtain another similar image that meets the requirements of the user. Compared with text editing, image editing has higher complexity and is more difficult. For example, inserting an irrelevant sentence into a fluent statement will affect the semantics, but the paragraph structure will not be damaged. However, in the image editing problem, even if the semantic meanings of the image are not considered, the Poisson fusion [11] variational method is also needed to ensure the consistency of the color before and after the image is pasted. The problem of face attribute editing discussed in this paper requires the use of image completion technology and image attribute editing technology, which requires more in-depth and comprehensive research.

Image Completion is a research hotspot in the field of image editing. In the scene of image completion, an image is considered to be composed of two regions: valid region (contains valid pixel) and invalid region (contains invalid pixel). The goal of image completion is to use the content of the valid region to infer the content of the invalid region. How to use the information of the valid region leads to two different methods: methods based on partial differential equations [12]–[14] and patch-based method [15]–[18]. At the same time, the concept of the valid region can be extended to multiple images, which leads to the use of the image database based method [19], [20]. The disadvantage of this database based method is that the utilization of image data is very inefficient because it does not go deeper to discover the rich information contained by this large amount of data. So deep learning method which also bases on large amounts of data and is capable of mining potential information for large data are naturally applied to image completion [21], [22]. Purely supervised learning can only process small areas like text watermarking [23]. For the complement of an image with larger invalid regions, it is necessary to understand the semantics of the image valid region and generate vivid images, which stimulates the use of GAN [24] for image completion. Iizuka et al. [5] use the content coder (CE, Context Encoder) proposed by Pathak et al. [25], propose a method that contains three networks. Two auxiliary networks guarantee global and local consistency, and one network to complement the image. They get very natural completion results not only on the scene images but also on face images when retrained on a face dataset. However, at the intersection of objects, where the image semantics are ambiguous, such as the overlap of the human head and the distant peaks, it is impossible to obtain satisfactory results. The above deep learning method mainly focuses on constructing a neural network structure. Yeh et al. [6] proposed a completely different idea. They use a generative model that is used to generate images, then through a designed optimization function, optimize the latent variable.
in latent space to find the best generated image for image completion. The advantage of this approach is it divides the completion process into two relatively independent stages: generate images and use the optimization function for latent variable search. Only the first stage requires a well-trained unconditional generative model. Compared to the method that to train a complex model only aimed at image completion, a large number of unconditionally trained generative models are easy to get.

The attribute editing of the face is very similar to the image completion. The attribute editing can be regarded as setting the attribute area to be edited as an invalid area and then applying the completion technique. Different from image completion, it is crucial to control the content of the invalid region. For example, when the mouth part of the person is selected as an invalid region, image completion, which doesn’t care the mouth is open or close after filling, only focuses on filling the invalid region to make the person look real and natural. But attribute editing can provide extra control on what to use to fill the invalid region. There are two different ideas on adding the ability to control the generated image attributes: the first idea is to add conditions to control the generated image attributes when training the network [26], and how to add conditions also has different choices. For example, Odena et al. [27] stitch a conditional vector on the latent variable $z$. Another option is not to add the conditional vector, instead, let the mask be a parameter of the normalization layers of the network [28]–[30]; The second idea is not to add any conditions during training. But once the training is finished, a learning method is used to explore the relationship between the latent variable $z$ and the generated image attributes. The second idea does not bind the attributes to the training process, so it is convenient to add new attributes without retraining the model. At the same time, the learning cost of exploring the latent variable $z$ and image attributes is lower than training a generative model. Some work based on the second idea includes: Engel et al. [31] train a VAE generative model [32], and then train a transformation network to transform the random latent variable into the directed latent variable with specific attributes. Guan uses a simpler general linear model to fit the latent variable and attributes of generated face images [33]. It only takes 5 minutes to fit, and the attribute editing result on the face is excellent. However, as its scope is the entire face, this method cannot edit other attributes while maintaining certain attributes, such as editing the closure of the mouth without changing the eyes and nose of this person.

The problem that this paper focuses on is the attribute editing of the face, which does not change other face attributes while editing a particular face attribute. Similar to the work of Suzuki et al. [8], but they use the idea of adding conditions in the training process, and our proposed method does not bind conditions to the training process. Since PGGAN [34] can only generate images from latent variables, the inverse transformation of images to latent variables cannot be performed as a reversible model like flow model [35]–[37]. So the inverse estimation of GAN [38] is also applied to the method in this paper. By combining Yeh’s optimization function [6] and TLGAN’s attribute editing method [33], the face attribute editing proposed in this paper maintains the advantage of releasing the binding of conditions and generative model training, and at the same time, it can also edit the specific face attributes while the rest face attributes maintain unchanged.

### III. OVERVIEW OF OUR METHOD

We propose a new face attribute editing method, which uses the pre-trained PGGAN model and combines with searching and editing of the latent variable to achieve face image attribute editing. Compared with the most recent face editing methods based on the deep neural network, our method has two advantages: First, based on the pre-trained GAN model, any GAN that can generate high-quality and various face images can be applied to our method, which means that there is no need to re-build new network structures and training from scratch. Second, face attribute editing depends on the interpolation of the latent variable of GAN. It is irrelevant to the training configuration of GAN, for example, conditional or unconditional. Adding control of new attributes does not need to retrain GAN, only a cost of labeling generated images with new attribute tags, which is far below the cost of retraining GAN. Our method pipeline of face attribute editing is shown in Fig. 1. It is mainly divided into two stages: (1) latent variable search. The latent variable $z$ of PGGAN is the variable to be sought, and we define a loss function to measure the similarity of the generated image and the input image. By minimizing this loss, we find $z$ in the latent space, so that the image generated by $z$ through PGGAN has a high similarity to the input image. (2) Edit $z$. Move $z$ in the latent space where the moving direction is determined by a linear classification model which is trained by $<z$ latent variable of PGGAN, face image attribute$> data samples. After moving, the latent variable $z_{attr}$ which could generate a face image with the specified attribute is obtained. Finally, the image generated by $z_{attr}$ and the input image are fused by Poisson-blending on the user-defined area $m_1$ to get a final face image in which the target attribute appears.

The rest of this section is organized as follows: section III.A describes the latent variable search stage in detail, indicating the core idea is to apply the gradient descent method to the optimization function. III.A.1) discusses how to define a meaningful optimization function for GAN trained by Wasserstein distance. III.A.2) shows that doing some pre-processing on the input mask will be beneficial to the latent variable search. III.B focuses on the second stage: latent variable editing, explaining how to edit the latent variable to indirectly achieve the target of attribute editing of the generated image.

### A. LATENT VARIABLE SEARCH

As shown in Fig. 1, the input of our face attribute editing method consists of three parts: (1) the input image $x$, (2) the user-defined editing area $m_1$ (will be
FIGURE 1. Method pipeline. The orange arrows represent the data flow of the first stage, and the gray arrows represent the dataflow of the second stage.

represented by term mask in the rest of the paper), (3), the attribute \( y \) that needs to be added or removed from the input image \( x \). These three inputs divide the image application scene hierarchically. For example, when editing one person in a self-photograph, the target is to only edit the closed eyes into an open state, without changing any other face elements like nose or mouth in the face image, so the mask \( m_1 \) is necessary. At the same time, the purpose of editing is to open the eyes rather than to modify the eyelashes or pupil color, so the target editing attribute \( y \) is also necessary.

The first stage of our method is to find the latent variable \( z \) that corresponds to the input image \( x \), which satisfies

\[
    f^{-1}(x) = z
\]

However, unlike the generative network of the flow model [35]–[37], GAN itself does not provide the capability of inverse transformation. Therefore, it is necessary to use an estimation method to obtain the approximate solution \( z \) of the inverse transformation for the input image. This stage is called latent variable search or inverse transform estimation in this paper.

The idea of using the estimation method to get the approximate solution \( \hat{z} \) of the inverse transform is as follows: First, define a similarity loss \( L(f(z), x) \), the smaller the value, the more similar the generated image \( f(z) \) and input image \( x \) are. The goal is to find an input \( z \) that minimizes \( L(f(z), x) \), so \( L \) is also called optimization function in our paper. Then use an arbitrary random input \( z_0 \), get the generated image \( x_0 = f(z_0) \) through PGGAN, and search for a better \( z \) using the gradient descent method with a learning rate \( \gamma \) as

\[
    z_{i+1} = z_i - \gamma \frac{dL(f(z_i), x)}{dz_i}
\]

The core issue in the latent variable search is how to define the optimization function \( L(f(z), x) \), which will be discussed in detail in Section III.A.1). In short, we use the \( L_1 \) norm and the loss function provided by the adversarial network of PGGAN to build an optimization function, which can achieve excellent inverse transform estimation results, as shown in Fig. 2.

To be noted, it can be seen from Fig. 2 that the image \( \hat{x} \) generated by \( \hat{z} \) cannot reach the exact result of the input image \( x \), and the estimation method cannot obtain a strict
inverse $f^{-1}(x) = z$. For the problem discussed in this paper, it is appropriate and fine to get a similar estimated image, because in the second stage (Section 3.2) for attribute editing (such as adding/eliminating glasses), it will inevitably change the content of the generated image, so that it deviates from the input image under the defined similarity criteria. Therefore, when searching in the latent space, it is not necessary to find a strict inverse transform $f^{-1}(x)$, and it is also impossible for GAN. The use of the estimation method is an alternative idea that GAN cannot be reversed, and it is also a reasonable choice based on the problem discussed in this paper.

1) OPTIMIZATION FUNCTION

In this section, we first derive the optimization function of the latent variable search when the generator network is trained by Wasserstein distance by the idea of previous related research. Then we explain that the Wasserstein GAN must use a different loss for latent variable search. Unlike the traditional GAN has a generator loss term for latent variable search, Wasserstein GAN instead uses a discriminator loss term. This difference leads to a simple but also theoretically effective optimization function for Wasserstein GAN. We will verify the conclusion by evaluating the performance of both two kinds of optimization functions in Section IV.B.

First, make some simple explanations about the symbols used in the rest of the paper. The input image is $x$. The $z$ represents the latent variable, and it is a high-dimensional vector (in PGGAN, it is 512-dimensional), which is obtained by sampling from a multivariate normal distribution. The generator network is noted as $f$, so $f(z)$ represents an image generated by the latent variable $z$. The symbol $m_1$ is a user-defined editing area, in fact, a binary image, and $m_2$ is an actual calculation area obtained by expanding $m_1$. The symbol $D$ is the discriminator network whose input is an image. For the traditional GAN based on the JS divergence, the output of $D$ is a probability value between [0, 1], but for the GAN based on the Wasserstein distance, the output is not a probability and thus can exceed the range of [0,1]. We use $D_W$ to represent the discriminator of Wasserstein distance GAN to distinguish the discriminator $D$ of the traditional GAN.

Pixel loss $L_{sq}$ and realness loss $L_G$ are defined by (2), (3).

$$L_{sq} = \| f(z) - x \|_2$$  \hspace{1cm} (2)

$$L_G = \log (1 - D(f(z)))$$ \hspace{1cm} (3)

The symbol $\odot$ represents element-wise multiplication. $L_{sq}$ computes the $L_1$ norm of $f(z)$ and $x$ on the area defined by $m_2$. $D$ will output a high probability when the image is realistic, so when the image that fused by applying Poisson blending on the area $m_1$ of $f(z)$ and $x$ are more natural, the smaller the $L_G$ is. The optimization function defined by Yeh et al. [6] to execute latent variable search is (4), a weighted combination of (2), (3):

$$\hat{z} = \arg \min_z \alpha L_{sq} + \beta L_G$$ \hspace{1cm} (4)

They use $L_G$ as a part of the optimization function to avoid image smoothing caused by merely utilizing the pixel loss $L_{sq}$. The loss function of PGGAN used in this paper is defined as (5):

$$L_{WD} = E [D_W (f(z)) - D_W (x)],$$

$$L_{WG} = -E (D_W (f(z)))$$ \hspace{1cm} (5)

Inspired by the (4), using the $L_{WG}$ in (5), we can directly write the optimization function of GAN based on Wasserstein distance training as (6):

$$\hat{z} = \arg \min_z \alpha \| f(z) - x \|_2 + \beta L_{WG}$$ \hspace{1cm} (6)

where $\alpha$ and $\beta$ are both positive numbers. However, since the output $D_W$ is no longer in the fixed [0, 1] interval at this time, it is doubted that whether $L_{WG}$ can still be regarded as a quantitative measure of the image realness. We will explain that the change in $L_{WG}$ cannot represent a change in the realness of the image, that is, a decrease in $L_{WG}$ does not mean the image has higher quality. At the same time, we will also get the theoretically effective optimization function for the latent variable search when GAN is trained by Wasserstein distance.

We start our derivation from revisiting how the Wasserstein distance is associated with the training procedure (5). First, beginning with two random variables $x, f$ (same as the definition of previous symbols, $x$ representing the input image, that is, the real image, $f(z)$ representing the generated image, here abbreviated as $f$), the Wasserstein distance of these two random variables is defined as (7):

$$EMD (P_R, P_B) = \inf_{\gamma \in \Pi} \mathbb{E}_{(x,y) \sim \gamma} \| x - f \|_2$$ \hspace{1cm} (7)

Let the probability distribution functions of $x, f$ are $P_R(x), P_B(f)$, and $\gamma$ is the joint probability distribution of these two random variables. The joint distribution $\gamma$ cannot be determined by edge distribution $P_R(x), P_B(f)$, $\gamma$ is unknown. To solve the (7), we first discretize it. We get a discrete form of $\gamma$ and $P_R(x), P_B(f)$ as (8), (9).

$$u = [\gamma(x_1, f_1), \gamma(x_1, f_2), \ldots, \gamma(x_2, f_1), \ldots, \gamma(x_n, f_m)]$$ \hspace{1cm} (8)

$$b = [P_R(x)P_B(f)]^T$$ \hspace{1cm} (9)

The discrete form of (7) becomes (10):

$$\min_{d} d = c^Tu, \quad s.t. A \cdot u = b, u \geq 0$$ \hspace{1cm} (10)

$A$ is a matrix of $(n + m) \times nm$, which is constructed such that each row of the first $n$ rows has only $m$ elements equal to one and other zero, each row of the last $m$ rows has only $n$ elements equal to one and other zero, $c$ is the discrete form of distance as (11).

$$c = [\| x_1 - f_1 \|, \| x_1 - f_2 \|, \ldots, \| x_2 - f_1 \|, \ldots, \| x_n - f_m \|]^T$$ \hspace{1cm} (11)

For the GAN using Wasserstein as the loss, it does not usually directly use (10) but its dual-type as (12):

$$\max \hat{d} = b^Tv, \quad s.t. A^Tv \leq c$$ \hspace{1cm} (12)
It can be proven that the maximum $\hat{d}$ found in (12) is the minimum $d$ of (10) using Kantorovich-Rubinstein duality. The unknown $v$ in (12) has only $n + m$ parameters, and the unknown $u$ in Equation (10) has $nm$ parameters, so (12) with fewer parameters to be determined is more suitable for neural network training than (10).

Recall that $b$ is a column vector that consists of the probability of a real image and a false image defined in (9), so $b^T v$ represents an expectation. Comparing it to $L_{WD}$ in (5) which is also an expectation, we can get an equation about $v$ and $D_w$ as (13):

$$v = [D_W(x)|-D_W(f(z))]^T$$  \hspace{1cm} (13)

The $v$ in the left-hand side is from the Wasserstein distance, and the $D_w$ in the right hand is from the WGAN loss. The $|$ is a separator, the same as the separator symbol in the partitioned matrix, indicating the very high dimensional vector $v$ grouped by two parts. Therefore, we successfully associate the training procedure of WGAN to the Wasserstein distance. Equation (13) tells us that $v$ is a large vector consist of the output value $D_w$ of all real images $x$ and false images $f(z)$. The training process of $D_w$ in (5) corresponds to change the value of $v$ to find the optimal $\hat{d}$ in (12).

When (12) is optimal, the condition $A^T v \leq c$ can get the equal sign, that is $A^T v = c$, expand $A^T v$ and use (11) to replace $c$, then there will exist (14):

$$v_i + v_j = c_{ij} = \| x_i - f_j \|_2$$  \hspace{1cm} (14)

We could continue use (13) to replace $v$, and finally get (15):

$$v_i + v_j = D_W(x_i) - D_W(f_j) = \| x_i - f_j \|_2$$  \hspace{1cm} (15)

Equation (15) illustrates two facts: Firstly, any two components of $v$ added are equal to $L_2$ norm of the real image and the generated image. Secondly, it is meaningless to use $D_w(f)$ or $D_w(x)$ alone. So we give up the optimization function (6) that uses $L_{WG}$, because equation (15) says a single value like $D_w(f)$ is meaningless, and only combining $D_w(f)$ and $D_w(x)$ like $L_{WD}$ makes sense. Therefore, we redefine an optimization function using $L_{WD}$ as (16):

$$L = \alpha \| (f(z) - x) \odot m_2 \|_1 - \beta L_{WD}$$
$$= \alpha \| (f(z) - x) \odot m_2 \|_1 - \beta (D_W(f(z)) - D_W(x))$$
$$= \alpha \| (f(z) - x) \odot m_2 \|_1 + \beta \| x - f(z) \|_2$$  \hspace{1cm} (16)

The first equal sign uses $-L_{WD}$ to replace $L_{WG}$ in (6). Then the $L_{WD}$ is expanded by (5), and the expectation can be omitted as there is only one real image and false image. The last equal sign use (15) to replace $D_w(f)$ and $D_w(x)$. We use $-L_{WD}$ rather than $L_{WD}$ is because in training process, $L_{WD}$ is used for discriminator network to distinguish real and false image, but the target of optimization function is to make the generated image $f(z)$ as similar as possible to the input image $x$. So there is a negative sign in front of $L_{WD}$. The latent variable search stage is to use gradient descent method (1) to find the minimal $\hat{z}$ as (17):

$$\hat{z} = \arg \min L$$  \hspace{1cm} (17)

2) MASK PRE-PROCESSING

There is another noteworthy question when computing the optimization function: how to choose the area of the generated image $f(z)$ and the input image $x$ to compute the similarity? In Fig. 1, we use the area indicated by the mask $m_2$ to compute the similarity. Although the mask $m_1$ indicates the attribute editing area defined by the user, we find that a larger computing area is beneficial to measure the similarity and therefore contributes to latent variable search. The above findings can be explained simply by the fact that when a larger mask $m_2$ is used, the edge of the original user-defined mask $m_1$ is also included, which makes the optimization function also evaluate the edge consistency of the generated image $f(z)$ and the input image $x$. Therefore, after fusion, the editing area of the fusion image will be more consistent with the area outside the editing area.

Another possible way to define $m_2$ is to select several feature areas of the face, such as eyes, nose, mouth, etc., which will make $f(z)$ iterate faster to the image close to $x$. However, this will create new problems. For different face images, it is necessary to be able to detect the position of these facial feature areas, so a face detector has to be added. Adding a face detector will make the latent variable search more complicated. The pre-processing method of the mask can be further studied, but it is not the focus of this paper.

Using simple morphological dilation as a pre-processing can get good results, so we use it directly to get the balance of estimation performance and computation efficiency.

B. ATTRIBUTE EDITING

After obtaining the inverse transform estimation $\hat{z}$ of the input image, the next stage is editing $\hat{z}$ to convert it into a latent variable with the needed attribute $z_{attr}$. The transform from $\hat{z}$ to $z_{attr}$ requires us to find a mapping between the latent variable $z$ and the attribute $y$ of the generated image.

The traditional GAN cannot control the mapping of the latent variable $z$ and the attribute $y$ of the generated image. One way to obtain this extra control ability is to add a condition $C$ to the latent variable and manually specify its mapping $y$ when training the GAN. Such a GAN is called a conditional GAN. The biggest problem with this method is that, after the training finished, if you want to add control about the new attribute $y_{new}$, in most cases, you can only retrain this conditional GAN to add a new mapping of condition $C$ and image attribute $y$.

Another method for finding the mapping between the latent variable $z$ and the image attribute $y$ is to use the auxiliary classification network and the linear classification model to get the $(z,y)$ mapping after the GAN training is completed, which is the method used in this paper. As shown in Fig. 3, the generated image data set $Y$ is first obtained by using
PGGAN in a large number of random samples in the latent space. Then, relying on the auxiliary image classification network to mark the generated image \( f(z) \) with the attribute label \( y \), we get a labeled image data set \( Y \). Finally, a pair set \( \{f(Z), Y\} \) of corresponding \( z \) and \( y \) is entirely constructed and will be used to train the linear classification model to fit the mapping relationship between the latent variable \( z \) and face attribute label \( y \).

We will explain how to use the trained linear classification model to edit \( \hat{z} \) and convert it to a variable with the needed attribute \( z_{\text{attr}} \) in the following simple case. Suppose \( z \) is a two-dimensional vector rather than a 512-dimensional vector, and \( y \) is the attribute of "glasses". After linear classification, we obtain the linear classification boundary as shown in Fig. 4.

We can get a line \( az_1 + bz_2 + c = 0 \) which is the classification boundary of latent space point(variable) \( z \). This line divides the points with the "glasses" attribute to side \( az_1 + bz_2 + c > 0 \), and the points with the "no glasses" attribute to side \( az_1 + bz_2 + c < 0 \). Therefore, when we want to add the "glasses" attribute to a point \( P \) which has "no glasses" attribute, meaning it lies in the side \( az_1 + bz_2 + c < 0 \) initially, what we do is to move it along the normal vector \( \vec{n} \) direction of the classification boundary line. After this translate transformation, the point \( P' \) will be located on the "glasses" side of the classification boundary, meaning it will then have the "glasses" attribute. In the case where \( z \) is a higher dimensional vector, the classification boundary will be a hyperplane, and the above conclusion is also correct. That is, the point \( z \) on one side of the hyperplane is moved along the direction of the plane normal vector to reach the other side of the hyperplane, which means an attribute point \( z_{\text{attr}} \) with the shortest distance from \( z \). The Celeba dataset has a total of 40 attributes, so the training for the linear classification model will get 40 normal vectors. When editing the \( k \)th attribute for a face image \( x \), get inverse transform estimation \( \hat{z} \) of \( x \) by latent variable search, and then add the corresponding \( k \)th hyperplane normal vector as (18) to get the edited latent variable \( z_{\text{attr}} \) with the specified attribute \( y \).

\[
    z_{\text{attr}} = \hat{z} + \beta \vec{n}_k
\]

**IV. EXPERIMENTS AND ANALYSIS**

To evaluate the effect of the method proposed in this paper, we conduct a series of experiments. The hardware and software environment of experiments are, Intel CPU i7-7700HQ 2.8GHz, 8GB RAM, NVIDIA 1060 6GB, 64-bit Windows operating system. The pre-trained PGGAN model provided...
by Karras et al. [34] is used to generate natural human face images, and the linear classification model is from the work of Guan [33]. When evaluating attribute editing accuracy, the face attribute classifier used comes from [39].

We divided the experiments into four groups: The first group is the verification experiment for the effectiveness of mask pre-processing. The second group compares the performance of using \( L_{WD} \) and \( L_{WG} \) as the optimization function. The third group qualitatively analyzes our method for editing face attributes. The last group uses the face attribute classification network to analyze the success rate of face attribute editing quantitatively.

**A. MASK PRE-PROCESSING**

The user-defined mask describes the area in which the face attribute needs to be edited. A natural choice is to use the user-defined mask directly as a mask for the optimization function to do the latent variable search. However, our experiments find that a larger mask often yields better results, as shown in Fig. 5. The test image is taken from the Suzuki et al. [8] and rescaled to 1024 × 1024 size.

The first row in Fig. 5 is the editing result by using a user-defined mask. The second row represents the edit results obtained by using a larger mask that contains the user-defined mask \( m_1 \) directly (second column). When using the mask \( m_1 \), the inverted face that the optimization function finds for the latent variable search is inconsistent with the input on camera shooting direction, but the face found using a larger mask \( m_2 \) is consistent with the input (third column). As we explained in the previous section, a larger mask gives the optimization function more information about the edit area with surround face components (in this case, the orientation of the face), making the searched image more consistent with the input.

Although using a larger mask is better for latent variable search, what is the best size for the mask needs some more discussion. To find the best mask size, we define a unit loss according to (16) as (19):

\[
\text{unit loss of each pixel} = \frac{L}{N} \tag{19}
\]

\( N \) is the total number of mask pixels and \( L \) could be calculated by (16). When a larger mask is used, the value of \( L \) should be larger than a small mask, so to objectively evaluate the effect of masks of different sizes on minimizing the optimization function \( L \), let \( L \) be divided by the mask pixels \( N \) to get the unit loss of each pixel. As shown in Fig. 6, we plot the trend of unit loss of each pixel with the number of latent variable search iterations in the case of different mask sizes.

The curves of different colors represent the scale factor \( s \) of the mask compared to the original user-defined mask, that is, area of \( m_2 = s \times \text{area of } m_1 \). Fig. 6 illustrates when \( s = 1.2 \), the trend of the loss curve is the most stable, and the unit loss of each pixel after 25 iterations is also the smallest. Although this empirical conclusion comes from the relative size of the mask, we believe it is also a general conclusion. Because when the aligned face images are scaled to the same size, the absolute sizes of the same facial element on different face images are usually close. Under the assumption that the user-defined mask contains the area to be edited, it is more convenient to use the relative size. Depending on user-defined masks reduces the generality of this method, but the automatic selection of masks requires more discussion and effort, which is beyond the scope of this paper, so we make a simple but not complicated assumption about user-defined masks.

**B. OPTIMIZATION FUNCTION**

According to the analysis, in our method, unlike the traditional GAN trained by minimax loss uses the \( L_G \) term as part of the optimization function for latent variable search, the GAN trained by Wasserstein distance should use the \( L_{WD} \) term to form the optimization function instead of the \( L_{WG} \) term. In this section, we will compare the \( L_{WD} \) item and the \( L_{WG} \) item separately as the optimization function to prove the correctness of our analysis of (6) and (17).
FIGURE 7. Latent variable \( z \) search using different optimization functions: first row: \( L_{WG} \). The second row: \( L_{WD} \). The \( L_{WD} \) term approximates the mean square error, which directly compares the pixel difference between the image generated by \( z \) and the input image, showing the image changes significantly during the iteration (male beard, mouth, etc.). The \( L_{WG} \) item makes iterated variable \( z \) keep generate realistic images, but the change of image is relatively small (female eye and eyebrow, the color of the hair in the branch).

FIGURE 8. Images generated by PGGAN using latent variable \( z \) sampled from the different probability distribution.

Firstly, use \( L_{WD} \) and \( L_{WG} \) as optimization functions to perform the latent variable search. The image generated by \( z \) during the search process is shown in Fig. 7. We only pay attention to whether the image is realistic. Iteratively generated images maintain realistic regardless of which of the two optimization functions is used. In this case, it cannot be said that both \( L_{WG} \) and \( L_{WD} \) work well for latent variable search, because the outstanding result is relay on that the PGGAN is robust to the input \( z \). As shown in Fig. 8, even if \( z \) is outside of the defined latent space, PGGAN can also generate real face images.

The input latent variable of the PGGAN generator network is a standard normal distribution. By simply using the 3\( \sigma \) principle, it can be seen that the range of all components of \( z \) should between \([-3, 3]\). In the case of using the mean distribution of the offset, the non-standard Gaussian distribution, and the mixed Gaussian mean distribution. The latent variable \( z \) will lie in intervals different from the standard Gaussian distribution, but the generated images remain realistic. This result not only shows that PGGAN’s generative network is highly robust, which can still generate high-quality images when \( z \) outside the latent space, but also leads to the fact that even if we obtain high-quality images by using \( L_{WG} \) and \( L_{WD} \) terms, it may be because PGGAN itself is robust to input \( z \) but not the effectiveness of \( L_{WG} \) and \( L_{WD} \).

In order to correctly evaluate which \( L_{WG} \) and \( L_{WD} \) can guarantee to generate a realistic image, choose not to optimize \( z \), but directly optimize the input noise image \( x_{\text{noise}} \), thus avoiding the influence of PGGAN robustness for the latent variable \( z \). The generated images during iterations are shown in Fig. 9. The optimization function \( L_{WD} \) gets results close to the first experiment in this section, because in (16), in the ideal case, \( L_{WD} \) is equivalent to the mean square error, so the optimized value of the latent variable \( z \) and random noise image \( x_{\text{noise}} \) are similar. For \( L_{WG} \), although the noise image can be gradually transitioned from an invisible noise image to an image with a somewhat face contour, the color of the face, the surrounding background, and the details are far from the standard of the real image.

The three experiments in this section show that \( L_{WD} \) is more suitable for making generated image realistic than \( L_{WG} \), and prove the validity of (16) using \( L_{WD} \) to construct an optimization function of latent variable search.

C. QUALITATIVE ANALYSIS

1) VARIOUS ATTRIBUTE EDITING

This group of experiments shows that the method of this paper can be applied to edit various face attributes. As shown in Fig. 10, the user-defined image editing area does not need to be a regular area, and it can be any shape, so that the editing area mask used in the first image is a simple rectangle, and the rest all use irregular editing area masks. All test images come from the FFHQ dataset [10].
FIGURE 9. Use $L_{WG}$ and $L_{WD}$ to directly optimize the random input noise image: first row, $L_{WG}$; second row, $L_{WD}$.

FIGURE 10. Edit four different attributes (mouth slightly open, mustache, glasses, big nose) of one person from the FFHQ data set.

We edit four different face elements of the people, including mouth, beard, eyes, and nose. For the mouth, after defining the black rectangular area of the image as the editing area and giving Celeba’s “mouth slightly open” attribute, the image of the person with the mouth more open and exposing the tooth is obtained. Such a simple mouth attribute edit already has a significant meaning in the application, because the smile of the person is often closed to the degree of the mouth opening. These edited face images are very realistic, and the selected editing area not only has been changed obviously according to the target attribute but also keeps consistent and semantic with the surrounding area.

2) VARIOUS FACE ATTRIBUTE EDITING
As shown in Fig. 11, this group of experiments shows that the method of our paper can be applied to edit different face images that vary with race, gender, age, and camera shooting angle. At the top left, we use the “mouth slightly open” attribute to edit the male and female images, either the front female or the male facing the side of the camera, the lips changing from closed to open. At the bottom left, the target “mouth closed” attribute is also edited successfully for images of different genders and races. In the upper right, the male-only “goatee” attribute is edited, and the result of clearing the beard is obtained. At the lower right, the “narrow eye” attribute is edited, and two opposite operations (let the eye bigger or let the eye smaller) are performed, successfully opening the eyes of the originally narrow-eyed woman and editing the woman with the original big eyes into a small eye.

3) COMPARED WITH OTHER METHOD
We then compare our method with the method proposed by Suzuki et al. The results are shown in Fig. 12.
The qualitative comparison with the two methods shows that requiring a mask is the main deficiency of our method: This does add some manual work. Although most of our masks can be replaced by simple circles or rectangles, while for some complicated structures, such as the case of glasses, a more ingenious mask may be required. However, our method has no constraints on training and can easily add new attributes to edit after the training of GAN finished, which has greater advantages in flexibility. In the future, our main research goal will focus on how to make the mask selection step intelligent and automated.

D. QUANTITATIVE ANALYSIS

To quantitatively and objectively analyze the effect of our face attribute editing method, a face attribute classification network mxnet-face[39] is used as an evaluation tool. This classification network achieved an attribute classification accuracy of 87.41% on the Celeba dataset. This accuracy is calculated by counting the number of images that the 40-dimensional prediction vector is the same as the label vector. We use part of the FFHQ dataset as a test set, firstly artificially select images with five types of attributes: “mouth close”, “mouth slightly open”, “narrow eye”, “big eye”, “no beard” as candidate test images. The “mouth close” and “big eye” are the opposite of the original “mouth slightly open” and “narrow eye” attributes in the Celeba dataset. Above candidate test images selected by the human are then attribute classified by mxnet-face, and the images that match both artificial selection and mxnet-face classification are taken as the final test images. The step of adding artificial selection is to eliminate the effect of the 12.59% misclassification rate of mxnet-face. We use our method to perform five types of face attribute editing operations: “mouth close” → “mouth slightly open”, “mouth slightly open” → “mouth close”, “narrow eye” → “big eye”, “big eye” → “narrow eye”, “no beard” → “beard”. The mxnet-face network is
TABLE 1. Statistics for five attributes editing of partial FFHQ images.

| original attribute | target attribute | success editing num | total editing num | success rate |
|--------------------|------------------|---------------------|------------------|-------------|
| mouth closed       | mouth slightly open | 15                  | 15                       | 100%        |
| mouth slightly open| mouth closed      | 19                  | 20                       | 95%         |
| narrow eye         | big eye           | 19                  | 20                       | 95%         |
| big eye            | narrow eye        | 15                  | 20                       | 75%         |
| no beard           | beard             | 20                  | 20                       | 100%        |

| FIGURE 14. Comparison of successfully editing images of five attributes and the total editing success rate. |
|--------------------|------------------|---------------------|------------------|-------------|
| FIGURE 15. Image before “narrow eye” attribute editing (left), image after “narrow eye” attribute editing (right). |

used again to label the attributes of result images and count the number of images in which the target attribute appears and other attributes keep the same as before. This data can objectively reflect the success rate of our face attribute editing method. The statistical results are shown in Table 1. The graphical comparison of these five attributes is shown in Fig. 14.

From Table 1, Fig. 14, we find that the attribute editing method has a very high editing success rate for four types of attributes, but the attribute editing of “big eye”→ “narrow eye” is inferior to the other four attributes. Fig. 15 is a failed case of this attribute.

Surprisingly, our method edits the big eye along the direction to the narrow eye. However, the evaluation tool mxnet-face does not assign the “narrow eye” label to the edited image. We think the reason is that the mxnet-face used for evaluating the success rate of face attribute editing and the auxiliary classification network used for preparing train data of linear classification model is inconsistent with the criteria for judging “narrow eye” attribute. Unlike the two opposite attributes of “mouth slightly open” and “mouth closed”, “narrow eye” and “big eye” are all states of the opened eye, the main difference is the degree of the opening so that it will be more difficult for classification. This experiment shows that mxnet-face tends to label a part of the test faces that will be labeled as “narrow eye” by the auxiliary classification network, as “big eye”, which means the mxnet-face is more strict with “narrow eye” attribute. However, it is apparent in the intuitive feeling that the edited image moves toward the “narrow eye” attribute as shown in Fig 13.

V. CONCLUSION

In this paper, we propose a method for face attribute editing based on pre-trained unconditional GAN. The ability to generate high-quality images is decoupled from the ability to control the attribute of the generated face. The experiment results demonstrate that our method can obtain the state of art result. The advantages of the proposed method are that there are no constraints on training and new attributes can be easily added even after the training of GAN finished, which makes our method more flexible and potential. The main deficiency of our method is that the mask selection is manual. Therefore, in future work, our main research goal will focus on how to make the mask selection step intelligent and automated. After that, we are going to optimize each stage of our method pipeline, such as using styleGAN for better image quality and using other models to learn the mapping between latent variables and face attributes, etc.

REFERENCES

[1] B. Dolhansky and C. C. Ferrer, “Eye in-painting with exemplar generative adversarial networks,” in Proc. CVPR, Salt Lake City, UT, USA, Jun. 2018, pp. 7902–7911.
[2] T. Portenier, Q. Hu, and A. Szabó, “Faceshop: Deep sketch-based face image editing,” ACM Trans. Graph., vol. 37, no. 4, Aug. 2018, Art. no. 99.
[3] L. Song, J. Cao, L. Song, Y. Hu, and R. He, “Geometry-aware face completion and editing,” 2018, arXiv:1809.02967. [Online]. Available: http://arxiv.org/abs/1809.02967
[4] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi, “EdgeConnect: Generative image inpainting with adversarial edge learning,” 2019, arXiv:1901.00212. [Online]. Available: http://arxiv.org/abs/1901.00212
[5] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Globally and locally consistent image completion,” ACM Trans. Graph., vol. 36, no. 4, pp. 1–14, Jul. 2017.
[6] R. A. Yeh, C. Chen, T. Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, “Semantic image inpainting with deep generative models,” in Proc. CVPR, Honolulu, HI, USA, Jul. 2017, pp. 5485–5493.
[7] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, “AttGAN: Facial attribute editing by only changing what you want,” IEEE Trans. Image Process., vol. 28, no. 11, pp. 5464–5478, Nov. 2019, doi: 10.1109/TIP.2019.2961751.
[8] R. Suzuki, M. Koyama, T. Miyato, T. Yonetsuji, and H. Zhu, “Spatially controllable image synthesis with internal representation collaging,” 2018, arXiv:1811.10153. [Online]. Available: http://arxiv.org/abs/1811.10153
[9] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in Proc. IEEE Int. Conf. Comput. Vls. (ICCV), Dec. 2015, pp. 3730–3738.
[10] T. Karras, S. Karras, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in Proc. CVPR, Long Beach, CA, USA, Jun. 2019, pp. 4401–4410.

[11] T. Karras, S. Karras, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in Proc. CVPR, Long Beach, CA, USA, Jun. 2019, pp. 4401–4410.

[12] M. Bertalmio, G. Sapiro, and V. Caselles, “Image inpainting,” in Proc. 27th Ann. Conf. Comput. Graph. Interact. Techn., 2000, pp. 417–424.

[13] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, “Filling-in by joint interpolation of vector fields and gray levels,” IEEE Trans. Image Process., vol. 10, no. 8, pp. 1200–1211, Aug. 2001.

[14] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, “Simultaneous structure and texture image inpainting,” IEEE Trans. Image Process., vol. 12, no. 8, pp. 882–889, Aug. 2003.

[15] A. Criminisi, P. Perez, and K. Toyama, “Region filling and object removal by exemplar-based image inpainting,” IEEE Trans. Image Process., vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[16] Y. Wexler, E. Shechtman, and M. Irani, “Space-time completion of video,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 29, no. 3, pp. 463–476, Mar. 2007.

[17] C. Barnes, E. Shechtman, and A. Finkelstein, “PatchMatch: A randomized correspondence algorithm for structural image editing,” ACM Trans. Graph., vol. 28, no. 3, Aug. 2009, Art. no. 24.

[18] X. He and J. Sun, “Statistics of patch offsets for image completion,” in Proc. ECCV, Tuscany, Italy, 2012, pp. 16–29.

[19] J. Hays and A. A. Efros, “Scene completion using millions of photographs,” Commun. ACM, vol. 51, no. 10, p. 87, Oct. 2008.

[20] O. Whyte, J. Sivic, and A. Zisserman, “Get out of my picture!” Internet-based inpainting,” in Proc. Brit. Mach. Vis. Conf., London, U.K., 2009, p. 5.

[21] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep residual networks,” in Proc. NIPS, Stateline, NV, USA, 2012, pp. 341–349.

[22] R. Köhler, C. Schuler, and B. Schölkopf, “Mask-specific inpainting with deep neural networks,” in Proc. GCPR, Münster, Germany, 2014, pp. 523–534.

[23] J. S. J. Ren, L. Xu, and Q. Yan, “Shepard convolutional neural networks,” in Proc. NIPS, Montreal, QC, Canada, 2015, pp. 901–909.

[24] I. Goodfellow, J. Pouget-Abadie, and M. Mirza, “Generative adversarial nets,” in Proc. NIPS, Montreal, QC, Canada, 2014, pp. 2672–2680.

[25] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, “Context encoders: Feature learning by inpainting,” in Proc. CVPR, Las Vegas, NV, USA, Jun. 2016, pp. 2536–2544.

[26] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” 2014, arXiv:1411.1784. [Online]. Available: http://arxiv.org/abs/1411.1784

[27] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier GANs,” in Proc. ICML, Sydney, NSW, Australia, 2017, pp. 2642–2651.

[28] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” 2019, arXiv:1903.07291. [Online]. Available: http://arxiv.org/abs/1903.07291

[29] H. De Vries, F. Strub, and J. Mary, “Modulating early visual processing by language,” in Proc. NIPS, Long Beach, CA, USA, 2017, pp. 6594–6604.

[30] V. Dumoulin, J. Shlens, and M. Kudlur, “A learned representation for artistic style,” 2016, arXiv:1610.07629. [Online]. Available: http://arxiv.org/abs/1610.07629

[31] J. Engel, M. Hoffman, and A. Roberts, “Latent constraints: Learning to generate conditionally from unconditional generative models,” 2017, arXiv:1711.05772. [Online]. Available: http://arxiv.org/abs/1711.05772

[32] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” 2013, arXiv:1312.6114. [Online]. Available: http://arxiv.org/abs/1312.6114

[33] S. Guan. TL-GAN: Transparent Latent-Space GAN. Accessed: Mar. 4, 2019. [Online]. Available: https://github.com/SummitKwan/transparent_latent_gan

[34] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” 2017, arXiv:1710.10196. [Online]. Available: http://arxiv.org/abs/1710.10196

[35] L. Dinh, D. Krueger, and Y. Bengio, “NICE: Non-linear independent components estimation,” 2014, arXiv:1410.8516. [Online]. Available: http://arxiv.org/abs/1410.8516

[36] L. Dinh, J. Sohn-Dickstein, and S. Bengio, “Density estimation using Real NVP,” 2016, arXiv:1605.08803. [Online]. Available: http://arxiv.org/abs/1605.08803

[37] D. P. Kingma and P. Dhariwal, “Glow: Generative flow with invertible 1x1 convolutions,” in Proc. NIPS, Montreal, QC, Canada, 2018, pp. 10215–10224.

[38] Z. C. Lipton and S. Tripathi, “Precise recovery of latent vectors from generative adversarial networks,” 2017, arXiv:1702.04782. [Online]. Available: http://arxiv.org/abs/1702.04782

[39] W. Wu. Using MxNet for Face-Related Algorithm. Accessed: May 13, 2019. [Online]. Available: https://github.com/tornadomeet/mxnet-face

SHUANG LIU received the bachelor’s degree from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2018, where he is currently pursuing the master’s degree in computer science and technology with the School of Computer Science and Technology. His research interests span computer graphics and computer vision.

DAN LI received the B.E. and M.S. degrees in mechanical design, manufacturing, and automation and the Ph.D. degree in computer science from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1998, 2002, and 2008, respectively. She is currently an Associate Professor with the School of Computer Science and Technology, HUST. Her research interests span computer graphics, multimedia, and intelligence.

TIANCHI CAO received the bachelor’s degree from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2019, where he is currently pursuing the master’s degree in computer science and technology with the Department of Computer Science and Technology. His main research directions are computer graphics and computer vision.

YUKI SUN received the bachelor’s degree from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2017, where she is currently pursuing the master’s degree in computer science and technology with the School of Computer Science and Technology. Her research interests span computer graphics and computer vision.
YINGSONG HU received the B.E., M.S., and Ph.D. degrees in computer science from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1988, 1991, and 2011, respectively. He is currently an Associate Professor with the School of Computer Science and Technology, HUST. His research interests span multimedia and intelligence.

JUNWEN JI received the B.E. degree in fluid transmission and control and the M.S. degree in mechanical design, manufacturing, and automation from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1994 and 2001, respectively. He is currently a Lecturer with the School of Computer Science and Technology, HUST. His research interests span computer graphics, multimedia, and intelligence.

* * *