

# LC-NeRF: Local Controllable Face Generation in Neural Randiance Field

Wenyang Zhou<sup>1</sup> Lu Yuan<sup>2</sup> Shuyu Chen<sup>3</sup> Lin Gao<sup>3</sup> Shimin Hu<sup>1</sup>

<sup>1</sup>Tsinghua University <sup>2</sup>Stanford University

<sup>3</sup>Institute of Computing Technology, CAS and University of Chinese Academy of Sciences

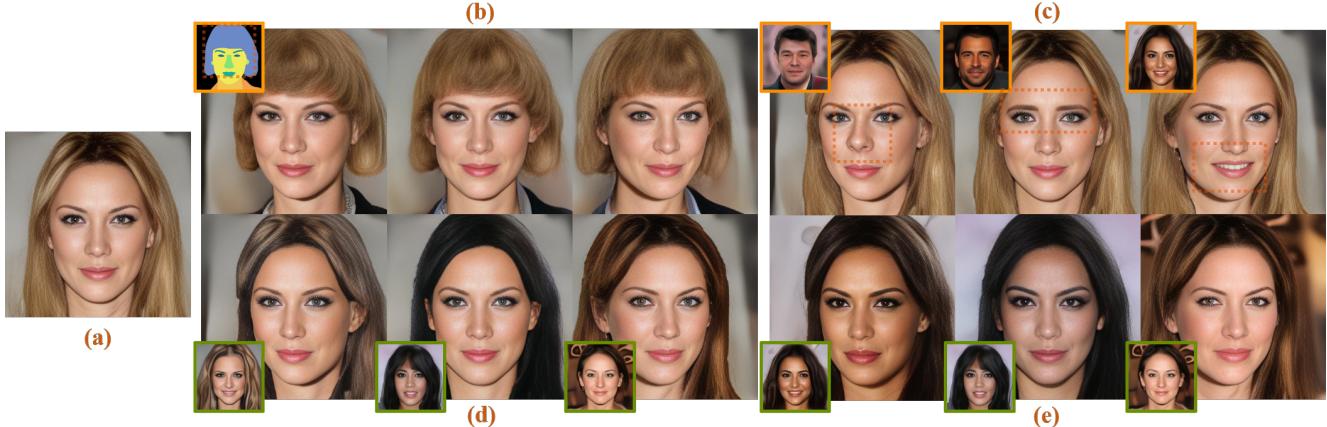


Figure 1. Given an input image (a) and a given reference face image, our method can independently edit the geometry of any local region, such as hair, nose, mouth, eyebrows, etc. We show these local and global face editing tasks achieved with our method: (b) editing the geometry of the hair by modifying the semantic mask while retaining the geometry of other regions and 3D consistency; (c) editing the geometry of more local regions, such as nose, eyebrows and mouth; (d) editing the local texture of the hair while retaining the geometry and 3D consistency; (e) editing the global texture while retaining the geometry and 3D consistency.

## Abstract

3D face generation has achieved high visual quality and 3D consistency thanks to the development of neural radiance fields (NeRF). Recently, to generate and edit 3D faces with NeRF representation, some methods are proposed and achieve good results in decoupling geometry and texture. The latent codes of these generative models affect the whole face, and hence modifications to these codes cause the entire face to change. However, users usually edit a local region when editing faces and do not want other regions to be affected. Since changes to the latent code affect global generation results, these methods do not allow for fine-grained control of local facial regions. To improve local controllability in NeRF-based face editing, we propose LC-NeRF, which is composed of a Local Region Generators Module and a Spatial-Aware Fusion Module, allowing for local geometry and texture control of local facial regions. Qualitative and quantitative evaluations show that our method provides better local editing than state-of-the-art face editing methods. Our method also performs well in downstream

tasks, such as text-driven facial image editing.

## 1. Introduction

Realistic face image generation and editing is a useful topic in image synthesis and is widely used in portrait generation and artistic creation. Many efforts [11–13] have been paid to improve the quality and increase the resolution of the generated face images. At the same time, users want to have more interaction with and control over the generated images. To increase the controllability of the generation process, many methods are proposed to edit the face images by different interfaces, such as sketches [4], texts [18], semantic masks [21], etc.

Benefiting from the implicit 3D representation of neural radiance fields (NeRF) [16], the image synthesis models have shown significant progress in transferring 2D image generation task [12] to 3D [2, 8, 17], addressing 3D consistency in perspective transformation. EG3D [2], StyleNeRF [8], and StyleSDF [17] use implicit three-dimensional representations to improve the quality of 3D face generation.

Recently, some NeRF-based face editing methods [10, 23, 24] have shown excellent results in decoupling the geometry and texture of faces. FENeRF [24], IDE-3D [23] and NeRFFaceEditing [10] decouple geometry and texture by using separate geometry and texture networks. These methods use the global latent code to generate global 3D representation, so editing the latent code will affect the whole face. This will inevitably affect non-editing regions when editing local facial regions, and even lead to inconsistent facial identities.

To improve the controllability of NeRF-based face editing, we propose a local controllable face generation and editing method, named LC-NeRF, for fine-grained facial local region control and the decoupling of geometry and texture. There are two core issues that need to be solved, one is the decomposition of the global 3D representation and representations of the local 3D regions, and another is the fusion of local 3D regions. It is challenging to decompose a complete 3D representation into multiple local 3D representations and stably complete the training process. To overcome this issue, we design our generator network with multiple local generators to generate the content for each local region. In addition, for more flexible control over geometry and texture, we further subdivide the local generator into a geometry network and a texture network controlled by geometry code and texture code separately. Through these designs, our method can modify the geometry and texture of local regions without affecting other regions by editing multiple local latent codes. Another core challenge is how to fuse local 3D representations of all local regions to generate the final face image. We propose a Spatial-Aware Fusion Module to complete the fusion of multiple local regions. Specifically, each local geometry generator predicts the semantic confidence of spatial points, and the fusion module fuses the features of different local generators in a soft and smooth way through all confidences.

Qualitative and quantitative experiments show that our method not only better achieves the stability of non-editing regions during editing, but also better ensures the consistency of face identities than state-of-the-art face editing methods. The main contributions of this paper are summarized as followed:

- We propose a local controllable NeRF face generation and editing method, named LC-NeRF, to control and edit the geometry or texture of local regions in a decoupled manner.
- We propose a *Local Region Generators Module* to decompose the global 3D representation and latent codes into multiple local regions, and a *Spatial-Aware Fusion Module* that aggregates these regions into a whole image.
- Our method achieves state-of-the-art 3D local geom-

etry and texture editing results for face editing, as demonstrated by both qualitative and quantitative evaluations.

## 2. Related Work

### 2.1. Neural Face Generation

Generative models, such as Stylegan v1-v3 [11–13], have achieved high-quality generation of 2D images. In recent years, NeRF [7] has emerged as a method that can implicitly model 3D geometry from 2D images and then render photorealistic and 3D consistent images. Subsequently, NeRF-based face generative models have been investigated. PI-GAN [1] proposes a SIREN-based [22] implicit radiance field to generate 3D faces via sampled latent and positional encoding. Furthermore, due to the advantages of StyleGAN [12] in image generation, some methods [8, 17] based on StyleGAN can generate high resolution and quality images. StyleNeRF [8] provides a 3D GAN approach that fuses style-based generation with scene representation by neural radiance fields. StyleSDF [17] is similar, but incorporates an SDF-based 3D representation to ensure that images generated from different viewpoints have 3D geometric consistency. In addition, some methods study different forms of space representation. For example, EG3D [2] uses three projection planes (tri-plane) to represent the 3D space, generated by a backbone of StyleGAN. GRAM [5] proposes a radiance manifolds based generative model that divides the space into multi-manifolds. These methods improve the quality of generated images but lack the editability and controllability of geometry and texture. Our method enhances the effect of disentanglement of facial features while maintaining generative quality.

### 2.2. Neural Face Editing

With the high-quality generation of images, many portraits are generated by the generation models, as described above. Meanwhile, some methods [19] take the editing as an application. Editing tasks are no longer limited to the 2D domain, and research on how to perform editing and control on 3D faces becomes popular. In the image domain, Faceshop [19] treats the face editing task as sketch-based image completion that can only edit the facial geometry. The demands for face editing are no longer editing geometry but also modifying texture, such as editing hair colors. DeepFaceEditing [4] decouples facial local regions by using sketches to represent geometry. SofGAN [3] trains a semantic occupancy field (SOF) and uses 2D semantic masks to generate face images to decouple geometry and texture. SemanticStyleGAN [21] enhances the control over local regions by generating the features of each region separately and then fusing the features of different regions in the 2D feature domain. The implicit 3D representation and gen-

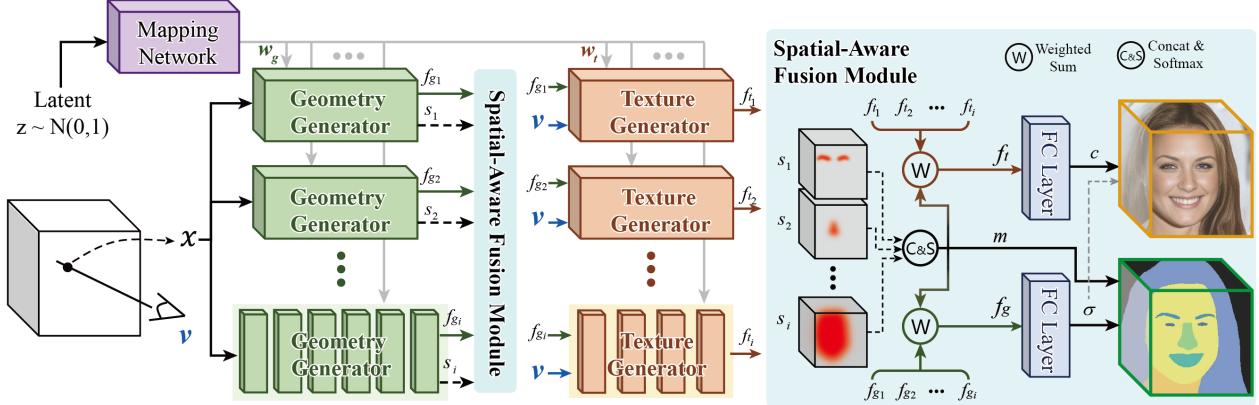


Figure 2. Pipeline of our framework LC-NeRF. Our pipeline is composed of multiple local generators and a spatial aware fusion module. The local generators include geometry and texture generators, separately controlled by geometry latent code  $w_g$  and texture latent code  $w_t$ . LC-NeRF can modify the geometry or texture of an local region directly by editing its latent code  $w_g$  or  $w_t$ .

eration of high-quality multi-view images in NeRF inspire works on 3D face decoupling and editing.

FENeRF [24] adds a mask branch to PI-GAN [1] for geometry control. Further, IDE-3D [23] and NeRFFaceEditing [10] realize the decoupled control of geometry and texture based on three projection planes [2]. IDE-3D [23] proposes a framework with separate geometry and texture networks to generate respective tri-plane features. Inspired by AdaIN, NeRFFaceEditing [10] decomposes the tri-plane features into geometry features and appearance features for decoupling the geometry and appearance. These methods are all implemented by optimizing the latent code during geometry editing, which is used to generate the whole face. Therefore, the global effect is prone to be affected during local editing.

### 3. Methodology

In this section, we introduce the architecture of our method in detail. We aim to control and edit local regions by editing the local geometry and texture latent codes. To achieve this goal, we need to solve two core problems: i) How to control the geometry and texture of each local region separately; ii) How to fuse the features of all the local regions into the global feature and generate a whole face image. For the first problem, we propose two independent lightweight local networks for each region: a geometry and a texture network, controlled by their respective geometry and texture latent codes (Section 3.1). For the second problem, we design a spatial aware fusion module to fuse the features generated by all the local networks and then generate the final face image (Section 3.2). We introduce two discriminators and detailed loss functions used in network training (Section 3.3). Then, we will introduce how to encode the real image to the latent code through GAN inversion and how to perform mask editing (Section 3.4).

#### 3.1. Local Region Generators

**Geometry Generator** The geometry generator is designed to determine the shape of the face. We assign a lightweight geometry generator  $\Phi s_i$  for each local region  $i$  of the face. If a 3D point belongs to a certain local region, the corresponding geometry generator provides the most information for this point. The generator plays a major role in determining the semantic category and geometry information of the point. As shown in Figure 2, each geometry generator contains 6 linear layers with SIREN [22] activation, and is controlled by the geometry latent code  $w_g$ .

Given a sampled point  $x \in \mathbb{R}^3$ , the  $i$ th geometry generator module  $\Phi s_i$  decodes it to obtain the semantic confidence  $s_i(x)$  and geometry feature  $f_{g_i}(x)$  from a geometry latent  $w_{g_i}$ :

$$s_i(x), f_{g_i}(x) = \Phi s_i(x, w_{g_i}) \quad (1)$$

Here,  $s_i(x)$  indicates the probability that the  $i$ th local geometry generator believes three-dimensional point  $x$  to be in its region.  $s_i(x)$  has two characteristics: i) The larger the value of  $s_i(x)$ , the more importance and more proportion the features of this generator acquire in the subsequent fusion module; ii) Sampling or modifying the geometry latent  $w_{g_i}$  can increase or reduce the  $s_i(x)$  value of the local region  $i$ , which enables local editing of geometry. Specifically, we use a linear layer following the geometry feature  $f_{g_i}(x)$  to calculate the geometry confidence  $s_i(x)$ .

**Texture Generator** The texture generator can be interpreted as a shader, which is used to fill the color of the geometry generated by the geometry generator. In other words, the texture generators do not participate in or affect the generation of geometry, and the geometry generator is only used to determine the shape of the face, so as to

achieve local region decoupling and geometry/texture decoupling. Each texture generator contains 4 linear layers with SIREN [22] activation, and is controlled by the texture latent code  $w_t$ .

Given the viewing direction  $v \in \mathbb{R}^3$ , the  $i_{th}$  local texture generator module  $\Phi_{t_i}$  decodes the texture feature  $f_{t_i}(x)$  from a geometry latent  $w_{t_i}$ :

$$f_{t_i}(x) = \Phi_{t_i}(f_{g_i}(x), v, w_{t_i}) \quad (2)$$

The texture features predicted by all the local texture generators will be fused in subsequent fusion module, and the final color value of the sampled 3D point  $x$  will be predicted.

### 3.2. Spatial-Aware Fusion Module

The spatial-aware fusion module is designed for interaction and aggregation among multiple local generators. The proposed fusion module fuses the features of different generators with a soft and adjustable mechanism and generates the whole image. We concatenate the semantic confidence  $s_i(x)$  of all the geometry generators and apply the softmax activation to obtain the semantic mask  $m(x)$ .

$$m_i(x) = \frac{e^{s_i(x)}}{\sum_{k=1}^K e^{s_k(x)}} \quad (3)$$

where  $K$  is the number of local regions. We use the semantic mask  $m(x)$  to fuse the geometry features  $f_{g_i}(x)$  to get the final geometry feature  $f_g(x)$ .

$$f_g(x) = \sum_i (m_i(x) * f_{g_i}(x)) \quad (4)$$

$f_g(x)$  is the geometry feature of the 3D point extracted by our proposed geometry generators. We use a linear layer after  $f_g(x)$  to predict the signed distance field (SDF) value  $d(x)$  of the 3D point  $x$ . Then we convert the SDF value to volume density  $\sigma(x)$  by the following formula [17].

$$\sigma(x) = \text{Sigmoid}(d(x)/\beta)/\beta \quad (5)$$

where  $\beta$  is a learnable parameter. The smaller  $\beta$  is, the more the volume density  $\sigma(x)$  will converge on the surface of the face. In our experiments, the initial value of  $\sigma(x)$  is set to 0.1. As the training progresses, the value of  $\beta$  will become smaller and smaller.

The texture features  $f_{t_i}(x)$  are also fused with the semantic mask  $m(x)$  to get the final texture feature  $f_t(x)$ . And then we use one linear layer after  $f_t(x)$  to get the color value  $c(x)$ :

$$f_t(x) = \sum_i (m_i(x) * f_{t_i}(x)) \quad (6)$$

We render the generated image  $I'$  and the generated semantic mask  $M'$  through the volume rendering. Given a

camera position  $o$ , by shooting a ray  $r(t) = o + tv$  at each pixel, we calculate the color and mask of  $N$  points sampled from  $t_n$  to  $t_f$  on the ray. In our experiments,  $N$  is set to 18.

$$\begin{aligned} I'(r) &= \int_{t_n}^{t_f} T(t) \sigma(r(t)) c(r(t), v) dt, \\ M'(r) &= \int_{t_n}^{t_f} T(t) \sigma(r(t)) m(r(t), v) dt, \\ \text{where } T(t) &= \exp \left( - \int_{t_n}^t \sigma(r(s)) ds \right) \end{aligned} \quad (7)$$

At this point, we complete the fusion operation through spatial aware fusion module to generate the whole image  $I'$  and the semantic mask  $M'$  with a super resolution model [26].

### 3.3. Discriminators and Loss Function

In order to ensure quality of the generated image and correspondence between the image and the mask, we propose a double discriminator supervision strategy. One discriminator is the image quality and pose aware discriminator  $D_I$ , which is used to distinguish between real images and generated images and predicts the azimuth and the elevation  $\theta'$ . In addition to the GAN loss [6], we use a smoothed L1 loss  $L_{pose}$  and R1 regularization loss to supervise the training of  $D_I$  for the generated images.

$$\begin{aligned} L_{D_I} &= \mathbb{E}[1 + \exp(D_I(I'))] + \mathbb{E}[1 + \exp(-D_I(I))] \\ &\quad + \lambda_{I_{reg}} \mathbb{E}\|\nabla D_I(I)\|^2 + \lambda_{pose} L_{pose}(\theta, \theta') \end{aligned} \quad (8)$$

where  $I'$  and  $M'$  are the fake image and the semantic mask generated by LC-NeRF with the sampled pose  $\theta$ .  $I$  and  $M$  are the ground truth image and the mask sampled from the real dataset. where  $\lambda_{I_{reg}}$ ,  $\lambda_{pose}$  are set to 10 and 15 respectively.

The other discriminator is the image and semantic mask discriminator  $D_{IM}$ , which is used to determine whether the image is consistent with the semantic mask. We also regularize the gradient norm for this discriminator with R1 regularization loss.

$$\begin{aligned} L_{D_{IM}} &= \mathbb{E}[1 + \exp(D_{IM}(I', M'))] \\ &\quad + \mathbb{E}[1 + \exp(-D_{IM}(I, M))] \\ &\quad + \lambda_{IM_{reg}} \mathbb{E}\|\nabla D_{IM}(I, M)\|^2 \end{aligned} \quad (9)$$

where  $\lambda_{IM_{reg}}$  is set to 10.

The generator  $G$  is supervised by the two discriminators  $D_M$  and  $D_{IM}$  and the camera pose loss  $L_{pose}$ . In addition, we also introduce geometry supervision of SDF with eikonal loss [7] and minimal surface loss [17].

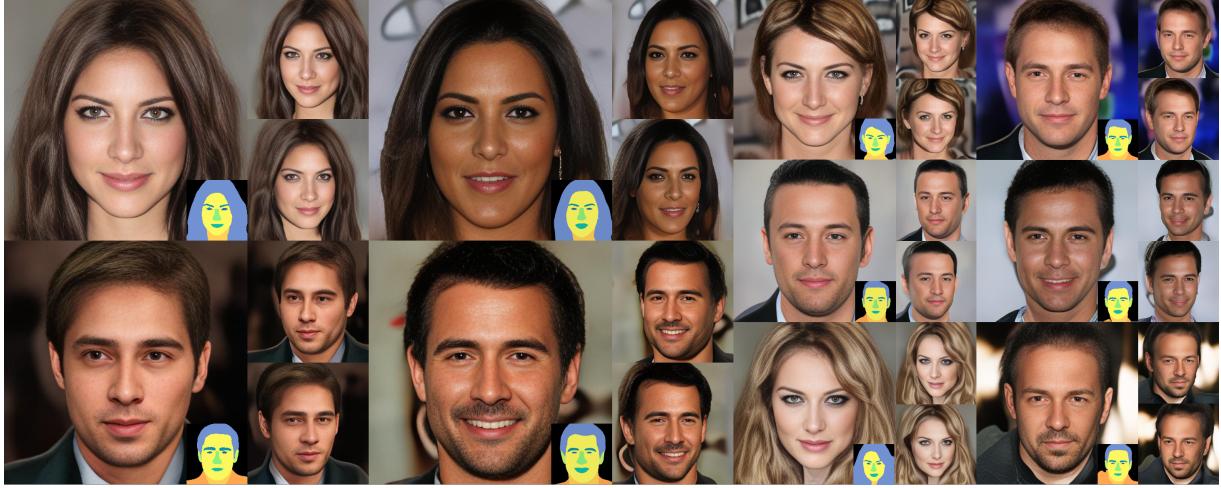


Figure 3. Multi-view face images and semantic masks with a resolution of 512, generated by LC-NeRF trained on CelebAMask-HQ dataset.

$$\begin{aligned}
 L_G = & \mathbb{E}[1 + \exp(-D_I(I'))] \\
 & + \lambda_{IM} \mathbb{E}[1 + \exp(-D_{IM}(I', M'))] \\
 & + \lambda_{pose} L_{pose}(\theta, \theta') + \lambda_{eik} \mathbb{E}[\|\nabla d(x)\|_2 - 1]^2 \\
 & + \lambda_{sur} \mathbb{E}[\exp(-100|d(x)|)]
 \end{aligned} \quad (10)$$

where  $\lambda_{IM}$ ,  $\lambda_{pose}$ ,  $\lambda_{eik}$ ,  $\lambda_{sur}$  are set to 0.5, 15, 0.1, 0.05 respectively.

### 3.4. Inversion and Editing

We can edit the images generated by the latent codes  $w_g$  and  $w_s$  at a certain pose as well as the real images. To edit the real images, we need to encode the real images into the  $W^+$  [12] space through pivotal tuning inversion [20]. Given a real face image  $I$  and the corresponding semantic mask  $M$ , we first invert  $I$  to generate the latent code  $w$ . When the user edits the mask and gets the edited mask  $M_e$ , our optimization goal is to find an editing vector  $\delta w$  to make the mask  $M'$  generated by  $\delta w + w$  close to the editing mask  $M_e$ . We use the mean square error (MSE) between the edited mask  $M_e$  and generated mask  $M'$ . During editing, we optimize the geometry latent code of the corresponding local region for 500 iterations.

## 4. Experiments

In this section, we first introduce our experimental setup, then discuss the generation and comparison results. We present the results of multi-view generation and style transfer of local or global regions. We also discuss the comparison results with state-of-the-art face editing methods, including FENeRF [24], IDE-3D [23] and NeRFFaceEditing (NeRFFE) [10], to show the superior effectiveness of our LC-NeRF.

	FENeRF	IDE-3D	NeRFFE	Ours
Hair	0.0332	0.0410	0.0277	<b>0.0239</b>
Eyebrow	0.0368	0.0668	0.0188	<b>0.0068</b>
Nose	0.0495	0.0538	0.0163	<b>0.0078</b>
Mouth	0.0539	0.0666	0.0208	<b>0.0112</b>
Average	0.0433	0.0570	0.0209	<b>0.0124</b>

Table 1. Quantitative metrics of pixel level difference in non-editing region for local geometry editing on different local regions. The difference visualization results on the whole image are shown in Figure 6.

**Training datasets** We train LC-NeRF on the CelebAMask-HQ dataset [15], which contains 30,000 high-quality face images with  $1024 \times 1024$  resolution. For this dataset, each image provides an accurate segmentation image with 19 categories. In our experiments, we combine the left and right local regions into one, such as glasses and eyebrows. After processing, there are 13 types of face local regions.

**Implementation Details** We use the Adam [14] optimizer with  $\beta_1 = 0$  and  $\beta_2 = 0.9$  to train the generator and discriminators, and the learning rates of  $G$ ,  $D_I$ ,  $D_{IM}$  are 0.00002, 0.0002 and 0.0002 respectively. We train LC-NeRF on 8 NVIDIA GeForce 3090 GPUs for 48 hours with a batch size of 24. During inference, it takes 0.1s to generate a face image and corresponding semantic mask on 1 NVIDIA GeForce 3090 GPU. LC-NeRF is implemented on Jittor [9], which is a fast-training deep learning framework, especially in generating networks [27].

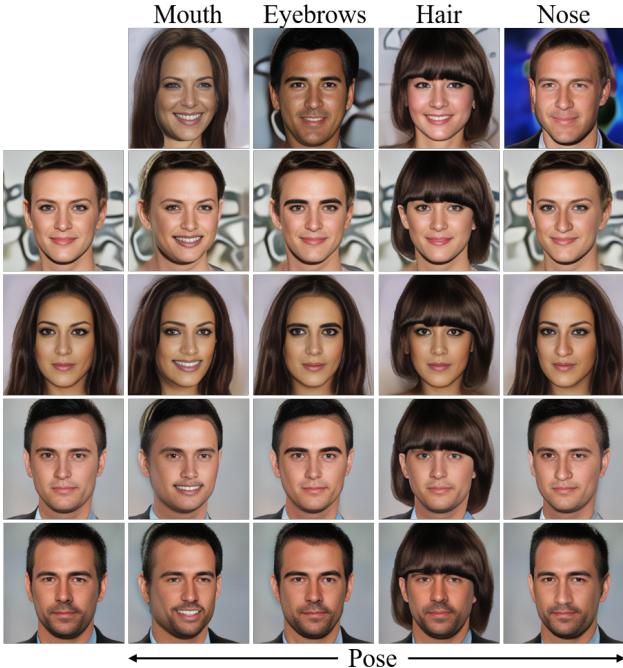


Figure 4. Results of local style transfer. LC-NeRF supports transferring the geometry and texture of any local region of other faces to the target face. Here we show the results of multi view synthesis that migrate the geometry and texture of an local region at the same time.

#### 4.1. Generation Results

In our framework, we can sample random latent codes to generate both face images and semantic masks. As shown in Figure 3, our method can generate diverse face images, and the multi-view results prove that our method maintains the 3D consistency across different views. In addition, we show the local and global style transfer effects of LC-NeRF.

**Local style Transfer** We can transfer the geometry and texture of local regions. For any given face image, we can modify the geometry of specific regions. We directly modify the geometry latent code  $w_g$  of an local region to complete local geometry editing. Figure 4 shows the multi view editing results of modifying mouth, eyebrows, hair, nose and. It can be observed that LC-NeRF can edit target regions accurately without affecting non-editing regions.

**Global style Transfer** We can transfer the geometry and texture of the face globally. We can modify the global texture of all the local regions while keeping the geometry unchanged. The global texture can be edited by directly modifying the texture latent codes  $w_t$  of all the local regions. The same goes for modifying global geometry. In Figure 5, we show an example of transferring styles of reference images

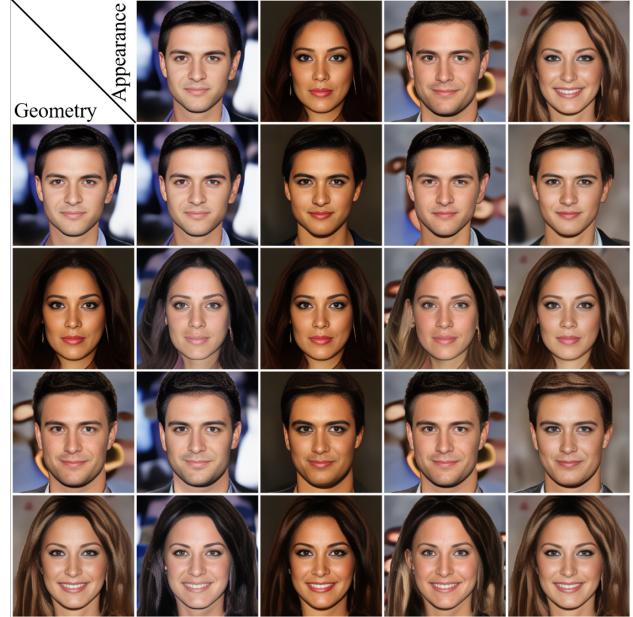


Figure 5. Results of global style transfer. LC-NeRF supports global modification of face texture. The figure shows examples of transferring the texture information of the reference face to the target face.

to target images. It can be observed that the geometry of all the local regions can remain unchanged when the texture is modified, which also verifies the decoupling property of geometry and texture.

#### 4.2. Evaluation

The most important quality of face editing is to change the target region while ensuring that the non-editing regions are not affected. Otherwise, the edited image may become too dissimilar to the original that it may be interpreted as another person entirely. For fair comparison, all evaluated methods are tested on the II2S [28] dataset without any fine-tuning. The II2S dataset contains 120 high-quality face images with different styles. We use a pretrained face parsing method [25] to extract the same semantic mask for all methods.

**Real Image Local Geometry Editing** Local geometry editing is an interactive and practical application of editing face images by modifying corresponding masks. In comparison, we appropriately increase the learning rate of IDE-3D inversion to ensure that it converges to the best effect. The inversion and local editing results are shown in Figure 6. Thanks to its local decoupling characteristics, our method LC-NeRF makes sure that the target regions are modified appropriately, while non-editing regions are not affected. On the other hand, the editing results of FENeRF appear un-

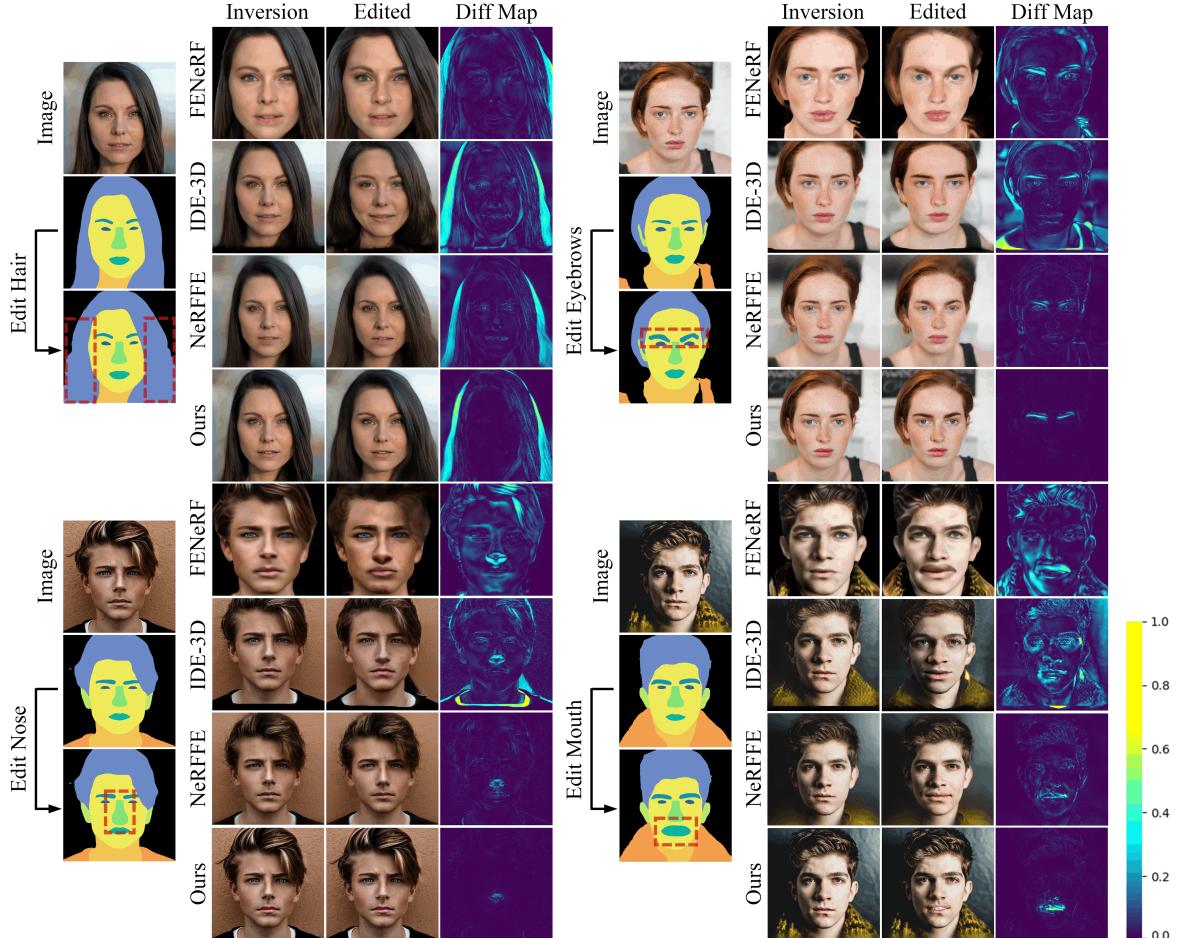


Figure 6. Comparison of local geometry editing with FENeRF [24], IDE-3D [23], NeRFFE [10]. For each sample, the left side displays the real image, source and edited mask in order from top to bottom. The right side is inversion results, edited results and difference maps of different methods.

natural and unrealistic. Moreover, since IDE-3D and NeRF-FaceEditing edit images in the global latent space, the edits inevitably also affect non-editing regions, resulting in obvious changes outside of the target region. In the case of editing the mouth, there are obvious modifications made to other regions of the face in the IDE-3D results. FENERF is limited in hair editing because the face occupies most of the image area during inversion. Because NeRFFaceEditing uses VGG loss to explicitly constrain the image invariance of non-editing regions in local editing optimization, there is a relatively good consistency of non-editing regions. However, our method can achieve the best results without such explicit constraints when editing local regions.

We use quantitative metric, i.e., pixel error of non-editing region to evaluate the effectiveness of editing. The pixel error maps  $\text{abs}(I - I_e)$  of the source image  $I$  and the edited image  $I_e$  for each editing operation are also visualized in Figure 6. The average values of error maps of

non-editing region edited locally by different local regions are shown in Table 2. It can be seen that LC-NeRF has the highest editing fidelity with lowest image difference value. We also conduct a usability study to evaluate image quality, editing accuracy, and the consistency of non-editing regions. Please refer to the supplementary for more details.

**Real Image Local Texture Editing** Local texture editing allows users to modify the texture of a local region, which emphasizes naturalness and harmony. FENeRF and NeRFFaceEditing are designed for local geometry editing and global texture editing, and do not support local texture editing. So here we compare the local texture editing results with IDE-3D. IDE-3D achieves local texture editing through extracting features from the two triplane features and combining them according to a mask to generate a new face image. This approach makes the generated images unnatural and there is a sense of splicing between different re-

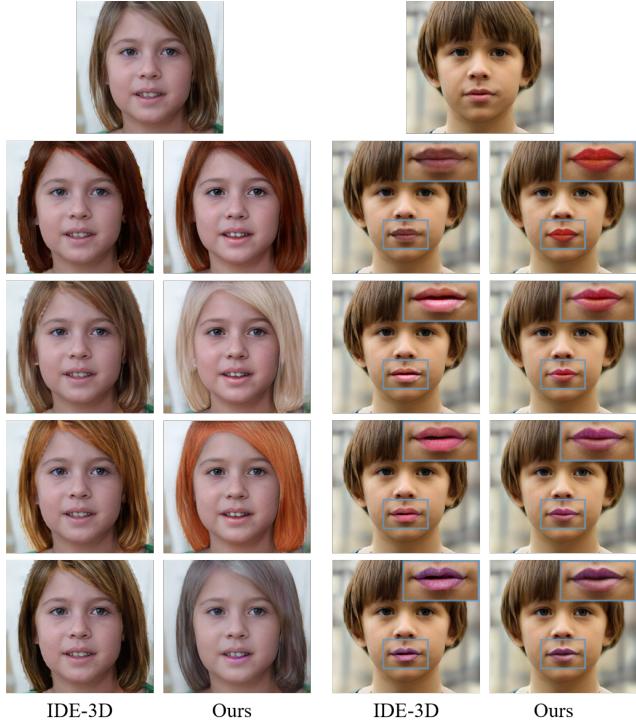


Figure 7. Qualitative comparison results of real image local texture editing between IDE-3D and LC-NeRF(Ours). Two cases show the editing effects of two methods to modify different hair and lip texture colors.

gions. LC-NeRF can directly change the texture latent code  $w_t$  of the certain local region and then fuse the edited high-dimensional texture features, so that the generated image is more natural and controllable. Because authors of IDE-3D has not released their code for local texture editing, we invert the texture editing images given in their papers, which are not real human face images, but images generated by IDE-3D. Local texture editing results are shown in Figure 7. It can be seen that IDE-3D hair texture editing results have a strong sense of border and contain jagged parts. In addition, after editing the mouth texture with IDE-3D, the geometry of the mouth is changed. Some of the results contain closed mouths, while others show open mouths. This shows that IDE-3D does not achieve effective decoupling of geometry and texture. At the same time, the edited mouth texture is unnatural and foggy and even spreads to non-mouth regions. The images edited by LC-NeRF are more natural and controllable, benefiting from our proposed local generators and high-dimensional feature fusion mechanism.

#### 4.3. Text-Driven Face Editing

Text-driven face editing allows users to edit face directly using text, which is an effective and convenient way of editing. Therefore, we also explore applicaiton of LC-NeRF in text-driven image editing. We used StyleCLIP [18] with

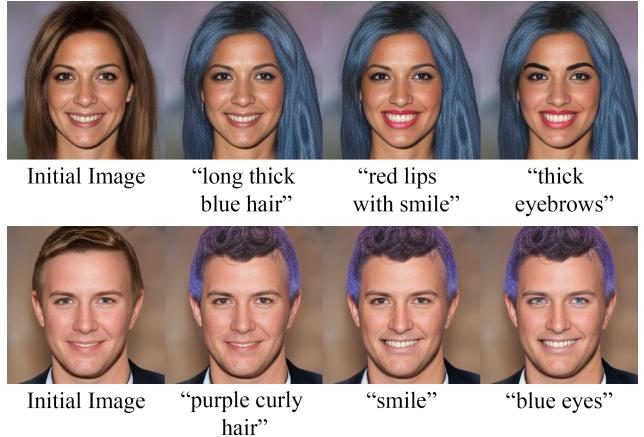


Figure 8. Results of text-driven face editing. Give an initial image (left), LC-NeRF can edit it directly through the text. The figure shows the results of multiple local region edits, accumulative from left to right.

ViT-B/32 pretrained model for text guided latent manipulation. The driving text can directly optimize the  $\mathcal{W}^+$  space latent code. In our experiments, generated images are controlled by short text clips, such as "thick eyebrows" and "red lips with smile", using CLIP loss [18]. We present sample edited images with corresponding prompt texts, with 100-300 latent optimizaiton steps, in Figure 8. In each line of Figure 8, editing results are accumulated, with each image using the optmized latent from the previous image as a starting point. The result shows that LC-NeRF allows for fine-grained control of facial features and accurate editing driven by text, enabling text-based editing of one facial feature without affecting other regions.

## 5. Conclusions, Limitations, and Future Work

We propose LC-NeRF, a local controllable and editable face generation method, which can generate view-consistent face images and semantic masks. Compared with the previous state-of-the-art face editing methods, LC-NeRF has achieved more fine-grained feature decoupling, including local region decoupling and decoupling of geometry and texture . Our method achieves the best performance in face editing, which ensures the stability of non-editing regions and the consistency of face identities. Our method supports local mask editing, local and global texture editing, and can easily be extended to downstream tasks, such as text editing.

The limitation of this work is that we can decouple the local regions and the geometry and texture, but we cannot control the local internal texture more finely, such as the hair texture, facial wrinkles, etc. In the future, how to control the content of local texture more finely will be one of our research directions.

## References

- [1] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *arXiv*, 2020. 2, 3
- [2] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks, 2021. 1, 2, 3
- [3] Anpei Chen, Ruiyang Liu, Ling Xie, Zhang Chen, Hao Su, and Jingyi Yu. Sofgan: A portrait image generator with dynamic styling. *ACM Transactions on Graphics (TOG)*, 41(1):1–26, 2022. 2
- [4] Shu-Yu Chen, Feng-Lin Liu, Yu-Kun Lai, Paul L. Rosin, Chunpeng Li, Hongbo Fu, and Lin Gao. Deepfaceediting: Deep face generation and editing with disentangled geometry and appearance control. *ACM Trans. Graph.*, 40(4), jul 2021. 1, 2
- [5] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2
- [6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Commun. ACM*, page 139–144, oct 2020. 4
- [7] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099*, 2020. 2, 4
- [8] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenet: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021. 1, 2
- [9] Shi-Min Hu, Dun Liang, Guo-Ye Yang, Guo-Wei Yang, and Wen-Yang Zhou. Jittor: a novel deep learning framework with meta-operators and unified graph execution. *Science China Information Sciences*, 63(12):1–21, 2020. 5
- [10] Kaiwen Jiang, Shu-Yu Chen, Feng-Lin Liu, Hongbo Fu, and Gao Lin. Nerffaceediting: Disentangled face editing in neural radiance fields. In *ACM SIGGRAPH Asia 2022 Conference Proceedings*, 2022. 2, 3, 5, 7, 11, 12
- [11] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021. 1, 2
- [12] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1, 2, 5
- [13] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 1, 2
- [14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5
- [15] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 5
- [16] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1
- [17] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513, 2022. 1, 2, 4
- [18] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2085–2094, October 2021. 1, 8
- [19] Tiziano Portenier, Qiyang Hu, Attila Szabó, Siavash Arjomand Bigdeli, Paolo Favaro, and Matthias Zwicker. Faceshop: Deep sketch-based face image editing. *ACM Trans. Graph.*, 37(4), jul 2018. 2
- [20] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Trans. Graph.*, 2021. 5
- [21] Yichun Shi, Xiao Yang, Yangyue Wan, and Xiaohui Shen. Semanticstylegan: Learning compositional generative priors for controllable image synthesis and editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11254–11264, 2022. 1, 2
- [22] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 2, 3, 4
- [23] Jingxiang Sun, Xuan Wang, Yichun Shi, Lizhen Wang, Jue Wang, and Yebin Liu. Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis. *arXiv preprint arXiv:2205.15517*, 2022. 2, 3, 5, 7, 11, 12
- [24] Jingxiang Sun, Xuan Wang, Yong Zhang, Xiaoyu Li, Qi Zhang, Yebin Liu, and Jue Wang. Fenerf: Face editing in neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7672–7682, 2022. 2, 3, 5, 7, 11, 12
- [25] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. *International Journal of Computer Vision*, 129(11):3051–3068, 2021. 6, 11
- [26] Shangchen Zhou, Kelvin C.K. Chan, Chongyi Li, and Chen Change Loy. Towards robust blind face restoration with codebook lookup transformer. In *NeurIPS*, 2022. 4
- [27] Wen-Yang Zhou, Guo-Wei Yang, and Shi-Min Hu. Jittorgan: A fast-training generative adversarial network model

- zoo based on jitter. *Computational Visual Media*, 7(1):153–157, 2021. 5
- [28] Peihao Zhu, Rameen Abdal, Yipeng Qin, John Femiani, and Peter Wonka. Improved stylegan embedding: Where are the good latents?, 2020. 6

## A. Comparison of Real Image Geometry Editing

In this section, we show results of real image geometry editing of single or multiple regions with our method LC-NeRF and three state-of-the-art methods, namely FENeRF [24], IDE-3D [23], and NeRFFaceEditing (NeRFFE) [10]. The qualitative results are shown in Figure 10 and Figure 11. It can be seen that our method modifies the images more accurately.

We also use two quantitative methods to evaluate the editing results. One is the average Pixel Difference (PD) between non-editing regions of the original and the edited images to measure whether the editing affects these regions. The other is Mask Consistency (MC) between the editing masks and the masks extracted from the edited images with a pretrained face parser [25] to measure whether the methods achieve correct and adequate editing. The results are shown in the Table 2. LC-NeRF achieves the best performance.

	FENeRF	IDE-3D	NeRFFE	Ours
PD	0.0441	0.0550	0.0235	<b>0.0211</b>
MC	0.1826	0.1073	0.0529	<b>0.0306</b>

Table 2. Quantitative comparison of face mask editing of four methods.

## B. Perception Study

For the application of face editing, we conduct a perception study to evaluate the effectiveness. We show the input image, the corresponding source mask and the edited mask, and four edited results (including FENeRF [24], IDE-3D [23], NeRFFaceEditing (NeRFFE) [10], and Ours) for each editing sample. Results from all four methods are placed in random order. In this perception study, each participant needs to rank images on 17 examples (Figure 10 and Figure 11 in this material as well as Figure 6 in the draft) in the following three aspects respectively:

- **Image quality**, which measures the quality of face images generated by different methods;
- **Face consistency**, which measures whether the edited face is consistent with the input face, that is, whether the face identity changes before and after editing;
- **Editing effect**, which measures the correctness and the quality of editing results.

In total, 40 participants participated in our perception study, and we got  $40(\text{participants}) \times 17(\text{questions}) = 680$  subjective evaluation for each method. On average, each researcher spent 21.85 minutes on our survey. When computing the final score, the method ranked as first in each

evaluation result translates to a score of four, the second translates to a score of three, the third a score of two, and the last a score of one. The results are shown in the Figure 9 in the form of a boxplot. Our method achieves scores of 3.49, 3.51, and 3.27 in image quality, face consistency, and editing results respectively, exceeding the scores of FENeRF, IDE-3D, and NeRFFE. We also perform the ANOVA tests on the three aspects and get the F values for image quality ( $F = 252.39, p < 0.001$ ), face consistency ( $F = 215.42, p < 0.001$ ), and editing effect ( $F = 90.95, p < 0.001$ ). It can be clearly seen that our method has a significant improvement over the other three methods.

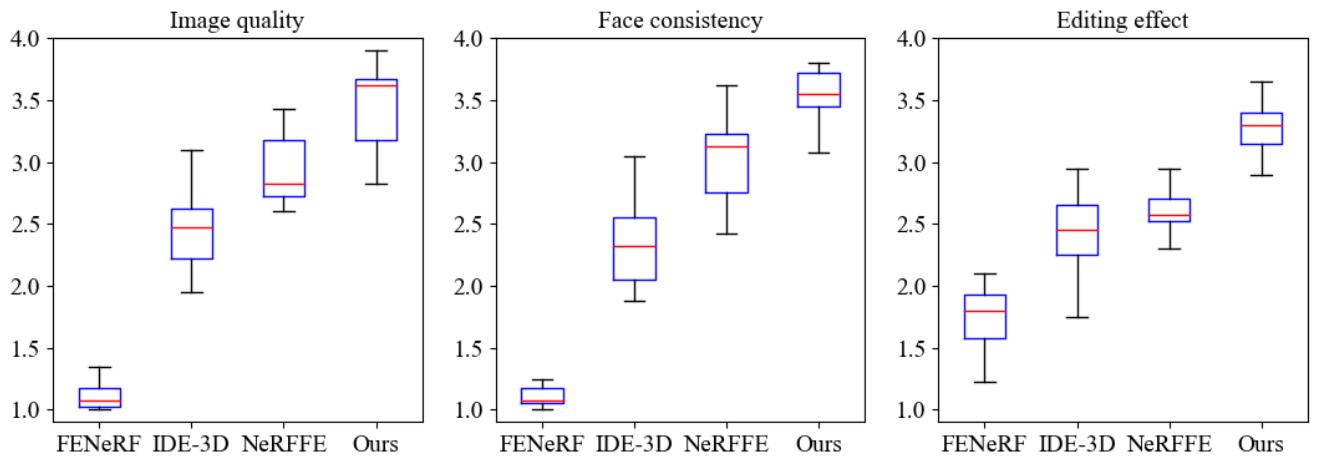


Figure 9. Box plots of image quality, face consistency, and editing effect, based on the participants in the perception study with four methods: FENeRF [24], IDE-3D [23], NeRFFE [10] and Ours.

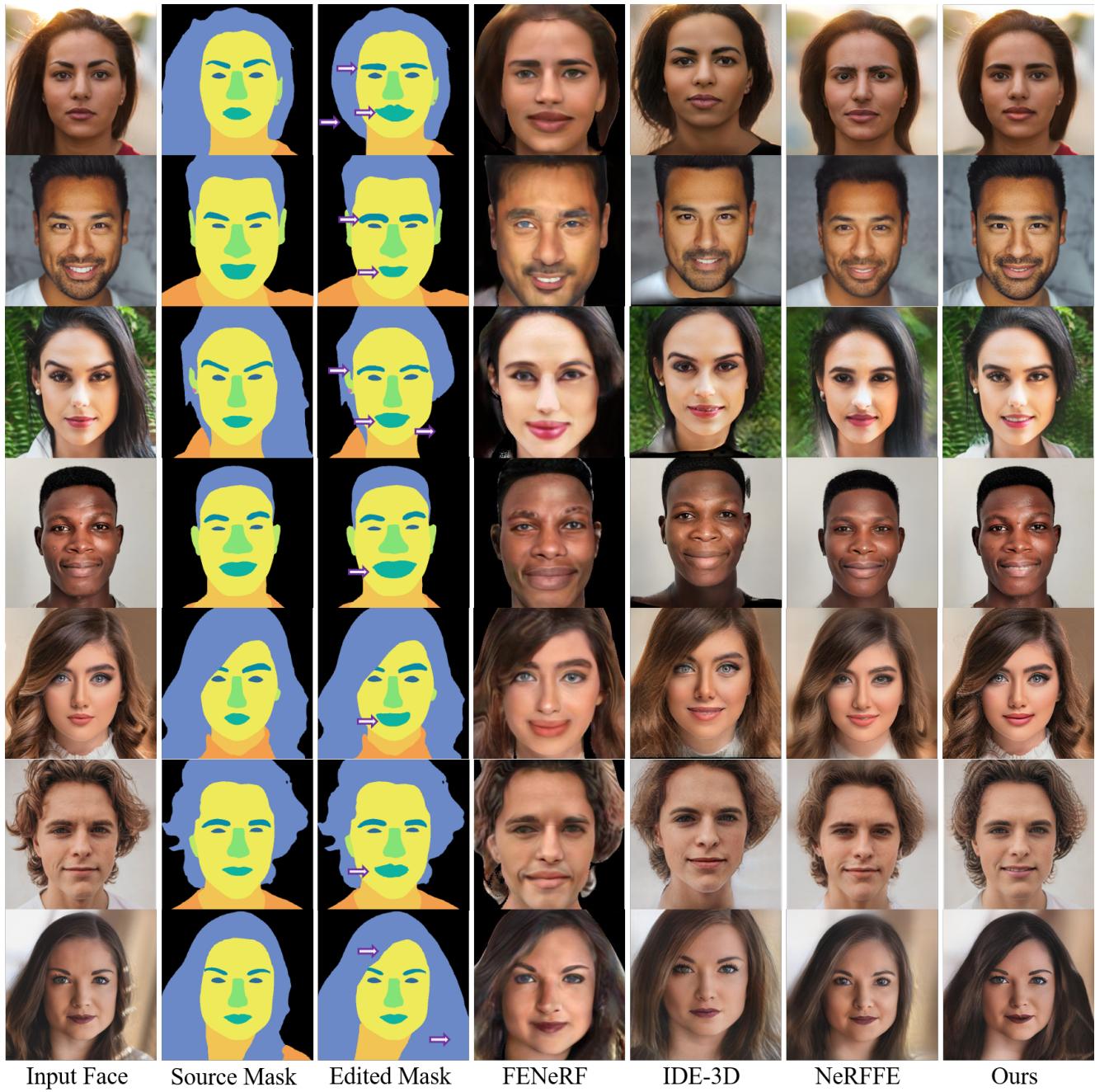


Figure 10. In each case, we show the input face, the source mask, the edited mask (the arrow marks the editing region), and the face editing results of the four methods.

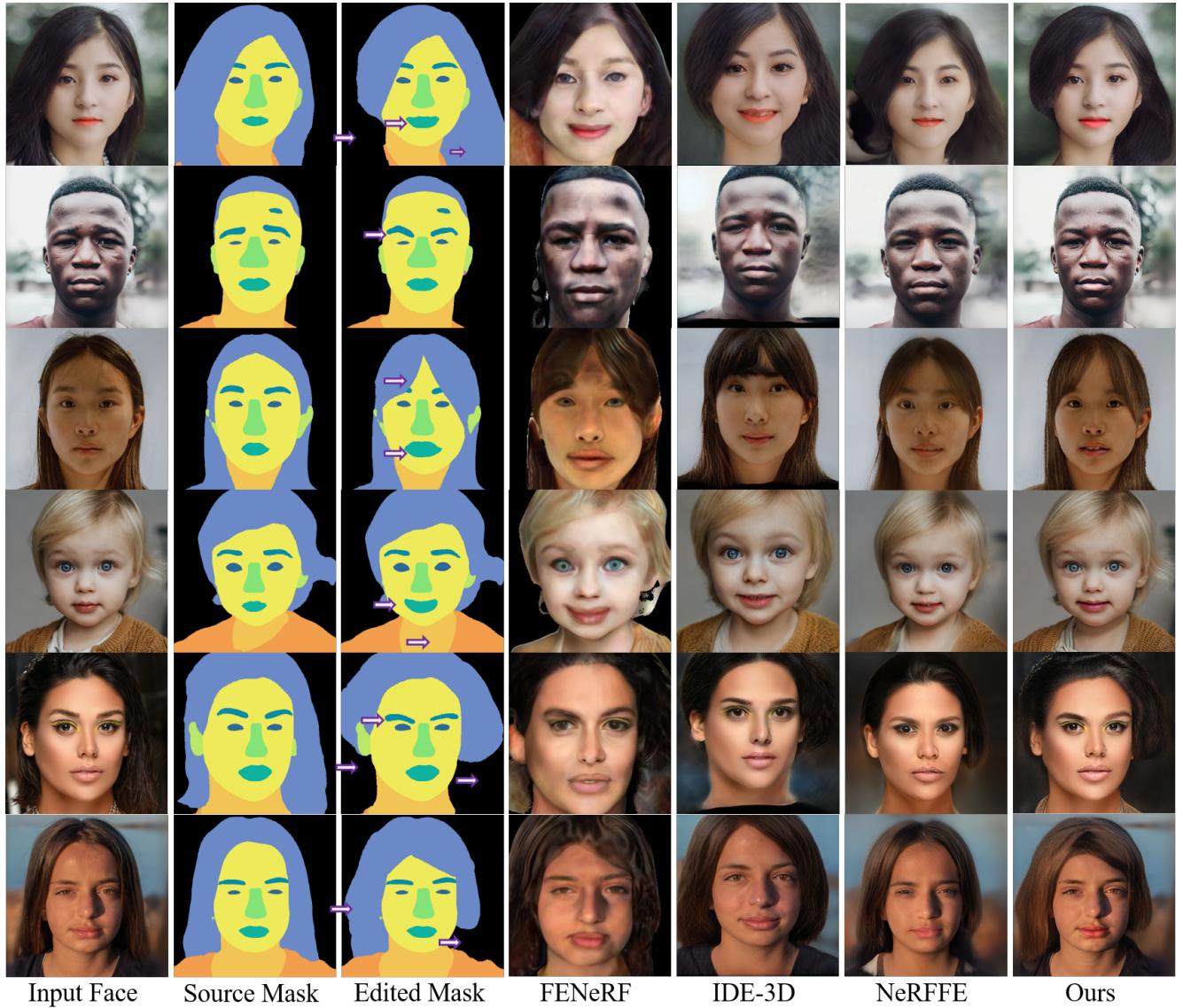


Figure 11. In each case, we show the input face, the source mask, the edited mask (the arrow marks the editing region), and the face editing results of the four methods.