IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 30, 2020, accepted July 19, 2020, date of publication July 23, 2020, date of current version August 4, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3011424

Generating and Editing Arbitrary Facial Images by Learning Feature Axis

NAN YANG[1,2,3,4], YUANYE XU[1,2,3,4], ZEYU ZHENG[1,2,3,4], LIANG QI[5], (Member, IEEE), XIWANG GUO[6,7], (Member, IEEE), AND TIANRAN WANG[1,2,4]

[1]Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China
[2]Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China
[3]Key Laboratory of Networked Control Systems, Chinese Academy of Sciences, Shenyang 110016, China
[4]University of Chinese Academy of Sciences, Beijing 100049, China
[5]Department of Intelligent Science and Technology, Shandong University of Science and Technology, Qingdao 266590, China
[6]Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA
[7]Computer and Communication Engineering College, Liaoning Shihua University, Fushun 113001, China

Corresponding authors: Zeyu Zheng (zhengzeyu@sia.cn) and Liang Qi (qiliangsdkd@163.com)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFF0214704, in part by the National Natural Science Foundation of China under Grant 61803367, Grant 61573089, Grant 61903229, and Grant 61973180, in part by the Natural Science Foundation of Liaoning Province under Grant 2019-MS-346, in part by the Liaoning Revitalization Talents Program under Grant XLYC1907166, in part by the Liaoning Province Department of Education Foundation of China under Grant L2019027, and in part by the Natural Science Foundation of Shandong Province under Grant ZR2019BF004 and Grant ZR2019BF041.

ABSTRACT There are mainly three limitations of the traditional facial attribute editing techniques: 1) incapability of generating an arbitrary facial image with high-resolution; 2) being unable to generate and edit new facial images synthesized by the computer and 3) limited diversity of edited images. This paper presents a method for generating and editing images simultaneously. It incorporates a high-resolution facial image generator, a multi-label classifier, and a Generalized Linear Model (GLM). Experimental results show that our method can generate arbitrary high-resolution facial images, edit computer-synthesized images, perform multi-attribute editing, and effectively control the intensity and style of the generated images. Besides, the approach has high efficiency and flexibility, allowing rapid migration of attribute information from the data set. We design a graphical interface program, which can be integrated as a mobile application.

INDEX TERMS Deep learning, generative adversarial networks, image generating, image editing.

I. INTRODUCTION

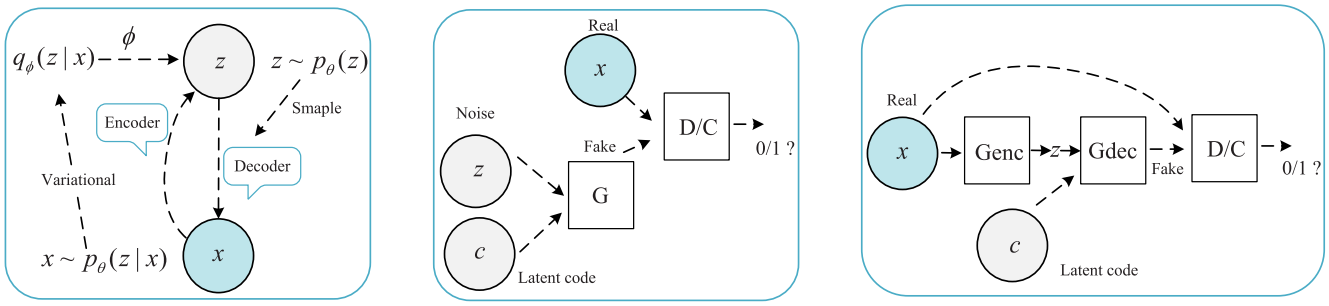
Facial attribute editing aims at manipulating an image to possess desired attributes while keeping the other details unchanged. It can be incorporated into other software products [1], especially mobile apps. Typically generative models include variational autoencoders (VAEs) [2], [3] and generative adversarial networks (GANs) [4], [5].

As shown in Figure 1(a), VAEs are generally designed to make the latent space satisfy a specific distribution and impose high-level semantics on the latent space. After the training, it can sample a latent vector $\mathbf{z}$ from the latent distribution $p_\theta(\mathbf{z})$ and generate a new face, while manipulating the variable factors in the latent space and realize the editing of the facial image. They are unsupervised methods, and the most prominent ones are β-VAE [6], β-TCVAE [7], and JointVAE [8]. However, their disadvantage is that they cannot generate high-resolution new facial images, nor can they specify the semantics of the variation factor.

GANs disentangle generative factors for facial image editing by maximizing the mutual information between the latent variables and the generated samples, such as InfoGAN [9] in Figure 1(b). It improves the quality of generated images, but it has some shortcomings, such as unstable training and low sample diversity. Recent improvements in the training of GANs have alleviated some of these problems [10]–[13]. However, stable GAN training remains a challenge due to the multimodal data, which prevents effective editing.

Some researchers have started to study the combination of VAEs and GANs [14], [15] to learn a latent representation and a decoder, as shown in Figure 1(c). The attribute editing is achieved by modifying the latent representation and capture the information on expected attributes and then decoding it. The facial image editing based on the encoder-decoder

The associate editor coordinating the review of this manuscript and approving it for publication was Jin-Liang Wang.



(a) VAE (b) GAN (c) VAE/GAN

FIGURE 1. Comparison of generator networks. a) VAE model. The variational approximation $q_\phi(\mathbf{z}|\mathbf{x})$ to the intractable posterior $p_\theta(\mathbf{z}|\mathbf{x})$. The variational parameters ϕ are learned jointly with the generative model parameters θ. $p_\theta(\mathbf{z})$ denotes a prior distribution, usually choosing a normal distribution. z denotes a latent vector, x denotes a generated image. b) GAN model. *G*, *D*, and *C* denote a generator, a discriminator, a classifier, respectively. x, z, and c denote real image, latent vector, and label vector, respectively. 0/1 denotes the result of the binary classification of real and fake images. (c) VAE-GAN model. The combination of VAE and GAN. *Genc*, *Gdec*, *D*, *C* denote encoder, decoder, discriminator, classifier. x, z, c denote real image, latent vector, label vector. 0/1 denotes the result of the binary classification of real and fake images.

architecture is a conditional generative model. Other prominent models include AttGAN [16] and STGAN [17]. They can edit existing images but not generate new facial images and edit computer-synthesized images.

Although the above models can edit facial attributes with an input image, they suffer from three limitations: 1) incapability of generating an arbitrary facial image with high-resolution; 2) being unable to generate and edit new facial images synthesized by computers and 3) limited diversity of edited images. To address the above dilemma, we investigate arbitrary attribute editing from uncovering feature axis perspective and present a novel facial image editing method. In terms of the *feature*, it refers to a multi-label classification vector of the synthesized image, and the *axis* is to find the correlation between a latent vector and a multi-label classification vector. Computer-synthesized images do not have labels and cannot be edited by using AttGAN and STGAN. Our method edits the image directly during the generative process without providing label information.

We fine-tune the pre-trained StyleGAN2 [18] generator and generate high-resolution facial images through latent vectors. Besides, we find that the latent space is dense, and the points in the latent space are relatively continuous. To edit the new images synthesized by the computer, we train a multi-label classifier *cls*. Then the classifier predicts a label vector $\mathbf{y}$ of the facial image $\mathbf{x}$. Finally, we use a Generalized Linear Model to perform regression between the latent vector $\mathbf{z}$ and its corresponding label vector $\mathbf{y}$. The regression slope $\mathbf{w}$ becomes the feature axis. Owing to the continuity of points in the GAN latent space and the diversification of classifier label vectors, moving along the feature axis can effectively control the style of the synthesized image. The code and model are available on `https://github.com/GreenLimeSia/Generating-and-Editing`. In summary, our key contributions are as follows:

- From the perspective of the learning feature axis, we propose a new approach to generate and edit images simultaneously. Our method supports high-resolution arbitrary facial image generation, editing of computer-synthesized images, multi-attribute editing, and effective control of the attribute intensity and style of generated images.
- We introduce a multi-label classifier to address the problem of computer-synthesized images without attribute labels. Then, the correlation between a latent vector and its corresponding label vector is constructed by a Generalized Linear Model. The regression slope becomes the feature axis.
- We create a graphical interface program that can be integrated into mobile applications.

II. PROPOSED METHOD

In this section, we present a framework, which includes new facial image synthesis and an editing process. Figure 2 shows the structure of our approach during the training and testing phases.

A. MODEL FRAMEWORK

G_mapping: StarGAN2 [19] has a mapping network f that produces diverse style codes $\mathbf{s}$ by sampling the latent vector $\mathbf{z}$ from the latent space $\mathcal{Z}$. However, f is to map latent vectors $\mathbf{z}$ to intermediate latent vectors $\mathbf{u}$. From [20], the intermediate latent space does not need to support any fixed distribution; its sampling density is derived from the learned piecewise continuous mapping $f(\mathbf{z})$. This mapping can be tuned to "untwist" $\mathcal{U}$ so that the factors of variation become more linear than the latent vector $\mathbf{z}$. The generator is easier to generate realistic images based on a linear disentangled representation than based on an entangled representation.

$$\mathbf{u} = f(\mathbf{z}), \quad f : \mathcal{Z} \to \mathcal{U} \tag{1}$$

G_synthesis: Given a disentangled intermediate latent vector $\mathbf{u}$, we use a synthetic network g to generate a high-resolution facial image $\mathbf{x}$. The synthetic network adopts the StyleGAN2 [18] architecture and fine-tunes the parameters. StyleGAN2 is a new approach proposed by NVIDIA,

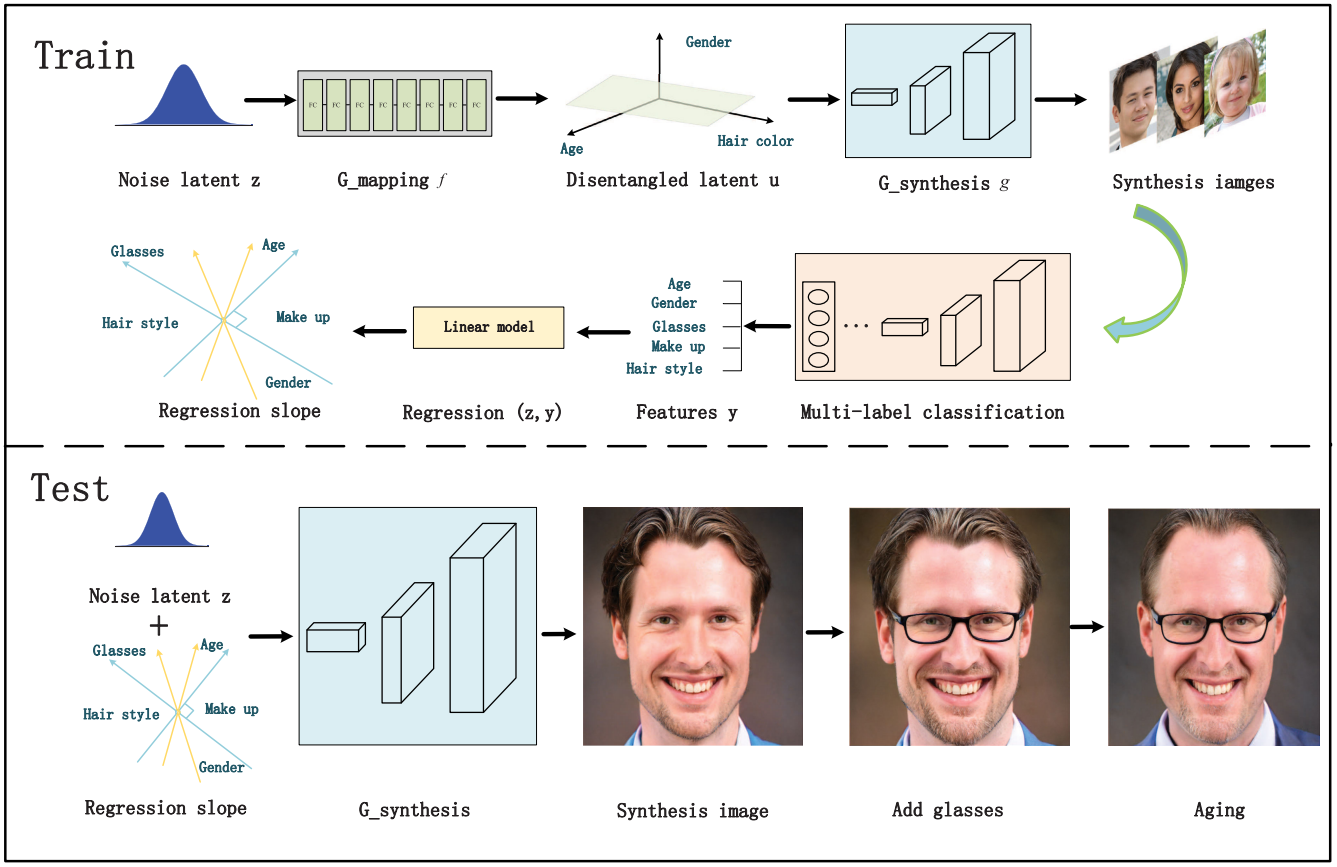


FIGURE 2. The architecture of our method. a) Training phase. The noise latent vector z generates the disentangled latent vector u by mapping the network *f*. u generates high-resolution images by the synthetic network *g*. It trains a multi-label classifier to capture the feature vector y of the synthetic image. Regression of the latent vector z and the feature vector y. The slope of the regression becomes the feature axis. b) Testing phase. A noise latent vector z is randomly generated, moving the latent vector in the direction of the feature axis. The moving latent vectors generate images through the synthetic network *g* and finally test the changes in the attributes of the generated images. The figure moves along the feature axis of glasses and age.

which is a redesign of the original StyleGAN [20].

$$\mathbf{x} = g(\mathbf{u}), \quad g : \mathcal{U} \to \mathcal{X} \tag{2}$$

Multi-label classifier: Given a synthetic image $\mathbf{x}$, a multi-label classifier *cls* extracts the features of $\mathbf{x}$ and predicts a label vector $\mathbf{y}$. The classifier is trained with real images and labels. Then, the classifier is used to predict the label vectors of the synthetic images. The value of each point in the label vector represents whether it contains a facial feature, such as young or old and male or female.

$$\mathbf{y} = cls(\mathbf{x}), \quad cls : \mathcal{X} \to \mathcal{Y} \tag{3}$$

Regression: Given a latent vector $\mathbf{z}$ and its corresponding label vector $\mathbf{y}$, we use a Generalized Linear Model to perform regression between latent vectors $\mathbf{z}$ and its corresponding label vectors $\mathbf{y}$. The regression slope $\mathbf{w}$ becomes the feature axis, $\mathbf{b}$ is a bias. The feature axis can be moved along the axis, which allows the G_synthesis network to synthesize an output image that controls the styles.

$$\mathbf{y} = \mathbf{w} \cdot \mathbf{z} + \mathbf{b} \tag{4}$$

Feature axis orthogonalization: Given a feature axis without orthogonalization, we find that it can cause attribute entanglement. To eliminate this issue, we perform a Gram-Schmidt [21] orthogonalization of the feature axis. Equation (5) is the Gram-Schmidt orthogonalization equation.

$$\begin{aligned}
\boldsymbol{\beta}_1 &= \boldsymbol{v}_1, \quad \boldsymbol{\eta}_1 = \frac{\boldsymbol{\beta}_1}{\|\boldsymbol{\beta}_1\|} \\
\boldsymbol{\beta}_2 &= \boldsymbol{v}_2 - \langle \boldsymbol{v}_2, \boldsymbol{\eta}_1 \rangle \boldsymbol{\eta}_1, \quad \boldsymbol{\eta}_2 = \frac{\boldsymbol{\beta}_2}{\|\boldsymbol{\beta}_2\|} \\
\boldsymbol{\beta}_3 &= \boldsymbol{v}_3 - \langle \boldsymbol{v}_3, \boldsymbol{\eta}_1 \rangle \boldsymbol{\eta}_1 - \langle \boldsymbol{v}_3, \boldsymbol{\eta}_2 \rangle \boldsymbol{\eta}_2, \quad \boldsymbol{\eta}_3 = \frac{\boldsymbol{\beta}_3}{\|\boldsymbol{\beta}_3\|} \\
&\vdots \quad \vdots \\
\boldsymbol{\beta}_n &= \boldsymbol{v}_n - \sum_{i=1}^{n-1} \langle \boldsymbol{v}_n, \boldsymbol{\eta}_i \rangle \boldsymbol{\eta}_i, \quad \boldsymbol{\eta}_n = \frac{\boldsymbol{\beta}_n}{\|\boldsymbol{\beta}_n\|}
\end{aligned} \tag{5}$$

We use $\boldsymbol{v} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ to represent the feature axis without orthogonalization. Equation (5) shows the process of orthogonalization of the feature axis. $\boldsymbol{\beta} = \{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n\}$ is the orthogonal basis of the feature axis $\boldsymbol{v}$. $\boldsymbol{\eta} = \{\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n\}$ represents the standard orthogonal basis of the feature axis $\boldsymbol{v}$.

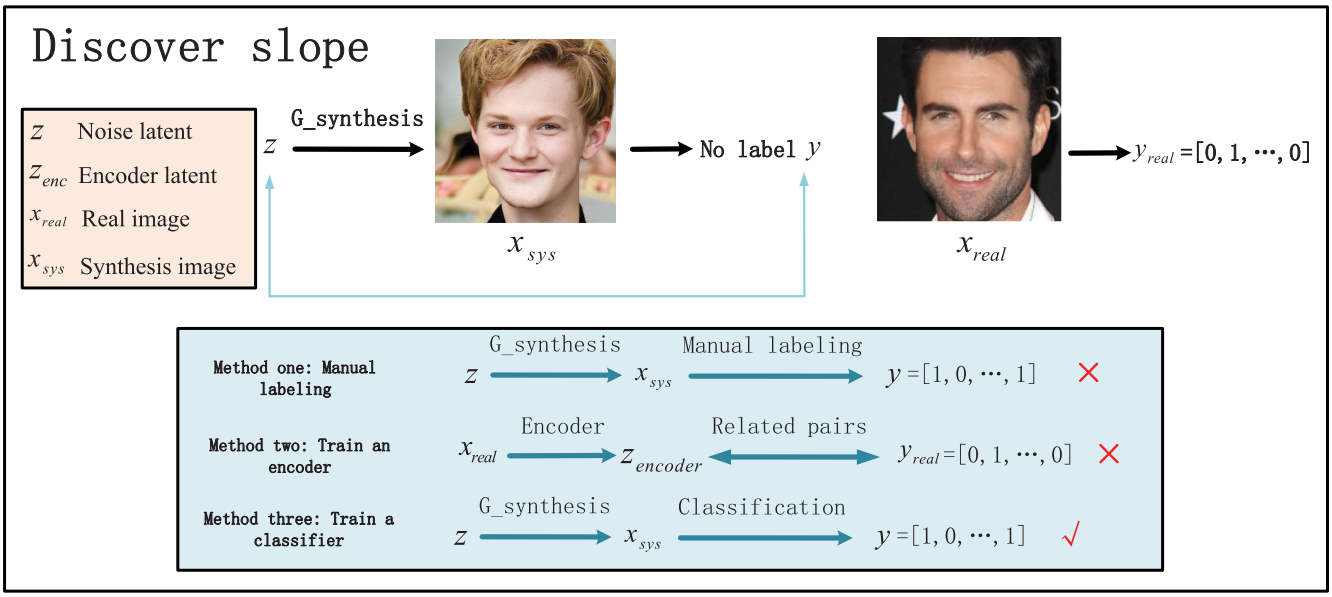


FIGURE 3. Discover the feature axis. Three main methods for obtaining data pairs of latent vector z and its corresponding label vector y.

$\frac{\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|}$ represents the unit vector of $\boldsymbol{\beta}$. $\langle \boldsymbol{v}_2, \boldsymbol{\eta}_1 \rangle$ represents the inner product of $\boldsymbol{v}_2$ and $\boldsymbol{\eta}_1$. The orthogonalized feature vector $\boldsymbol{\beta}$ eventually becomes the feature axis.

B. DETAILS OF DISCOVERING FEATURE AXIS

The purpose of finding the feature axis is to discover the relationship between a latent vector **z** and their corresponding label vector **y**. Then the feature axis can be gradually adjusted from the latent vector to obtain a facial image with the desired attributes. However, it is impracticable to make labels for computer-synthesized facial images. Therefore, our main task is to build label vectors and compose data pairs of latent vectors and label vectors. There are three major approaches to build label vectors, as shown in Figure 3.

Manual labeling: The synthesized facial images are labeled manually. This approach is time-consuming and laborious due to a large number of facial image samples. Therefore, this solution is not feasible.

Encoder strategy: Given a real facial image, we train an encoder to map the facial image to a latent vector. However, it is only possible to build data pairs of latent vectors and label vectors of real images. Since the two latent vectors are not identical, the latent vectors of the synthesized image and their corresponding label vector data pairs cannot be available. Therefore, this solution is not feasible.

Classifier strategy: It trains a multi-label classifier by using real facial images and its corresponding label vectors. Then, it predicts synthesized facial image features by using the above-trained classifier. The method quickly predicts the label vectors of facial images. The better the label classifier is, the more accurate the predicted label vectors will be. Therefore, this solution is feasible.

We train a multi-label classifier on the CelebA-HQ [22] dataset, which contains 30,000 images with 40 facial attributes per image. We perform 13 iterations with an average accuracy of 99.24%. In this way, we build data pairs of latent vectors **z** and their corresponding label vectors **y**.

Given a latent vector **z** and its corresponding label vector **y**, we have two ways to establish the relationship between the latent vector **z** and its corresponding label vector **y**. First, regression of latent vectors **z** and their corresponding label vectors **y** use a Generalized Linear Model, refer to 4. The regression slope **w** becomes the feature axis. Second, we train a single-label classifier by using latent vectors and their corresponding label vectors. The parameters of the intermediate hidden layer become the feature axis. The disadvantage of the second approach is that controlling multiple attributes requires training multiple single-label classifiers. To achieve multi-attribute control using a single model, we adopt the first method.

The regression slope becomes the feature axis, which ensures that we control multiple attributes with a single model. However, we find that the entanglement of feature axes leads to the entanglement of attributes. Manipulating one attribute leads to a change in another attribute, and we cannot manipulate multiple attributes. Therefore, we orthogonalize the feature axis. The dimensions are orthogonal to each other. Each dimension contains a single attribute. The orthogonalized feature axis can manipulate multiple attributes, while the other attributes remain unchanged.

In the test phase, the generator randomly synthesizes the facial image by using the latent vector **z**. The dimension of the feature axis **w** is 512×40. It contains 40 feature axes, and each feature axis has a dimension of 512. The dimension of the feature axis and the latent vector **z** are equal.

FIGURE 4. Generating high-quality facial images using our method. Images are labeled by cute baby, male, female. The generating network g synthesizes images by randomly sampling latent vectors, so our method supports arbitrary facial image generation.

The feature axis contains information about the attributes. Moving along the feature axis allows us to manipulate the attributes. By moving the feature axis of the glasses, the glasses can be added without changing other details. Besides, by adding the feature axis of age linearly, the facial image can age naturally as shown in Figure 2.

C. PROCEDURE OF THE ALGORITHM

The above sections illustrate the architecture of the model and the capability of each component. In this subsection, we give five steps to implement our approach and present algorithm for learning feature axis.

Step 1: Training a generator. Select a GAN model as a generator network. We train the GAN generator network, or fine-tune the parameters. The StyleGAN2 [18] parameter is fine-tuned, which provides us with high-resolution facial images.

Step 2: Training a multi-label classifier. We train a multi-label feature extractor, which extracts features of facial images, i.e., label vectors. The average accuracy rate of the multi-label classifier is 99.24% during the testing phase.

Step 3: Building label vectors. A large number of latent vectors are randomly generated and transferred to a well-trained StyleGAN2 generator to produce synthesized images. Then the features are captured for each image by using the above pre-trained feature extractor, i.e., the label vector of the synthesized image.

Step 4: Discovering feature slope. Given a latent vector and its corresponding label vector, we use a Generalized Linear Model to perform regression between latent vectors and features. The regression slope becomes the feature axis.

Step 5: Manipulating facial images. We start with a latent vector and move along one or more feature axes to control the attributes of the synthesized facial image.

Algorithm 1 The Training Pipeline of Learning Feature Axis Algorithm

Input: $\mathbf{x}$, real image. $\mathbf{y}$, real label vector. $\mathbf{z}$, latent vector. θ_G, θ_C denotes the initial network parameters for Generator Model, Classifier Model. $\rightarrow$ denotes space mapping, and $\xleftarrow{+}$ denotes updating gradient.

Output: $\mathbf{w}$, feature axis.

while θ_G *has not converged* **do**
  Sample latent vector $\mathbf{z} \sim \mathcal{Z}$, a batch data from latent space;
  $\mathbf{u} \leftarrow f(\mathbf{z}), f : \mathcal{Z} \rightarrow \mathcal{U}$;
  $\mathbf{x_{sys}} \leftarrow g(\mathbf{u}), g : \mathcal{U} \rightarrow \mathcal{X}_{sys}$;
  $\theta_G \xleftarrow{+} -\nabla_{\theta_G}(\mathcal{L}_G)$
end
while θ_C *has not converged* **do**
  Sample real image $\mathbf{x} \sim \mathcal{X}$, a batch data from real image space;
  $\mathbf{y} \leftarrow cls(\mathbf{x}), cls : \mathcal{X} \rightarrow \mathcal{Y}$;
  $\theta_C \xleftarrow{+} -\nabla_{\theta_C}(\mathcal{L}_C)$
end
for *epoch in range epochs* **do**
  Sample latent vector $\mathbf{z} \sim \mathcal{Z}$, a batch data from latent space;
  $\mathbf{u} \leftarrow f(\mathbf{z}), f : \mathcal{Z} \rightarrow \mathcal{U}$;
  $\mathbf{x_{sys}} \leftarrow g(\mathbf{u}), g : \mathcal{U} \rightarrow \mathcal{X}_{sys}$;
  $\mathbf{y_{sys}} \leftarrow cls(\mathbf{x_{sys}}), cls : \mathcal{X}_{sys} \rightarrow \mathcal{Y}_{sys}$;
  Build data pair $(\mathbf{z}, \mathbf{y_{sys}})$
end
if *data pair* $(\mathbf{z}, \mathbf{y_{sys}}) \neq$ *None* **then**
  Perform regression $\leftarrow \mathbf{y_{sys}} = \mathbf{w} \cdot \mathbf{z} + \mathbf{b}$;
  feature axis $\leftarrow$ orthogonalize $\mathbf{w}$;
end
final;
return feature axis $\mathbf{w}$;

We train a generator and a multi-label classifier, updating θ_G, θ_C, until convergence. $\mathcal{L}_G$ and $\mathcal{L}_C$ represent the loss of the generator and the classifier respectively. Using the well-trained generator model and the classifier model, we construct data pairs. We perform a regression of the data pairs using a Generalized Linear Model. The feature slope is orthogonalized by using Equation (5) and eventually used as the feature axis.

III. EXPERIMENTS AND RESULTS

In this section, we present our results through some experiments. The experiment focuses on three contributions, which include generating arbitrary facial images with high-resolution, editing computer-synthesized images, and realizing the diversity of edited images. We will make the Youtube video available at `https://youtu.be/uRwbQGzHIII`.

A. IMPLEMENTATION DETAILS

We extract the official StyleGAN2 face generator ffhq-config-f. We convert the generator to the Pytorch version

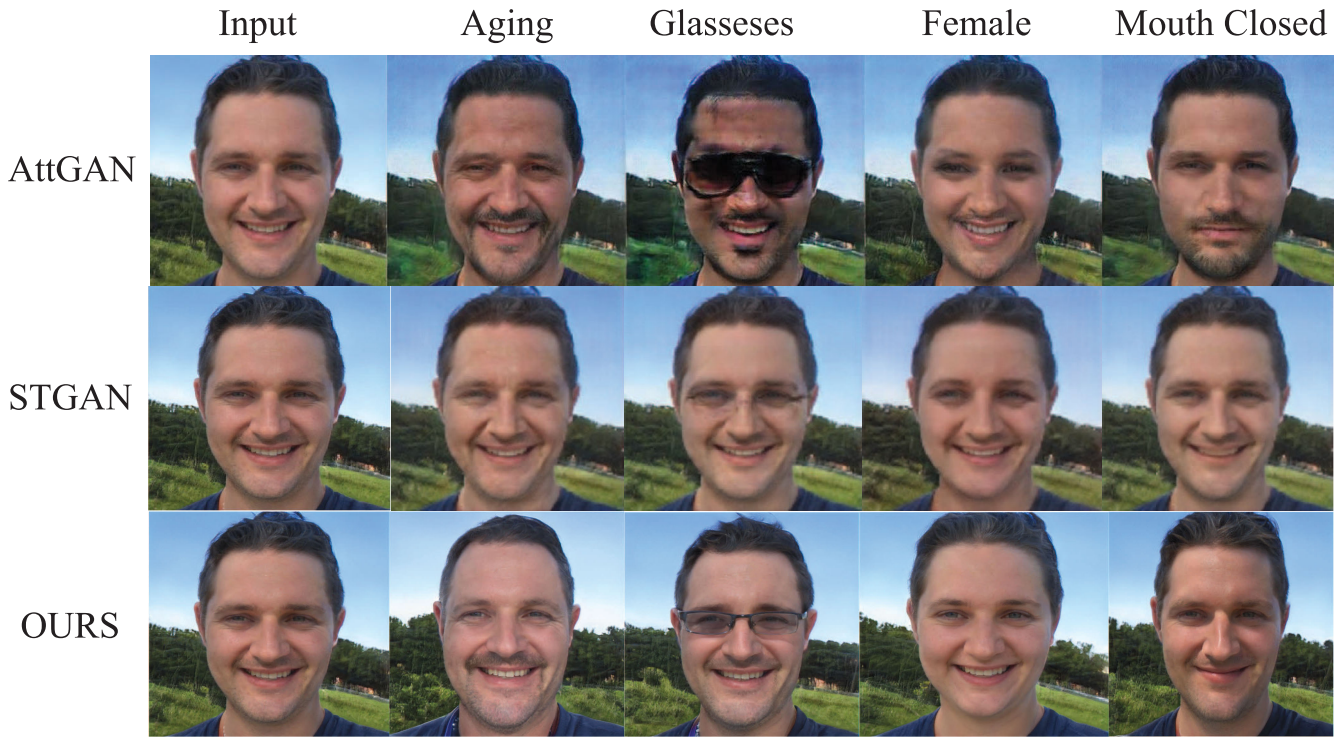


FIGURE 5. Comparison diagram of our method with AttGAN and STGAN in single attribute editing. For each specified attribute, the facial attribute editing here is to invert it, e.g., to edit male to female, or mouth open to mouth close.

and save it as our baseline model. To capture the feature axis, we train a multi-label classifier on the CelebA-HQ dataset. The models involved in the experiment are trained on a workstation equipped with Intel(R) Xeon(R) CPU @ 2.20GHz and a dual-channel NVIDIA Tesla P100 GPU. All experiments are performed in a Pytorch 1.5 environment, with Cuda 10.0.44 and cuDNN 10.0.20. The baseline model is fine-tuned in the original experimental setting. The classifier has 13 iterations during the training phase. The model is trained using *Adam* optimizer ($\beta_1 = 0.5$, $\beta_2 = 0.999$) with a learning rate of 0.002. During the testing phase, the average accuracy of the classifier model is 99.24%.

CelebA-HQ is a large-scale dataset of facial attributes, consisting of 30,000 facial images, each of which has 40 binary attribute labels. These attributes cover the most distinctive facial attributes, contain practical information about human-computer interaction, and are also widely used in [23]. We train a multi-label classifier using a 1024×1024 resolution. We divided CelebA-HQ into a training set, a validation set, and a test set. The training set and the validation set are used to train the classifier, and the test set is used in the evaluation phase.

B. GENERATING FACIAL IMAGES

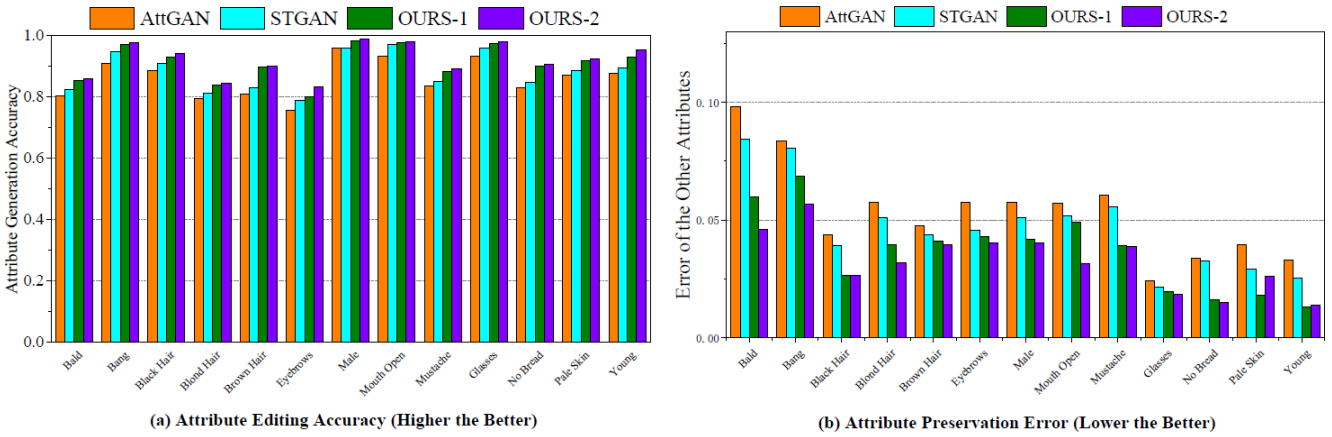
The generator adopts the architecture of StyleGAN2 [18], and the facial images generated are shown in Figure 4.

To measure the quality of the generated images, we quantitatively analyze the generated images by using three metrics: Frechet Inception Distance (FID), Precision and Recall (P&R), and Perceptual Path Length (PPL). FID [24] measures differences in the density of two distributions in the high dimensional feature space of an InceptionV3 classifier [25]. P&R [11], [26] provide additional visibility by explicitly quantifying the percentage of generated images that are similar to training data and the percentage of training data that can be generated, respectively. Many studies have shown that PPL [20] with low scores is indeed a sign of high-quality images, and vice versa.

TABLE 1. The main results of quantitative comparison with CelebA-HQ at 1024^2. For each training, we select the training snapshot with the lowest FID. The path length corresponds to the PPL metric. ↑ indicates that the higher the better, and ↓ that the lower the better.

CelebA-HQ, 1024×1024				
Method	FID ↓	Path length ↓	Precision ↑	Recall ↑
TL-GAN	7.30	412.0	0.61	0.42
StyleGAN	5.425	212.1	0.796	0.406
OURS	4.58	173.8	0.7343	0.625

In this paper, we calculate the FIDs using 20,000 images drawn randomly from the training set and report the lowest distance encountered throughout the experiment. Table 1 shows the results of the experimental comparison, and the data shows that the quality of the generated images performs a significant improvement. Our method yields the smallest FID score, which means that the distribution difference between the generated image and the real image is minimal, i.e., the generated image is more realistic. The PPL metric also gets



(a) Attribute Editing Accuracy (Higher the Better)

(b) Attribute Preservation Error (Lower the Better)

FIGURE 6. Comparisons among AttGAN, STGAN, and our method in terms of a) facial attribute editing accuracy and b) preservation error of the other attributes. OURS-1 and OURS-2 denote the activation of the Generalized Linear Model is linear and tanh.

the lowest score, which means that the images generated have a high quality. Our approach is better than TL-GAN [27] in precision and recall metrics, and the score is lower than StyleGAN [20] 6% in precision metric. However, in the recall metric, we exceed StyleGAN by 20%. By combining the four metrics, our approach can generate high-quality facial images. Among them, TL-GAN uses the baseline model of PG-GAN, and StyleGAN is the baseline model of StyleGAN2 [18].

C. EDITING COMPUTER-SYNTHESIZED IMAGES

Given a latent vector and its feature axis, we move along the feature axis on the latent vector to control the attributes of the synthesized image. To emphasize the effect of attribute editing, we compare AttGAN [16] and STGAN [17], which are designed for attribute editing. Figure 5 shows the attribute editing results of the three methods.

AttGAN adopts the encoder-decoder architecture, and the image becomes blurred after editing with attribute information. Hence, it is difficult for AttGAN to guarantee the quality of the generated images in practical applications. STGAN follows the architecture of AttGAN, which uses selective transfer units and differential signals as inputs. While the quality of the generated images is guaranteed to some extent, the trade-off between generating and editing remains. The disadvantage of both methods is that they require attribute information as additional input. Our method moves directly on the feature axis for attribute editing, which not only ensures the quality of image generation but also takes the editing effect into account.

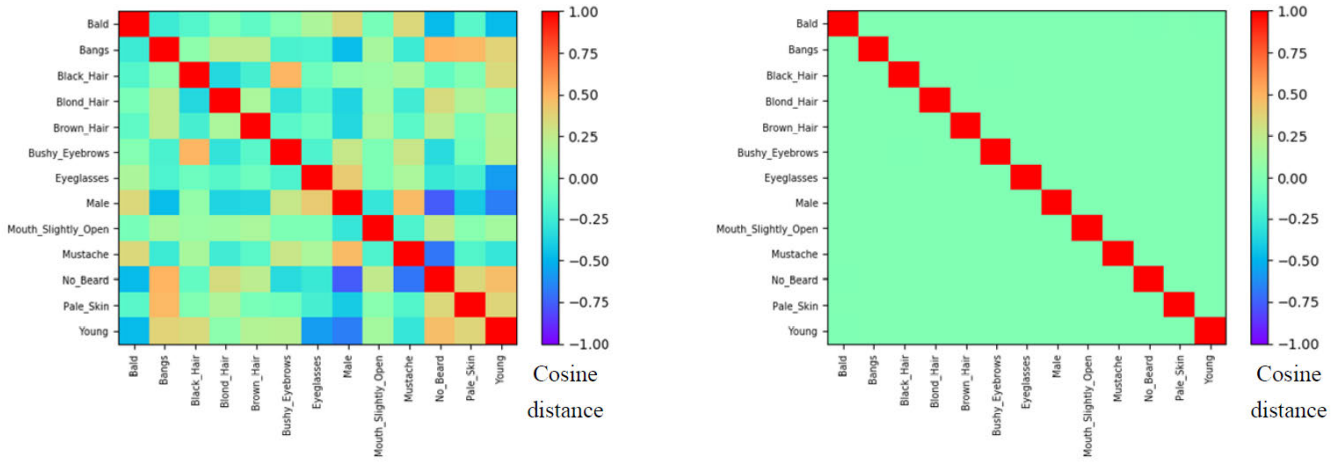
Table 2 shows the advantages of our approach compared to AttGAN and STGAN. Our method can generate new facial images, single attribute editing, and multi-attribute editing. AttGAN and STGAN can only perform single-attribute editing, cannot generate new facial images, and cannot perform multi-attribute editing. Besides, our method does not require additional attribute information to edit the facial image. Other methods require additional attribute information as input.

TABLE 2. A check table of different models. GNF denotes generating new facial image. SAE denotes single attribute editing. MAE denotes multi-attribute editing. NRL denotes no labels required.

Method	GNF	SAE	MAE	NRL
AttGAN	×	√	×	×
STGAN	×	√	×	×
OURS	√	√	√	√

The evaluation metrics [28] for image reconstruction results are Peak Signal to Noise Ratio (PSNR) [29] and Structural SIMilarity (SSIM) [30]. PSNR is the most common and widely-used evaluation metric for image reconstruction based on the corresponding pixel-to-pixel error. The PSNR has no consideration of human visual characteristics. As a full-reference image quality evaluation metric, SSIM measures the similarity of images in terms of brightness, contrast, and structure. The SSIM outperforms PSNR in terms of image denoising and similarity evaluation. The PSNR/SSIM results for the three image reconstruction methods are shown in Table 3. The quantitative results are consistent with the qualitative results in Figure 5. From Table 3, benefiting from the StyleGAN2 generator, our method can retain more image information and achieves much better reconstruction results than its two peers. Our approach can generate high-quality reconstruction results, which are more natural and realistic while retaining more details.

Quantitative evaluation of attribute editing is indispensable. There are two main aspects. We need to assess whether the facial image has the desired attributes, i.e., the attribute generation accuracy. We need to evaluate whether the other attributes remain unchanged, i.e., the attribute preservation error. We test 13 attributes with the above trained multi-label classifiers. The attribute generation accuracy and attribute preservation error are shown in Figure 6. Orthogonalized feature axes do not cause entanglement of attributes. Manipulating one attribute does not cause the other attributes to be changed. The classification results show that our method outperforms other methods in terms of attribute generation accuracy and attribute preservation error.



(a) Cosine similarity of feature axis

(b) Cosine similarity of orthogonalized feature axis

FIGURE 7. Cosine similarity between feature axis. a) denotes the cosine similarity of the feature axis without orthogonalization. b) denotes the cosine similarity of the feature axis with orthogonalization.

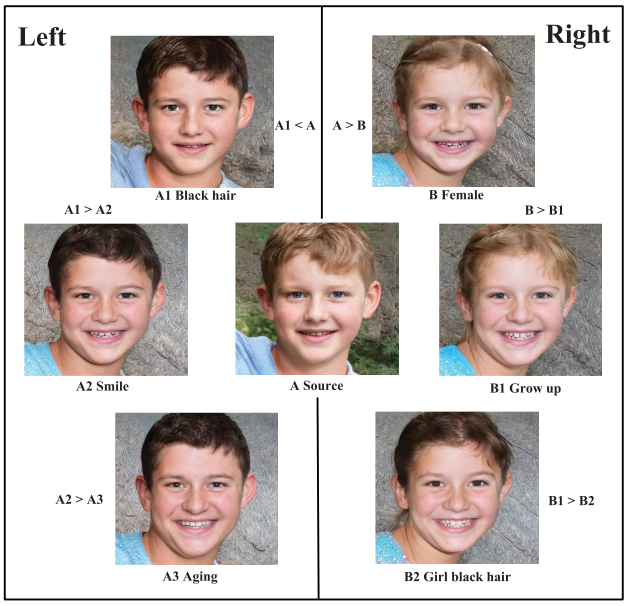


FIGURE 8. Multi-attribute editing schematic. The term multi-attribute editing refers to the fact that the edited image can continue with other manipulations without changing the previously edited attributes. A denotes the source image, and > denotes the editing direction. The left picture is edited in the direction of A>A1>A2>A3, which means black hair, smile, and aging. The right image is edited in the direction of A>B>B1>B2, which indicates female, grow up, and black hair.

TABLE 3. Image Reconstruction Quality. Comparison of reconstruction performance in terms of SSIM and PSNR (mean and standard deviation). The higher the value, the better the quality.

Metric	AttGAN	STGAN	OURS
PSNR	24.81 ± 0.1213	26.80 ± 0.5805	28.79 ± 0.3510
SSIM	0.70 ± 0.0029	0.88 ± 0.0534	0.92 ± 0.067

D. CONTROLLING MULTIPLE ATTRIBUTES AND STYLES

The lack of orthogonalization of the feature axis can cause entanglement among attributes. We characterize the cosine distance of each feature axis by cosine similarity. As shown in Figure 7, the cosine distance between the two non-orthogonal feature axis is not equal to zero. Moving the non-orthogonal feature axis causes other attributes to change. To eliminate similarities between the feature axis, we orthogonalize the feature axis. The cosine distance between the feature axis after orthogonalization is equal to zero. Only the diagonal element distance is equal to one. Orthogonalization of the feature axes allows us to control the style and intensity of the attributes exactly.

Our method supports controlling multiple attributes of a single image, as shown in Figure 8. The term multi-attribute control refers to the fact that a single image can manipulate any attribute simultaneously, and the manipulated resulting image can continue to be edited at will. Besides, our method can control the style and intensity of the attribute, such as the ability to add different styles of glasses for the same person, or the ability to control the gradient process of the face turning old. The experimental results are shown in Figure 9.

IV. DISCUSSION

Previous methods require providing a facial image and its corresponding label information to the model. The model encodes the image and then uses the label information as additional input. Besides, the model decodes the latent vectors to edit the facial image. Due to the complex structure of using the encoder-decoder, it is difficult to ensure the quality of the generated images. Without the label information, editing cannot be achieved either. However, our approach gets rid of these two limitations. We focus on the generative process and edit the image from the perspective of the feature axis. It does not need to provide label information as an additional input since our attribute information is contained on the feature axis. It is worth mentioning that the feature axis can be migrated repeatedly to other GAN models.

Our method can generate high-resolution arbitrary facial images because of the use of the powerful StyleGAN2

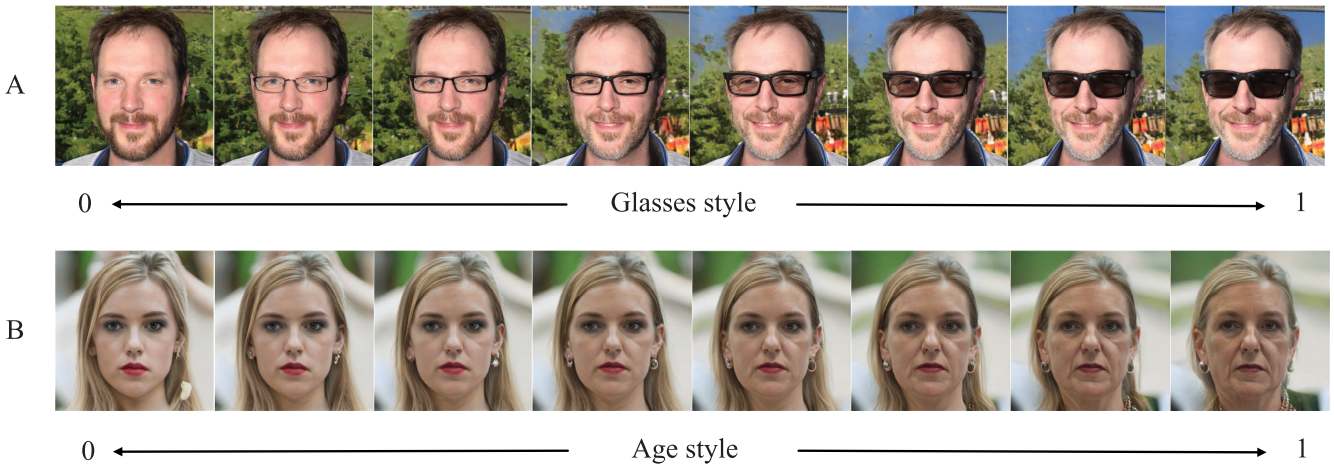


FIGURE 9. Add different styles to facial images. Moving the attribute values linearly along the feature axis, our model can change the style of the facial attribute. A indicates the style of adding glasses, and B indicates the style of facial aging.

generator. The architecture takes a progressive growing approach to image generation and can effectively control the details of the generated images. The latent space dimension of each image is 18×512, and, latent space has two excellent characteristics. First, the latent space is dense, which means that each point in the latent space corresponds to the generated image. Second, the points in the latent space are relatively continuous, which means that the difference between the two points causes a smooth transition of the images.

Learning the feature axis makes it possible to edit computer-synthesized images. Since the image has no label information, we additionally trained a multi-label classifier to capture the relationship between the latent vector and its corresponding label vector. Thus, given the latent vector and its corresponding label vector, the regression task is performed by using a Generalized Linear Model. The regression slope becomes the feature axis. Linear variations in the feature axis cause natural variations in the image.

Our method can control 40 attributes by the multi-label classifier, which correlates the relationship between the latent vector and its corresponding label vector. Besides, the two excellent characteristics of the latent space and the linear variation of the feature axis allow us to manipulate the style of the facial image smoothly.

V. CONCLUSION AND FUTURE WORKS

In this paper, a model for generating and editing facial attributes is proposed to 1) generate arbitrary high-resolution images, 2) edit new computer-synthesized images without providing additional label information, and 3) control the diversity of attribute styles. We use the StyleGAN2 generator as a baseline model, by combining a multi-label classifier and a generalized linear model to capture the relationship between the latent vector and its corresponding label vector. The regression slope becomes the feature axis. The feature axis contains attribute information. We move along the feature axis to control the attribute and style of the synthesized facial image.

Our approach has high efficiency and flexibility. We can rapidly train a multi-label classifier to capture the relationship between latent vectors and its corresponding label vectors. Besides, multi-label classifiers can be applied to any dataset. Thus, our approach can capture any feature on other datasets without retraining the GAN model. The proposed approach will be used in many other fields such as e-commerce systems [31], transportation systems [32], and manufacturing [33], [34].

For editing specific faces in reality, we will focus on the reverse mapping of generators for editing high-resolution specific faces in our future work. Face customization is another consideration for the robust generation capabilities of StyleGAN2.

REFERENCES

[1] X. Guo, S. Liu, M. Zhou, and G. Tian, "Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and Petri nets," *IEEE Trans. Cybern.*, vol. 46, no. 11, pp. 2435–2446, Nov. 2016.

[2] D. P Kingma and M. Welling, "Auto-encoding variational bayes," 2013, *arXiv:1312.6114*. [Online]. Available: http://arxiv.org/abs/1312.6114

[3] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," 2019, *arXiv:1906.02691*. [Online]. Available: http://arxiv.org/abs/1906.02691

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[5] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," 2020, *arXiv:2001.06937*. [Online]. Available: http://arxiv.org/abs/2001.06937

[6] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "β-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. ICLR*, vol. 2, no. 5, 2017, p. 6.

[7] T. Q. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud, "Isolating sources of disentanglement in variational autoencoders," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2610–2620.

[8] E. Dupont, "Learning disentangled joint continuous and discrete representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 710–720.
[9] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.
[10] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017, *arXiv:1701.07875*. [Online]. Available: http://arxiv.org/abs/1701.07875
[11] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila, "Improved precision and recall metric for assessing generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 3929–3938.
[12] X. Guo, S. Liu, M. Zhou, and G. Tian, "Dual-objective program and scatter search for the optimization of disassembly sequences subject to multiresource constraints," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 3, pp. 1091–1103, Jul. 2018.
[13] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5767–5777.
[14] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," 2015, *arXiv:1512.09300*. [Online]. Available: http://arxiv.org/abs/1512.09300
[15] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua, "CVAE-GAN: Fine-grained image generation through asymmetric training," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2745–2754.
[16] 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.
[17] M. Liu, Y. Ding, M. Xia, X. Liu, E. Ding, W. Zuo, and S. Wen, "STGAN: A unified selective transfer network for arbitrary image attribute editing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3673–3682.
[18] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," 2019, *arXiv:1912.04958*. [Online]. Available: http://arxiv.org/abs/1912.04958
[19] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "StarGAN v2: Diverse image synthesis for multiple domains," 2019, *arXiv:1912.01865*. [Online]. Available: http://arxiv.org/abs/1912.01865
[20] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4401–4410.
[21] Å. Björck, "Numerics of gram-Schmidt orthogonalization," *Linear Algebra Appl.*, vols. 197–198, pp. 297–316, Jan. 1994.
[22] 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
[23] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan++: How to edit the embedded images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8296–8305.
[24] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6626–6637.
[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[26] M. S. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly, "Assessing generative models via precision and recall," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 5228–5237.
[27] S. Guan. *Tl-Gan: Transparent Latent-Space Gan*. Accessed: 2018. [Online]. Available: https://github.com/summitkwan/transparentlatentgan
[28] A. Borji, "Pros and cons of GAN evaluation measures," *Comput. Vis. Image Understand.*, vol. 179, pp. 41–65, Feb. 2019.
[29] J. Snell, K. Ridgeway, R. Liao, B. D. Roads, M. C. Mozer, and R. S. Zemel, "Learning to generate images with perceptual similarity metrics," 2015, *arXiv:1511.06409*. [Online]. Available: http://arxiv.org/abs/1511.06409
[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[31] L. Qi, W. Luan, X. S. Lu, and X. Guo, "Shared P-type logic Petri net composition and property analysis: A vector computational method," *IEEE Access*, vol. 8, pp. 34644–34653, 2020.
[32] L. Qi, M. Zhou, and W. Luan, "A dynamic road incident information delivery strategy to reduce urban traffic congestion," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 5, pp. 934–945, Sep. 2018.
[33] X. Guo, M. Zhou, S. Liu, and L. Qi, "Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption," *IEEE Trans. Autom. Sci. Eng.*, early access, Jun. 19, 2020, doi: 10.1109/TASE.2020.2992220.
[34] X. Guo, M. Zhou, S. Liu, and L. Qi, "Lexicographic multiobjective scatter search for the optimization of sequence-dependent selective disassembly subject to multiresource constraints," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3307–3317, Jul. 2020.

NAN YANG received the B.S. degree in automation from Qufu Normal University, Rizhao, China, in 2016. He is currently pursuing the Ph.D. degree with the Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang. His current research interests include machine learning, information theory, deep generative learning, and computer vision.

YUANYE XU received the B.S. degree in automation from Shandong University, Weihai, China, in 2016. He is currently pursuing the Ph.D. degree with the Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang. His current research interests include machine learning, deep learning, deep generative learning, and computer vision.

ZEYU ZHENG received the B.S. degree in mechanical engineering from Zhejiang University, Zhejiang, China, in 1997, and the Ph.D. degree from The Graduate University for Advanced Studies, Japan, in 2005. He is currently a Professor with the Institute of Automation, Chinese Academy of Sciences, Shenyang, China. He has authored nearly 50 technical papers in journals and conference proceedings. His research interests include intelligent optimization algorithms, big data processing technology, data mining, project management, and complex systems.

LIANG QI (Member, IEEE) received the B.S. degree in information and computing science and the M.S. degree in computer software and theory from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively, and the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2017. From 2015 to 2017, he was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He is currently with the Shandong University of Science and Technology. He has published over 60 papers in journals and conference proceedings, including the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, the IEEE TRANSACTIONS ON SYSTEM, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and the IEEE TRANSACTIONS ON CYBERNETICS. His current research interests include Petri nets, optimization algorithms, machine learning, and intelligent transportation systems. He received the Best Student Paper Award-Finalist in the 15th IEEE International Conference on Networking, Sensing and Control (ICNSC'2018).

XIWANG GUO (Member, IEEE) received the B.S. degree in computer science and technology from the Shenyang Institute of Engineering, Shenyang, China, in 2006, the M.S. degree in aeronautics and astronautics manufacturing engineering from Shenyang Aerospace University, Shenyang, in 2009, and the Ph.D. degree in system engineering from Northeastern University, Shenyang, in 2015. He is currently an Associate Professor with the College of Computer and Communication Engineering, Liaoning Shihua University. From 2016 to 2018, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He has authored more than 50 technical papers in journals and conference proceedings, including the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON SYSTEM, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, and the IEEE/CAA JOURNAL OF AUTOMATICA SINICA. His current research interests include Petri nets, remanufacturing, recycling, and reuse of automotive, intelligent optimization algorithm.

TIANRAN WANG was born in Heilongjiang, China, in 1943. He graduated in computer science from the Harbin Institute of Technology, in 1967. From 1982 to 1985, he was a Visiting Scholar with Carnegie Mellon University. In 2003, he was elected as an Academician of the Chinese Academy of Engineering. He is currently a Researcher and the Ph.D. Tutor. He is the Dean of the School of Automation, Beijing University of Posts and Telecommunications. His main research interests include digital intelligent manufacturing technology, data mining, and robotics. He won the title of Outstanding Young and Middle-Aged Scientists and the Outstanding Scientific and Technological Workers of the Country and the Ho Leung Ho Lee Science and Technology Progress Award. He is the Chairman of the Liaoning Automation Society and the Executive Director of the China Automation Society.