# SemanticStyleGAN: Learning Compositional Generative Priors for Controllable Image Synthesis and Editing

Yichun Shi Xiao Yang Yangyue Wan Xiaohui Shen
{yichun.shi,yangxiao.0,wanyangyue,shenxiaohui.kevin}@bytedance.com
ByteDance Inc., USA

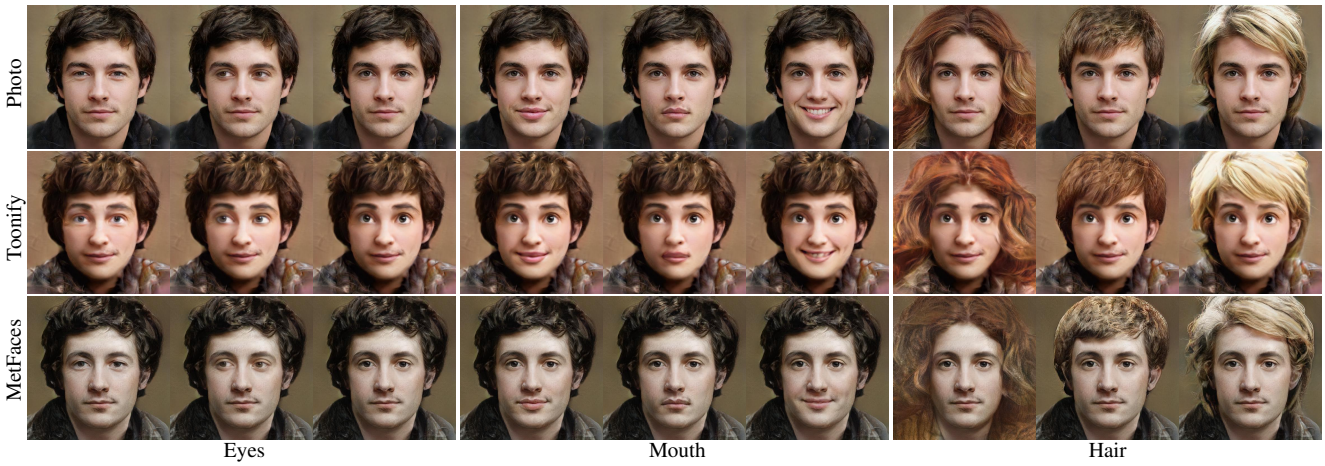


Figure 1. Synthesis results of SemanticStyleGAN. Our generator provides a fine-grained control over synthesized images by factorizing the latent space based on semantic areas. Thus, we can edit an area by simply replacing its latent code. Here, we sample three different latent codes for each area. Note that our models also disentangles shape and texture but we are simultaneously changing both here.

## Abstract

*Recent studies have shown that StyleGANs provide promising prior models for downstream tasks on image synthesis and editing. However, since the latent codes of StyleGANs are designed to control global styles, it is hard to achieve a fine-grained control over synthesized images. We present SemanticStyleGAN, where a generator is trained to model local semantic parts separately and synthesizes images in a compositional way. The structure and texture of different local parts are controlled by corresponding latent codes. Experimental results demonstrate that our model provides a strong disentanglement between different spatial areas. When combined with editing methods designed for StyleGANs, it can achieve a more fine-grained control to edit synthesized or real images. The model can also be extended to other domains via transfer learning. Thus, as a generic prior model with built-in disentanglement, it could facilitate the development of GAN-based applications and enable more potential downstream tasks*[1]

## 1. Introduction

Recent studies on Generative Adversarial Networks (GANs) have made impressive progress on image synthesis, where photo-realistic images can be generated from random codes in a latent space [11, 35–37]. These models provide powerful generative priors for downstream tasks by serving as neural renderers. However, their synthesis procedure is usually stochastic and no user control is naturally promised. Thus, it is still a challenging problem to achieve controllable image synthesis and editing utilizing generative priors.

One of the most famous work among such generative priors is the StyleGAN series [35–37], where each generated image is conditioned on a set of coarse-to-fine latent codes. However, the meanings of these latent codes are still relatively ambiguous. Thus, a plethora of studies have attempted to further investigate into the latent space of StyleGAN to improve controllability. It is shown that by learning a linear boundary or a neural network in the latent space of StyleGAN, one could control the global attributes [4, 26, 60, 61] or 3D structure [65] of the generated images. Furthermore, by using an optimization/encoder-based method, real images can also be embedded into the latent space to create a unified synthesis/editing model [2, 3, 5, 56, 66, 68, 76]. However, as pure learning-based methods, these solutions inevitability suffer from the biases in the StyleGAN latent space. For example, since different attributes could be correlated in StyleGAN, it often happens that unexpected attributes or local parts are changed while one wants to edit a

[1] Visit https://SemanticStyleGAN.github.io for more information.

certain attribute or area.

To obtain a more precise control, another solution is to train a new GAN model from scratch by introducing additional supervision or inductive biases. For example, by using 3D rendered faces, CONFIG [39] and DiscoFaceGAN [18] aim to build a GAN where pose, 3D information are factorized in the latent space. GAN-Control [62] disentangles the latent space by incorporating pre-trained attribute models for contrastive learning. Given the recent progress on neural rendering, it has also been shown that 3D-controllable GANs can be trained from images by injecting volumetric rendering into the synthesis procedure [13, 25, 47, 59, 75]. However, a major limitation of above-mentioned models is that they are designed for holistic attributes and there is no fine-grained local editability.

In this work, we propose SemanticStyleGAN, which introduces a new type of generative prior for controllable image synthesis. Unlike prior work, the latent space of SemanticStyleGAN is factorized based on semantic parts defined by semantic segmentation masks. Each semantic part is modulated individually with corresponding local latent codes and an image is synthesized by composing local feature maps. Different from layout-to-image translation methods [14, 70, 77], our local latent codes are able to control both the structure and texture of semantic parts. Compared to attribute-conditional GANs [18, 39, 62], our model is not designed for any specific task and can serve as a generic prior like StyleGAN. Thus, it can be combined with latent manipulation methods designed for StyleGAN to edit output images while providing more precise local controls. The contributions of this work can be summarized as follows:

- A GAN training framework that learns the joint modeling of image and semantic segmentation masks.
- A compositional generator architecture that disentangles the latent space into different semantic areas to control the structure and texture of local parts via implicit latent code manipulation.
- Experiments showing that our generator can be combined with existing latent manipulation methods to edit images in a more controllable fashion.
- Experiments showing that our generator can be adapted to other domains with only limited images while preserving spatial disentanglement.

## 2. Related Work

### 2.1. GAN Latent Space for Image Editing

Given the success of GANs on synthesizing high quality images [11, 36, 37], many studies have attempted to utilize GANs as a image prior to achieve controllable image synthesis and editing. These studies can be categorized into two types. The first type aims to learn a model to manipulate the latent space of a pre-trained GAN network to achieve editability. For example, InterFaceGAN [60], GANSpace [26] and StyleFlow [4] trains a attribute model in the StyleGAN latent space to control binary attributes. StyleRig [65] learns a set of latent space networks to change the pose and lighting. Similarly, StyleFusion [33] learns to fuse semantic parts from different images in the latent space. The second type aims to learn a GAN with more disentangled latent space using additional supervision. For example, CONFIG [39] and DiscoFaceGAN [18] uses 3D-rendered data to disentangle pose, identity, expression from other information. GAN-Control [62] separates attributes like identity and age in the latent space by utilizing pre-trained attribute models. Besides these, StyleMapGAN [38] propose to use style maps to modulate a synthesis network, but the meaning of each style pixel is unclear. Different from prior works, we propose a new type of factorization in the GAN latent space according to semantic labels. Our disentangled latent codes could independently control the shape and texture of each semantic part in the output image.

### 2.2. Compositional Image Synthesis

A plethora of studies have investigated how to build generative models to mimic the compositional nature of the world. To achieve compositionality, some studies propose to take images as input and compose a complicated scene with elements from real images [8, 10, 58]. On the other side, the majority studies aim to build a generative model that unsupervisedly discovered different objects in the training images and then synthesize them from independent latent codes. Most of these methods assume that objects are positioned independently in the scene and a compositional generative model is designed to discover such objects [6, 12, 19, 20, 23, 24, 31, 67, 72, 73]. Some other methods approach the compositional synthesis from a 3D perspective and disentangles objects and background by leraning multi-view datasets [28, 46, 48, 49]. Similar to these work, we inject composition as an inductive-bias to encourage disentanglement. However, we focus on semantic parts that are defined by humans. This allows us to decompose highly correlated local parts below object level (e.g. hair and face) and enables more fine-grained control during synthesis.

### 2.3. Layout-based Generators for Local Editing

In the layout-to-image translation problem, a layout image is provided as the condition for controllable image synthesis. The layout image can be a semantic segmentation mask [14, 15, 43, 50, 54, 69, 70, 77, 78], a sketch image [16, 56, 69], etc. Among these, some studies have attempted to represent different semantic parts with latent codes [14, 77, 78]. But since the layout is controlled by the input segmentation mask, they are only able to control the local texture. Our method also shares similarity with prior research that utilizes semantic masks as intermediate

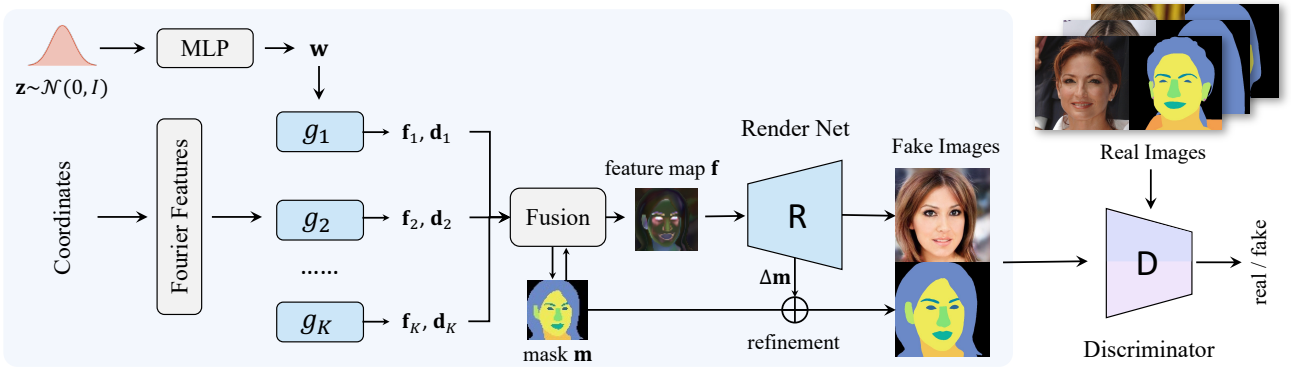


Figure 2. Overview of our training framework. A MLP first maps randomly sampled codes into $\mathcal{W}$ space. The $\mathbf{w}$ code is used to modulate the weights of local generators. Each local generator $g_k$ takes Fourier features as its input and outputs a feature map $\mathbf{f}_k$ and a pseudo-depth map $\mathbf{d}_k$, which are fused into a coarse segmentation mask $\mathbf{m}$ and a global feature map $\mathbf{f}$ for image synthesis. The render network $R$, which is only conditioned on the feature map, refines upsampled $\mathbf{m}$ into a high-resolution segmentation mask by learning a residual $\Delta\mathbf{m}$ and generates the fake image. A dual-branch discriminator models the joint distribution of RGB images and semantic segmentation masks.

representations for generation [9, 30, 32], but they are engineered to serve conditional generation tasks and not able to generate images from scratch. Recently, some researchers have also analyzed the correlation between StyleGAN style space and semantic masks [17, 33, 71] or supervise the latent manipulation with segmentation masks [21, 42, 52] to achieve local editing. In contrast to these methods, we build a semantic-aware generator that directly associates different local areas with latent codes, these codes can then be used to edit both local structure and texture.

## 3. Methodology

A typical GAN framework learns a generator that maps a vector $\mathbf{z} \sim \mathcal{Z}$ to an image, where $\mathcal{Z}$ is usually a standard normal distribution. In StyleGANs [36, 37], to handle the non-linearity of data distribution, $\mathbf{z}$ is first mapped into a latent code $\mathbf{w} \sim \mathcal{W}$ with an MLP. This $\mathcal{W}$ space is then extended into a $\mathcal{W}^+$ space that controls the output styles at different resolutions [36]. However, these latent codes do not have a strictly defined meaning and can hardly be used individually.

We propose to build a generator whose $\mathcal{W}^+$ space is disentangled for different semantic areas. Formally, given a labeled dataset $D = \{(x_1, y_1), (x_2, y_2)..., (x_n, y_n)\}$, where $y_i \in \{0, 1\}^{H \times W \times K}$ is the semantic segmentation mask of image $x_i$ and $K$ is the number of semantic classes, our generator gives a factorized $\mathcal{W}^+$ such that:

$$\mathcal{W}^+ = \mathcal{W}^{\text{base}} \times \mathcal{W}^1 \times \mathcal{W}^2 \times ... \times \mathcal{W}^K. \quad (1)$$

Here each local latent code $\mathbf{w}^k \in \mathcal{W}^k$ controls the shape and texture of $k_{th}$ semantic area defined in segmentation labels while $\mathbf{w}^{\text{base}} \in \mathcal{W}^{\text{base}}$ is a shared code that controls the coarse structure, such as pose. Each $\mathbf{w}^k$ is further decomposed into a shape code $\mathbf{w}_s^k$ and a texture code $\mathbf{w}_t^k$. The generator $G : \mathcal{W}^+ \to \mathcal{X} \times \mathcal{Y}$ maps the latent codes to an

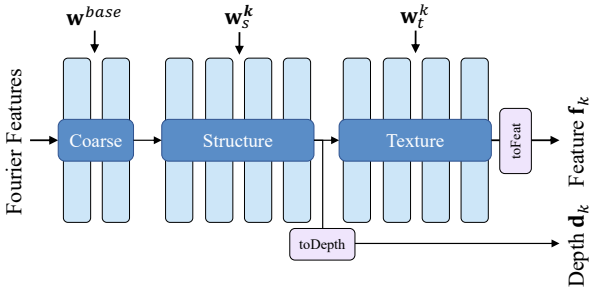


Figure 3. The architecture of local generator. Blue blocks are modulated 1×1 convolution layers whose weights are conditioned on input latent codes. Purple blocks are linear transformation layers.

RGB image and a semantic segmentation mask. To this end, we identify two major challenges:

1. How to decouple different local areas?
2. How to ensure the semantic meanings of these areas?

For the first problem, inspired by compositional generative models [12, 23, 49], we introduce local generators and a compositional synthesis procedure as the inductive bias. For the second problem, we use a dual-branch discriminator $D : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that models the joint distribution $p(x, y)$ to supervise the shapes of local parts after composition.

### 3.1. Generator

The overall structure of our generator is shown in Figure 2. Similar to StyleGAN2 [36, 37], an 8-layer MLP first maps $\mathbf{z}$ to the intermediate code $\mathbf{w}$. Then, $K$ local generators are introduced to model different semantic parts using $\mathbf{w}$. A render net $R$ takes in the fused results from local generators and outputs an RGB image and a corresponding semantic segmentation mask.

**Local Generator** Following recent work on continuous image rendering [7, 63, 75], we use modulated MLPs for

local generators (Fig. 3), which allows explicit spatial control over synthesized output. Given Fourier features [64] (position encoding) $\mathbf{p}$ and latent codes as inputs, a local generator $g_k$ outputs a feature map $\mathbf{f}_k$ and a pseudo-depth map $\mathbf{d}_k$:

$$g_k : (\mathbf{p}, \mathbf{w}^{\text{base}}, \mathbf{w}_s^k, \mathbf{w}_t^k) \mapsto (\mathbf{f}_k, \mathbf{d}_k). \quad (2)$$

Here, to reduce the computation cost, the input Fourier feature map as well as the outputs are of a reduced size $H^c \times W^c$, smaller than the final output image. In practice, we choose it to be $64 \times 64$ to balance the efficiency and quality. During training, style mixing [36] is conducted independently within each local generator between $\mathbf{w}^{\text{base}}$, $\mathbf{w}_s^k$ and $\mathbf{w}_t^k$ such that different local parts and different shapes and textures could work collaboratively for synthesis.

**Fusion** In the fusion step, we first generate a coarse segmentation mask $\mathbf{m} \in \mathbb{R}^{K \times H^c \times W^c}$ from pseudo-depth maps. Following prior work on compositional generation [12, 23], the pseudo-depth maps are used as logits for softmax function:

$$\mathbf{m}_k(i,j) = \frac{\exp(\mathbf{d}_k(i,j))}{\sum_{k'}^{K} \exp(\mathbf{d}_{k'}(i,j))}, \quad (3)$$

where $\mathbf{m}_k(i,j)$ denotes the pixel $(i,j)$ in the $k_{th}$ class of mask $\mathbf{m}$ and similarly for $\mathbf{d}_k(i,j)$. The feature maps are then aggregated by:

$$\mathbf{f} = \sum_{k=1}^{K} \mathbf{m}_k \odot \mathbf{f}_k. \quad (4)$$

Here $\odot$ denotes element-wise multiplication. The aggregated feature map $\mathbf{f}$ contains all the information about the output image and is sent into $R$ for rendering. We note that directly using $\mathbf{m}$ for feature aggregation could be problematic when some classes are transparent. Thus, we use a modified version $\tilde{\mathbf{m}}$ for feature aggregation in case of transparent classes, *e.g.* glasses (See appendix for details).

**Render Net** The render net $R$ is similar to the original StyleGAN2 generator with a few modifications. First, it does not use modulated convolution layers and the output is purely conditioned on the input feature map . Second, we input the feature map at both $16\times16$ and $64\times64$ resolutions, where feature concatenation is conducted at $64 \times 64$. The additional input of low-resolution feature map allows a better blending between different parts. Last, we find that directly training with $\mathbf{m}$ is difficult due to the intrinsic gap between softmax outputs and real segmentation masks. Thus, besides the ToRGB branch after each convolution layer, we have an additional ToSeg branch as in SemanticGAN [41] to output residuals to refine the coarse segmentation mask $\mathbf{m}$ into the final mask $\hat{y} = upsample(\mathbf{m}) + \Delta\mathbf{m}$ that has the same size as output image. Here a regularization loss is needed such that the final mask would not deviate too much from the coarse mask:

$$\mathcal{L}_{mask} = \|\Delta\mathbf{m}\|^2. \quad (5)$$

| Method | Data | Compositional | FID↓ | IS↑ |
|---|---|---|---|---|
| StyleGAN2 | img | ✗ | 4.45 | 3.40 |
| SemanticGAN | img&seg | ✗ | 18.54 | 2.77 |
| + proposed training | img&seg | ✗ | 7.50 | 3.51 |
| SemanticStyleGAN (ours) | img&seg | ✓ | 6.42 | 3.21 |

Table 1. Quantitative evaluation on synthesis quality. All the models are trained on CelebAMask-HQ at 256×256. "img" and "seg" refer to RGB image and segmentation mask, respectively.

### 3.2. Discriminator and Learning Framework

In order to model the joint distribution $p(x, y)$, the discriminator needs to take both RGB images and segmentation masks as input. We found that a simple conconcatenation does not work due to the large gradient magnitude on segmentation masks. Thus, we propose to use a dual-branch discriminator $D(x, y)$ that has two convolution branches for $x$ and $y$, respectively. The outputs are then summed up for fully connected layers. Such a design allows us to separately regularize the gradient norm of the segmentation branch with an additional R1 regularization loss $\mathcal{L}_{R1_{seg}}$. The resulting training framework is similar to StyleGAN2 with the loss function:

$$\mathcal{L}_{all} = \mathcal{L}_{StyleGAN2} + \lambda_{mask}\mathcal{L}_{mask} + \lambda_{R1_{seg}}\mathcal{L}_{R1_{seg}}, \quad (6)$$

where $\mathcal{L}_{StyleGAN2}$ denotes the loss functions used in the original StyleGAN2.

## 4. Implementation Details

We implement our methods using PyTorch 1.15 library. We use the same optimizer and batch settings as in StyleGAN2. $\lambda_{R1_{img}}$, $\lambda_{R1_{seg}}$, $\lambda_{mask}$ are set to 10, 1000 and 100, respectively. Style mixing probability and path regularization are reduced to 0.3 and 0.5, respectively. The 256×256 and 512×512 models are trained on 4 and 8 32GB Tesla V100 GPUs, respectively. For some experiments, we fine-tune our models on image-only datasets. In such cases, we drop the segmentation branch in discriminator and use the original StyleGAN2 loss functions to fine-tune the model. *Due to the space limit, more details about network architectures are given in appendix.*

## 5. Experiments

### 5.1. Semantic-aware and Disentangled Generation

We first evaluate our model on the synthesis quality and its local disentanglement. For synthesis quality, we compare our model with StyleGAN2 [37] and SemanticGAN [41]. The original StyleGAN2, which neither models segmentation masks nor provides local controllability,

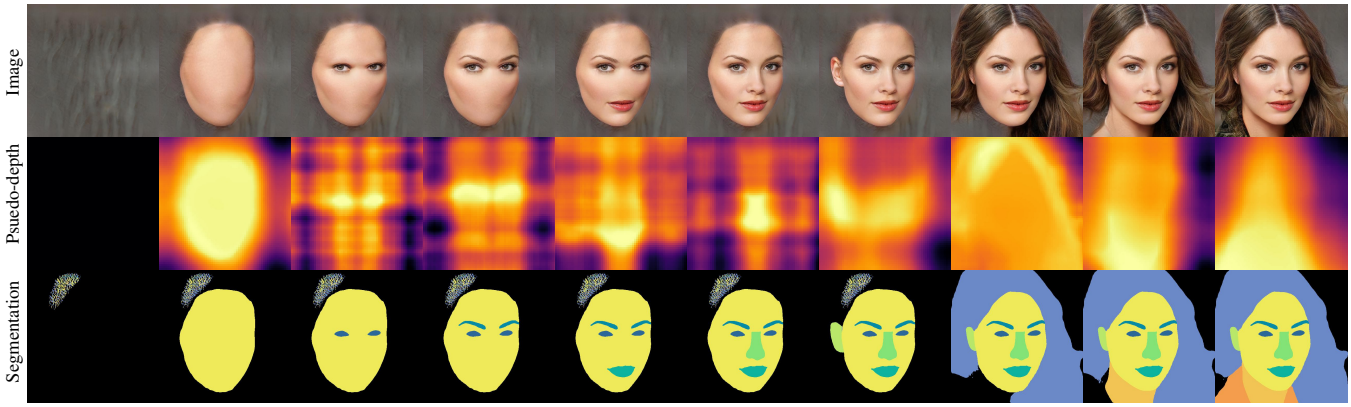


Figure 4. Illustration of compositional synthesis. Starting from background, we gradually add more components into the feature map. The second row shows the pseudo-depth map of each corresponding component used for fusion. Note that the "hair" generator outputs a complete shape even though it is covered by the face. During synthesis, all pseudo-depth maps are fused without an order.

Figure 5. Example generation results of our model trained on CelebAMask-HQ. The images are generated at a resolution of 512×512 with a truncation of 0.7.


Figure 6. Image composition and transformation via Fourier feature manipulation. Real images are used as background for synthesis and inverted into feature maps. Then foreground can be synthesized on this real image in the feature space. The location and size of foreground can be controlled via Fourier features.

is compared against as an upper bound of synthesis quality. SemanticGAN modifies StyleGAN2 into a joint training framework to output both image and segmentation masks. Since its goal is to conduct segmentation, it does not allows local control either. All the models are trained on the the first 28,000 images of CelebAMask-HQ resized to 256×256. Fréchet Inception Distance (FID) [29] and Inception Score (IS) [57] are used to measure the synthesis quality.

Our project is initially built on SemanticGAN framework for learning a semantic-aware model. The original SemanticGAN is semi-supervised and we change it to use all the training labels. As shown in Tab. 1, SemanticGAN achieves much lower quality compared to original StyleGAN, indicating that learning a joint model of images and segmentation masks is a challenging task. Hypothesizing that the main bottleneck of SemanticGAN is the additional patch discriminator used for learning segmentation masks, we replace it with the proposed dual-branch discriminator. The new training framework achieves much better synthesis score. We further replace the SemanticGAN generator with our SemanticStyleGAN generator. Compared to SemanticGAN generator, our model shows a similar synthesis quality while providing additional controllability on each semantic area. We then extend our model to 512×512 resolution and achieve a FID and IS of 7.22 and 3.47, respectively. For reference, the StyleGAN2 generator achieves a FID and IS of 6.47 and 3.55, respectively. Fig. 5 shows the synthesis results of the 512×512 model.

To interpret the compositional synthesis of our model, Fig. 4 shows the results of synthesis with limited components. We first disable all the foreground generators and gradually add them into the forward process. It can be seen that these local generators can work independently to generate a semantic part. The pseudo-depth maps, in spite of the lack of 3D supervision, learn meaningful shapes that could be used to collaboratively compose different faces. We note that unlike traditional GANs that generates a complete image, such a compositional process also allows our model to generate the foreground only and control it by manipulating the Fourier features (See Fig. 6).

Fig. 7 shows the results of latent interpolation of our generator model. The first row shows that our model could interpolate smoothly between two randomly sampled images. Besides, we can interpolate on a specific semantic area by changing the corresponding latent codes, e.g. face or hair, while fixing irrelevant parts. The results indicate that our model has learned a smooth and disentangled latent space


Figure 7. Results of latent interpolation on the whole latent space and specified subspaces. Here, "Face" refers to all the components relevant to face, including eyes, mouth, etc.

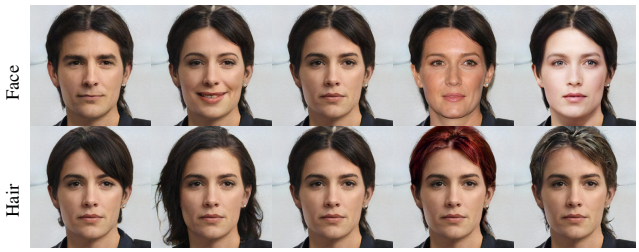


Figure 8. Random sampling in local latent spaces. The middle row shows a randomly sampled face. The first two and last two columns show the results of changing shape and texture codes, respectively. Here, "Face" refers to all the components relevant to face, including eyes, mouth, etc.

| Method | MSE↓ | ID↑ | LPIPS↓ |
|---|---|---|---|
| StyleGAN2 (FFHQ) | 0.031± 0.015 | 0.654± 0.097 | 0.309±0.046 |
| StyleGAN2 | 0.029± 0.016 | 0.575± 0.119 | 0.330±0.052 |
| SemanticStyleGAN | 0.031± 0.017 | 0.602± 0.122 | 0.335±0.051 |

Table 2. Quantitative evaluation of reconstruction performance using Restyle (psp) encoder. The bottom two rows (StyleGAN2 and Ours) are trained on the same split of CelebAMask-HQ.

for semantic editing. Fig. 8 further shows the results of individually changing shape and texture codes. Overall, even though there is no explicit constraint during training, we observe that our model could disentangle most local shapes and textures. We also refer the readers to the appendix for more results on semantic local style mixing.

## 5.2. Controlled Synthesis and Image Editing

With the semantic decomposition in the latent space, our model provides a more disentangled generative prior for image editing. Here, we evaluate our model on downstream editing tasks and compares it to StyleGAN2. We use the pytorch conversion of official StyleGAN2 (config-F on FFHQ 1024x1024) as our baseline, which is widely used in relevant studies on image editing. The 512×512 model is used for our method.

### 5.2.1 Encoding and Editing Real Images

To evaluate the editing results on real images, we first need to embed such images into the GAN latent space. Here, we adopt a state-of-the-art GAN encoder, i.e. Restylepsp [5], for both StyleGAN2 and our model. We use the official model from Restyle authors for StyleGAN2 while a new encoder is trained for our model with default hyperparameters. For reference, we also train a encoder for our StyleGAN2 that is trained on CelebAMask-HQ. Tab. 2 shows the quantitative results of image reconstruction using restyle encoders. Overall, our model achieves a comparable performance in terms of reconstruction.

The next question is whether our model can be applied to local editing on these reconstructed images. Here, we adopt two popular editing methods that were proposed for StyleGAN2: InterFaceGAN [60] and StyleFlow [4]. Both methods need to generate a set of fake images and label their attributes to train a latent manipulation model. In particular, InterFaceGAN learns a linear SVM while StyleFlow uses a conditional continuous normalizing flow [22] to model the latent attribute manipulation. For both generators, we randomly synthesize 50,000 images for labeling. Following InterFaceGAN, a ResNet-50 [27] is trained on CelebA dataset [45] to label these images. During the experiments, we found that our model trained on CelebAMask-HQ exhibits a much lower diversity compared to FFHQ-based StyleGAN2. Thus, we fine-tune our model on FFHQ for $1,000$ steps (See Sec. 4), for which we observe a sufficient improvement of diversity without loss of controllability.

We choose 4 local attributes covering different parts of the face image for editing experiments, namely smile, baldness, beard and bangs, and test on the last $1,000$ images of CelebAMask-HQ, which were not used for training. For StyleGAN2, we keep the original selection of latent dimensions in these methods for content preservation. For ours, we manually choose relevant areas for editing, *e.g.* hair for baldness and face for beard, which can be regarded as a trivial step during deployment. Fig. 9 shows the qualitative results of applying InterFaceGAN to StyleGAN2 and our model. Although InterFaceGAN successfully edits the attributes on StyleGAN2, irrelevant parts are inevitably altered due to the entanglement in the latent space. In comparison, our model focuses only on specified semantic areas. We also conduct a quantitative evaluation of the editing task. For each image, we control the degree of manipulation to generate 10 images. Then a "preservation-score" curve is plotted using the attribute classifier. Here, $score\ gain$ refers to the average gain in classification score of the target attribute. $pixel\ preservation$ refers to 1 minus the $\ell 1$ loss between the two images. The $\ell 1$ loss is an approximation of $\ell 0$ loss, which computes the number of pixels that has been altered. In our experiments, we found this simple metric best correlates with the spatial difference be-

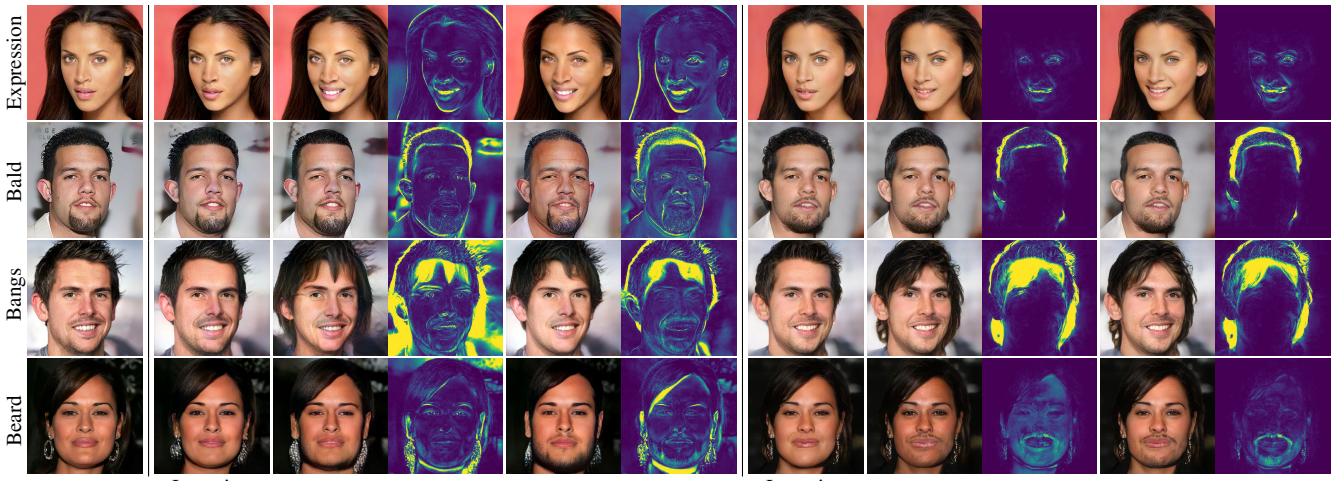


Figure 9. Results of GAN inversion and editing. For each attribute and method, we show the inversion result of Restyle encoder, the edited image and the difference map between them.

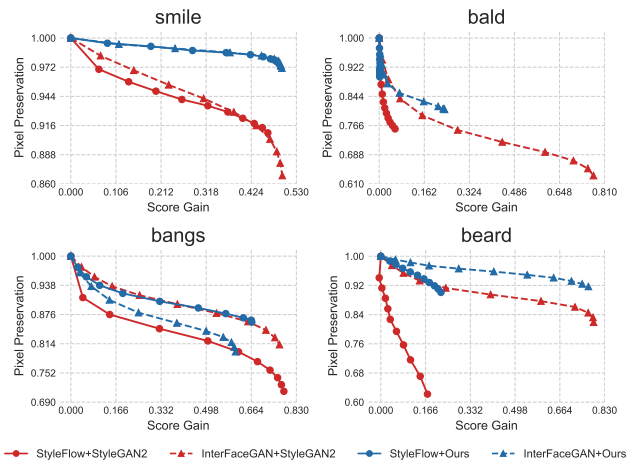


Figure 10. Quantitative comparison of local attribute editing using StyleGAN2 and our model when combined with StyleFlow and InterFaceGAN.

tween images. From Fig. 10, it can be seen that our model achieves a better overall performance. Note that for baldness, our model stops when it removes all hairs, but InterFaceGAN+StyleGAN2 keeps increasing the score by adapting into correlated attributes (such as aging). For bangs, our model tends to increase the overall length of hairs, which could be an inherited bias from original training data. Besides, we found that StyleFlow is more sensitive to label imbalance. Thus, given the small number of bald examples, it fails to learn the baldness attribute for both generators.

#### 5.2.2 Text-guided Synthesis

Recent work have shown that one could use a text-image embedding, such as CLIP [55], to guide the synthesis of StyleGAN2 for controlled synthesis [51]. Similar to attribute editing, StyleGAN2 suffers from the local disentanglement problem on. Fig. 11 shows a few examples of using StyleCLIP [51] to manipulate a synthesized image with a sequence of text prompts. Here, we use the optimization-based version of StyleCLIP as it is flexible for any input text. It can be seen that the original StyleCLIP often modifies the whole image while the text is trying to change only a specified area. Our model, by additionally let the user to choose relevant areas, can faithfully constrain the editing to local parts. The results indicate that our model could be a more suitable tool for text-guided portrait synthesis where detailed descriptions are provided.


Figure 11. Results of text-guided image synthesis under sequential editing. Starting from an average fake face, the first row (from left to right) shows the results of sequentially applying optimization-based StyleCLIP [51] with StyleGAN2 while the second row shows the results of our model with the same input texts.

### 5.3. Results on Other Domains

Training our model from scratch requires access to images and segmentation masks at the same time, which might not be feasible in some cases. Thus, we would like to ask

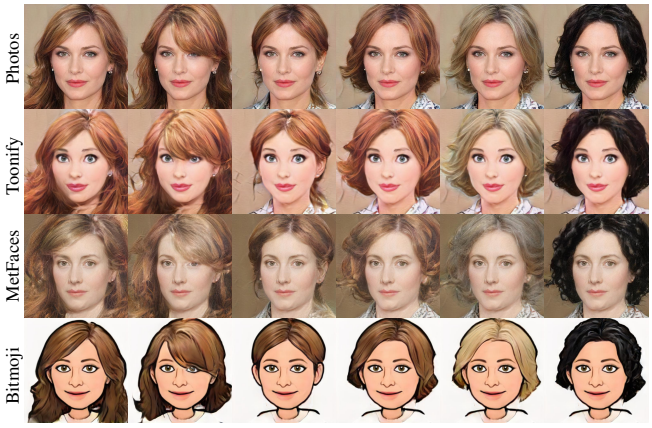


Figure 12. Example results of changing hair styles on adapted new domains. The first four columns and last three columns show the results of sampling different latent codes for hair shapes and textures, respectively.


Figure 13. Controlled generation results on the DeepFashion dataset. Our model can generate various style for different semantic parts.

whether the model can be fine-tuned on image-only datasets while preserving the local disentanglement (See Sec. 4 for fine-tuning). Fig. 12 shows the results after fine-tuning our model on the Toonify [53], MetFaces dataset [34] and BitMoji [1]. All of these datasets have a much smaller number of images compared to CelebAMask-HQ and no segmentation masks. We train our model for hundreds of steps until perceptually good results are generated. It can be seen that, for datasets with a limited domain gap, our model is able to maintain local controllability even after fine-tuning.

In spite of the experiments on face datasets so far, our method indeed does not include any module that is designed for face only and hence can be applied to other objects as well. Fig. 13 shows the results of training our model on the DeepFashion dataset [44], for which we obtain the labels from [78]. With the default hyper-parameters, we find that our model can be successfully trained on fashion datasets and we can similarly control the structure and texture of different semantic parts in the latent space.

## 6. Limitations and Discussion

**Applicable Datasets** Although we have shown that our method can be applied to other domains beyond face photos, we still see a limitation caused by the design and supervision. Since we need to build a local generator for each class, the method would not scale to datasets that have too many semantic classes, such as scenes [74]. Besides, for the purpose of synthesis quality, we change the semi-supervised framework of SemanticGAN [41] into fully-supervised, which limits our model from training on image-only datasets from scratch. It would be beneficial to develop a semi-supervised version of our method in the future.

**Disentanglement** As the disentanglement between pose, shape and texture is only enforced by the design of layer separation in local generators, we see that the boundary between them is still sometimes ambiguous. For example, the shared coarse structure code could encode some information about expression and the shape code could affect the beard. However, in this work, we mainly focus on the spatial disentanglement between different semantic parts and we believe additional regularization losses or architecture tuning could be incorporated in the future to better decouple those information.

**Societal Impact** Our work focuses on the technical problem of improving controllability of GANs and is not specifically designed for any malicious uses. This being said, we do see that the method could be potentially extended into controversial applications such as generating fake profiles. Therefore, we believe that the images synthesized using our approach should present itself as synthetic.

## 7. Conclusion

In this paper, we present a new type of GAN method that synthesizes images in a controllable way. Through the design of local generators, masked feature aggregation and joint modeling of images and segmentation masks, we are able to model the structure and texture of different semantic areas separately. Experiments show that our method is able to synthesize high-quality images while disentangling different local parts. By combining our model with other editing methods, we can edit synthesized images with a more fine-grained control. Experiments also show that our model can be adapted to image-only datasets while preserving disentanglement capability. We believe the proposed method presents a new and interesting direction of GAN priors for controllable image synthesis, which could shed light on many potential downstream tasks.

## References


[1] Bitmoji dataset. https://www.kaggle.com/mostafamozafari/bitmoji-faces/version/1, 2013. Released under CC BY 4.0. 8, 12

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *CVPR*, 2019. 1

[3] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *CVPR*, 2020. 1

[4] Rameen Abdal, Peihao Zhu, Niloy J Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *TOG*, 2021. 1, 2, 6

[5] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. *arXiv:2104.02699*, 2021. 1, 6

[6] Titas Anciukevicius, Christoph H Lampert, and Paul Henderson. Object-centric image generation with factored depths, locations, and appearances. *arXiv:2004.00642*, 2020. 2

[7] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhenkov. Image generators with conditionally-independent pixel synthesis. In *CVPR*, 2021. 3

[8] Relja Arandjelović and Andrew Zisserman. Object discovery with a copy-pasting gan. *arXiv:1905.11369*, 2019. 2

[9] Oron Ashual and Lior Wolf. Specifying object attributes and relations in interactive scene generation. In *ICCV*, 2019. 3

[10] Samaneh Azadi, Deepak Pathak, Sayna Ebrahimi, and Trevor Darrell. Compositional gan: Learning image-conditional binary composition. *IJCV*, 2020. 2

[11] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv:1809.11096*, 2018. 1, 2

[12] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv:1901.11390*, 2019. 2, 3, 4

[13] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 2

[14] Anpei Chen, Ruiyang Liu, Ling Xie, Zhang Chen, Hao Su, and Jingyi Yu. Sofgan: A portrait image generator with dynamic styling. *ACM TOG*, 2021. 2

[15] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *ICCV*, 2017. 2

[16] Shu-Yu Chen, Wanchao Su, Lin Gao, Shihong Xia, and Hongbo Fu. Deepfacedrawing: Deep generation of face images from sketches. *TOG*, 2020. 2

[17] Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *CVPR*, 2020. 3

[18] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and controllable face image generation via 3d imitative-contrastive learning. In *CVPR*, 2020. 2

[19] Sébastien Ehrhardt, Oliver Groth, Aron Monszpart, Martin Engelcke, Ingmar Posner, Niloy Mitra, and Andrea Vedaldi. Relate: Physically plausible multi-object scene synthesis using structured latent spaces. *arXiv:2007.01272*, 2020. 2

[20] SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Geoffrey E Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. *NIPS*, 2016. 2

[21] David Futschik, Michal Lukáč, Eli Shechtman, and Daniel Sỳkora. Real image inversion via segments. *arXiv:2110.06269*, 2021. 3

[22] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. In *ICLR*, 2018. 6

[23] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *ICML*, 2019. 2, 3, 4

[24] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *ICML*, 2015. 2

[25] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv:2110.08985*, 2021. 2

[26] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. In *NIPS*, 2020. 1, 2

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[28] Paul Henderson and Christoph H Lampert. Unsupervised object-centric video generation and decomposition in 3d. *arXiv:2007.06705*, 2020. 2

[29] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NIPS*, 2017. 5

[30] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *CVPR*, 2018. 3

[31] Drew Arad Hudson and C Lawrence Zitnick. Compositional transformers for scene generation. In *NIPS*, 2021. 2

[32] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *CVPR*, 2018. 3

[33] Omer Kafri, Or Patashnik, Yuval Alaluf, and Daniel Cohen-Or. Stylefusion: A generative model for disentangling spatial segments. *arXiv:2107.07437*, 2021. 2, 3

[34] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. 2020. 8

[35] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *arXiv:2106.12423*, 2021. 1, 12

[36] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 2, 3, 4, 12

[37] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1, 2, 3, 4, 12

[38] Hyunsu Kim, Yunjey Choi, Junho Kim, Sungjoo Yoo, and Youngjung Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *CVPR*, 2021. 2

[39] Marek Kowalski, Stephan J Garbin, Virginia Estellers, Tadas Baltrušaitis, Matthew Johnson, and Jamie Shotton. Config: Controllable neural face image generation. In *ECCV*, 2020. 2

[40] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *CVPR*, 2020. 12

[41] Daiqing Li, Junlin Yang, Karsten Kreis, Antonio Torralba, and Sanja Fidler. Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization. In *CVPR*, 2021. 4, 8

[42] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. In *NIPS*, 2021. 3

[43] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, and Hongsheng Li. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. *arXiv:1910.06809*, 2019. 2

[44] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *CVPR*, 2016. 8, 12

[45] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015. 6

[46] Li Nanbo, Cian Eastwood, and Robert B Fisher. Learning object-centric representations of multi-object scenes from multiple views. In *NIPS*, 2020. 2

[47] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *CVPR*, 2019. 2

[48] Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *arXiv:2002.08988*, 2020. 2

[49] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 2, 3

[50] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 2

[51] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *ICCV*, 2021. 7

[52] Martin Pernuš, Vitomir Štruc, and Simon Dobrišek. High resolution face editing with masked gan latent code optimization. *arXiv:2103.11135*, 2021. 3

[53] Justin NM Pinkney and Doron Adler. Resolution dependent gan interpolation for controllable image synthesis between domains. 2020. 8

[54] Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis. In *CVPR*, 2018. 2

[55] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. 2021. 7

[56] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *CVPR*, 2021. 1, 2

[57] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *NIPS*, 2016. 5

[58] Othman Sbai, Camille Couprie, and Mathieu Aubry. Surprising image compositions. In *CVPR*, 2021. 2

[59] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *NIPS*, 2020. 2

[60] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020. 1, 2, 6

[61] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. *arXiv:2007.06600*, 2020. 1

[62] Alon Shoshan, Nadav Bhonker, Igor Kviatkovsky, and Gerard Medioni. Gan-control: Explicitly controllable gans. *arXiv:2101.02477*, 2021. 2

[63] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *CVPR*, 2021. 3

[64] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *NIPS*, 2020. 4

[65] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. In *CVPR*, 2020. 1, 2

[66] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *TOG*, 2021. 1

[67] Sjoerd van Steenkiste, Karol Kurach, Jürgen Schmidhuber, and Sylvain Gelly. Investigating object compositionality in generative adversarial networks. *Neural Networks*, 2020. 2

[68] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. High-fidelity gan inversion for image attribute editing. *arXiv:2109.06590*, 2021. 1

[69] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 2

[70] Yi Wang, Lu Qi, Ying-Cong Chen, Xiangyu Zhang, and Jiaya Jia. Image synthesis via semantic composition. *arXiv:2109.07053*, 2021. 2

[71] Zongze Wu, Dani Lischinski, and Eli Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation. In *CVPR*, 2021. 3

[72] Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. Lr-gan: Layered recursive generative adversarial networks for image generation. *arXiv:1703.01560*, 2017. 2

[73] Yanchao Yang, Yutong Chen, and Stefano Soatto. Learning to manipulate individual objects in an image. In *CVPR*, 2020. 2

[74] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 8

[75] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis. *arXiv:2110.09788*, 2021. 2, 3

[76] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *ECCV*, 2020. 1

[77] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Sean: Image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020. 2

[78] Zhen Zhu, Zhiliang Xu, Ansheng You, and Xiang Bai. Semantically multi-modal image synthesis. In *CVPR*, 2020. 2, 8

# A. Implementation Details

## A.1. Fusion with Transparent Classes

Following Sec 3.1 in the main paper, a coarse semantic mask $\mathbf{m}$ is fused from pseudo-depth maps, which is further used to aggregate local feature maps. In general cases, the aggregation can be simply achieved by computing $\mathbf{f} = \sum_k^K \mathbf{m}_k \odot \mathbf{f}_i$, where the frontal class in the semantic mask will be chosen for the output feature. However, in the case of transparent classes, this formulation could be problematic. For example, although the whole eye area could be labeled as glasses in the semantic masks, we are still able to see the skin behind it. Thus, we treat such transparent classes separately during feature aggregation. In particular, we use a modified mask:

$$\tilde{\mathbf{m}}_k(i,j) = \frac{\mathbb{1}_{NT}(k)\exp(\mathbf{d}_k(i,j))}{\sum_{k'}^K \mathbb{1}_{NT}(k')\exp(\mathbf{d}_{k'}(i,j))} + \mathbb{1}_T(k)\mathbf{m}_k(i,j), \tag{7}$$

where $\mathbb{1}_T(k)$ is an indicator function that equals 1 if $k$ is a transparent class and 0 otherwise. $\mathbb{1}_{NT}(k)$ is the opposite indicator function for non-transparent classes. The first part of Eq. (7) here means that we first aggregate the features without the transparent classes. Then in the second part of Eq. (7), we add the transparent features using their their original weights in mask $m$. In this way, the feature map will not be affected if there are no transparent classes. If there are, they would be added onto the feature map as additional residuals. Note that this formulation assumes that transparent classes do not overlap with each other. In our experiments, we set glasses and earings as transparent glasses. In fact, the model can also be trained stably by simply using the original mask $\mathbf{m}$ for fusion, but the texture behind transparent classes could be distorted.

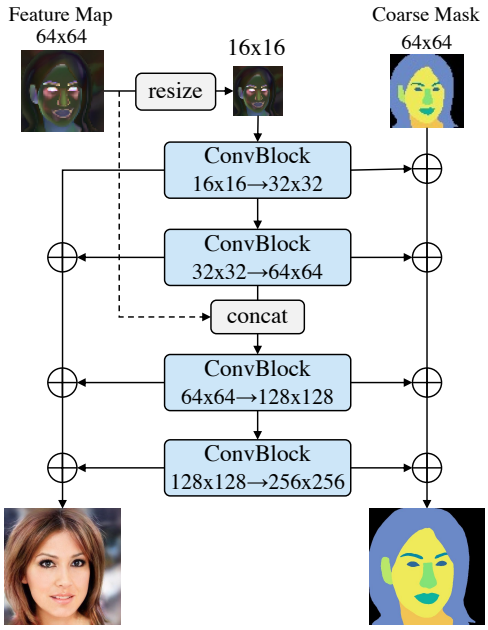


Figure 14. Details of the render net. Here, we take 256×256 model as the example. A "ConvBlock" is a StyleGAN2 convolution block that have 2 convolution layers. We remove the style modulation and add a linear segmentation output branch in each convolution layer. $\oplus$ indicates upsampling and summation.

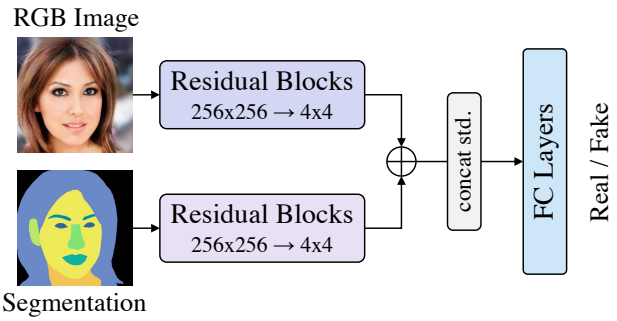


Figure 15. Details of the dual-branch discriminator. The "Residual Blocks" are the convolution layers. The image branch and segmentation are symmetric except the input channels. "concat std." is the step to of calculating standard deviation. The discriminator would be equivalent to StyleGAN2 discriminator if we remove the segmentation branch.

## A.2. Architecture Details

As shown in Fig. 3 in the main paper, each **local generator** is a modulated MLP (implemented by 1×1 convolution)

that has 10 layers. The input and output feature maps are both of size $64 \times 64$. All the hidden layers has 64 channels. The Fourier feature at the input is first transformed into the hidden feature map with a linear fully connected (FC) layer. The "toDepth" layer is a FC layer that outputs a 1-channel pseudo-depth map. The "toFeat" is a FC layer that outputs a 512-channel feature map. To encourage the disentanglement between shape and texture, we stop the gradient between shape and texture layers except for the background generator. We also fix the pseudo-depth map of background generator to be all 0s.

The detailed architecture of **render network** is shown in Fig. 14. Note that there is an upsampling and residual operation every layer for the segmentation mask, so $\Delta\mathbf{m}$ is not explicitly computed. Instead, we calculate $\mathcal{L}_{mask}$ by the difference between downsampled output segmentation and coarse mask $\mathbf{m}$.

The detailed architecture of **discriminator** is shown in Fig. 15. It is similar to StyleGAN2 discriminator except that we add an additional segmentation branch that is symmetric to image branch. During fine-tuning, we remove this branch and the discriminator reduces to an image discriminator.

### A.3. Efficiency

For the 512×512 model, our model takes about two and a half day to train 150,000 steps with a batch size of 32 on 8 32GB Nvidia Tesla V100 GPUs, where the best model is then selected. For inference, it takes 0.137s for our model to generate an image on a single GPU without parallelizing local generators.

## B. Style Mixing and Additional Results

In the main paper, we showed that the proposed model can interpolate smoothly in a local latent space. Here, we show results on more fine-grained style mixing using our model. Different from StyleGANs [35–37], we can conduct style mixing between local generators to transfer a certain semantic component from one image to another. This conducted by transferring both the shape code $\mathbf{w}_s^k$ and texture code $\mathbf{w}_t^k$. Fig. 16 shows the results of semantic style mixing using our model trained on CelebAMask-HQ [40]. Besides the local latent codes, we also show the transferring results of the coarse structure code $\mathbf{w}_{base}$. It can be seen that our model is able to transfer most local component styles between images, including small components such as eyes and mouth. However, it is also observable that the coarse structure code is currently encoding some information about these local components, such as expression and hair. Although a user or developer is able to change the number of coarse structure codes dynamically during testing (and even manipulate all the layers in a local generator), we believe it would be beneficial to further regularize the information in the coarse code in the future. Fig. 17 and Fig. 18 show the semantic style mixing results of a model after transfer learning (on BitMoji dataset [1]) and the model trained on DeepFashion dataset [44]. A similar effect can be seen on the DeepFashion that the coarse structure would affect certain components. Also, we see that sometimes the hair color is affected by the background on this dataset. Since the head in this dataset is rather small, we believe such entanglement is caused by the low-resolution (16×16) feature map that is fed into render network for blending, which was originally selected for face datasets. Further tuning the hyper-parameters of the render net might alleviate such issues.

Fig. 19 and Fig. 20 show more results on randomly sampled images and pseudo-depth maps, respectively Figs. 21 to 24 show more results on real face editing using our model and original StyleGAN2. As mentioned in the main paper, we see that StyleFlow is more sensitive to data imbalance and less robust. Taking bangs for instance, it tries to reduce the hair on the side but not in the front for our model. For beard, it tries to make face skin look darker for our model while completely fails on the original StyleGAN2. Note that we re-train both StyleFlow and InterFaceGAN using newly sampled images and our own attribute prediction model. Overall, we can observe that our model achieves much more localized control when editing output images.


Figure 16. Local style mixing of the model trained on CelebAMask-HQ. The first column shows randomly sampled images for editing. The remaining columns show the results of mixing local styles using the reference images in the first row.

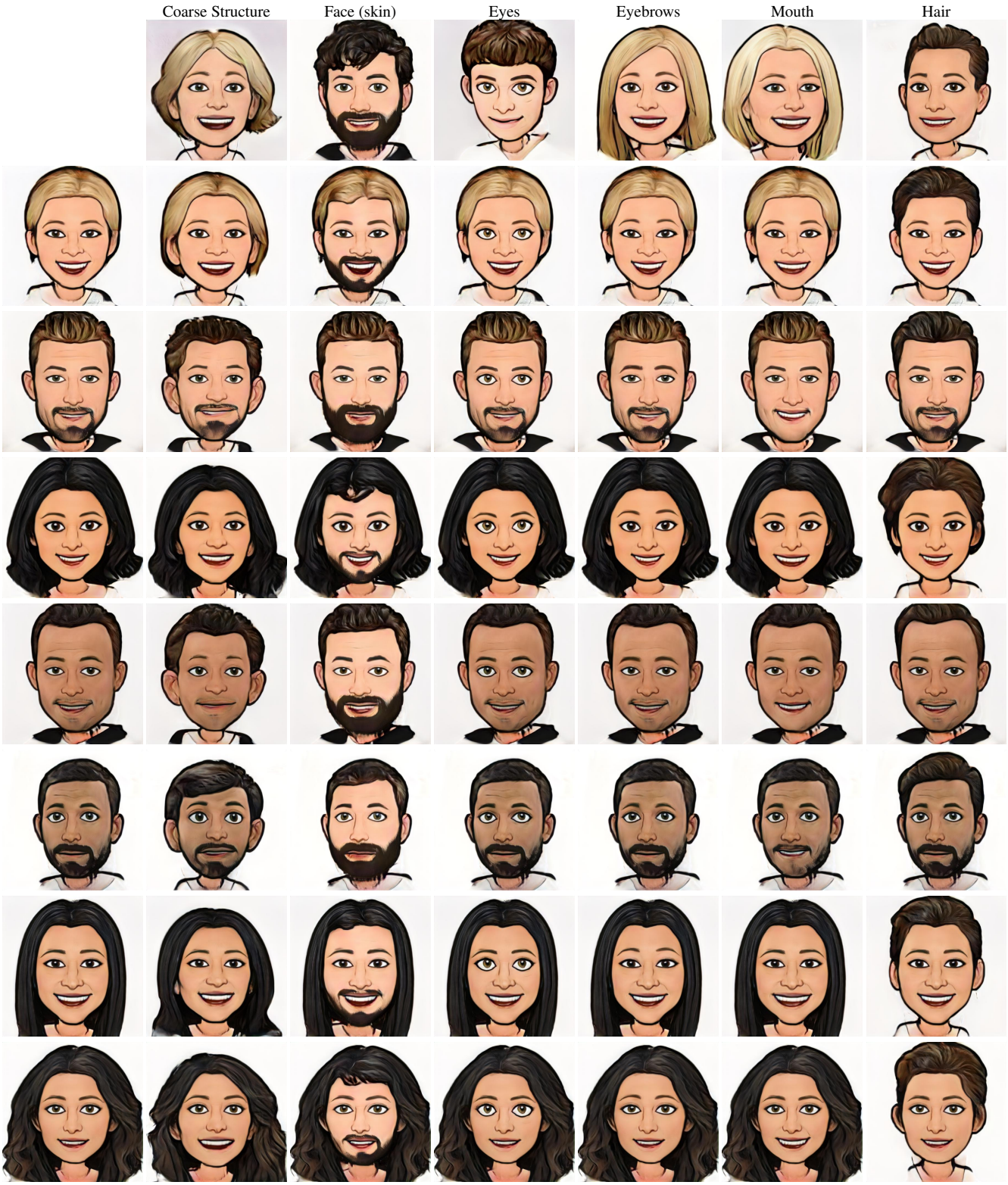


Figure 17. Local style mixing of the model fine-tuned on the BitMoji dataset. The first column shows randomly sampled images for editing. The remaining columns show the results of mixing local styles using the reference images in the first row.

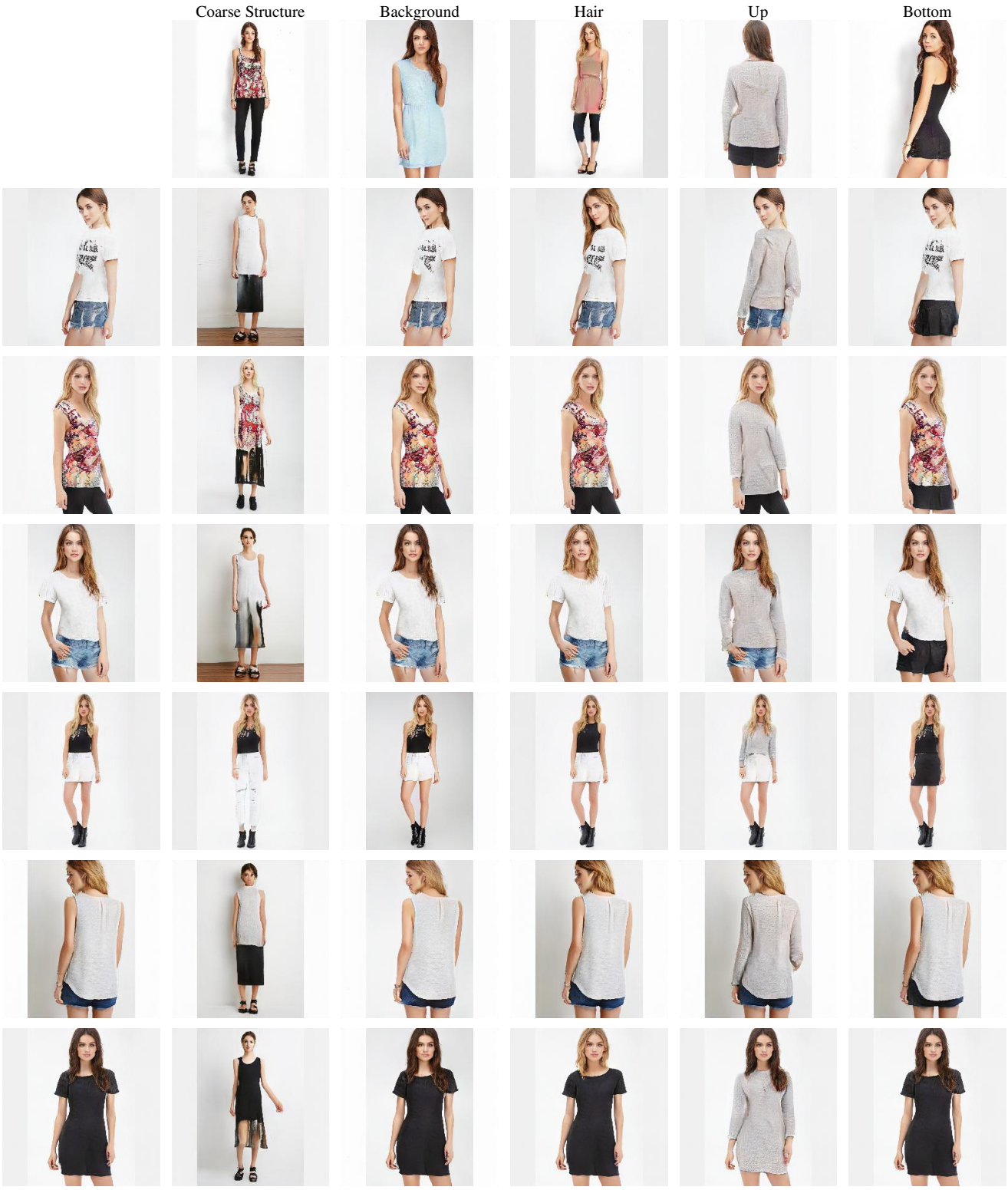


Figure 18. Local style mixing of the model trained on the DeepFashion dataset. The first column shows different randomly sampled images for editing. The remaining columns show the results of mixing local styles using the reference images in the first row.

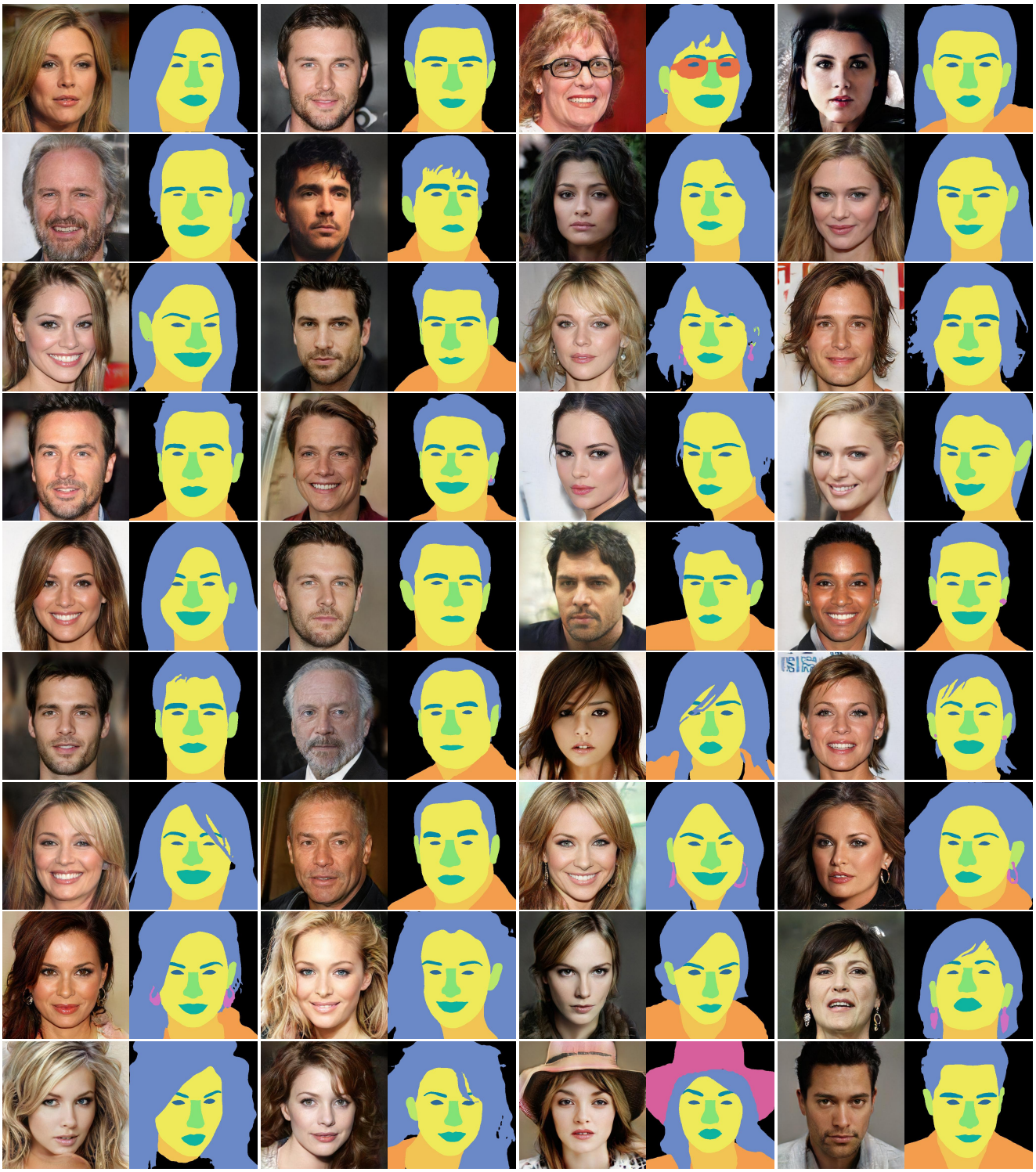

Figure 19. Example generated images and using our 512×512 trained on CelebAMask-HQ. On the right of each generated photo is the refined segmentation mask output by the model.

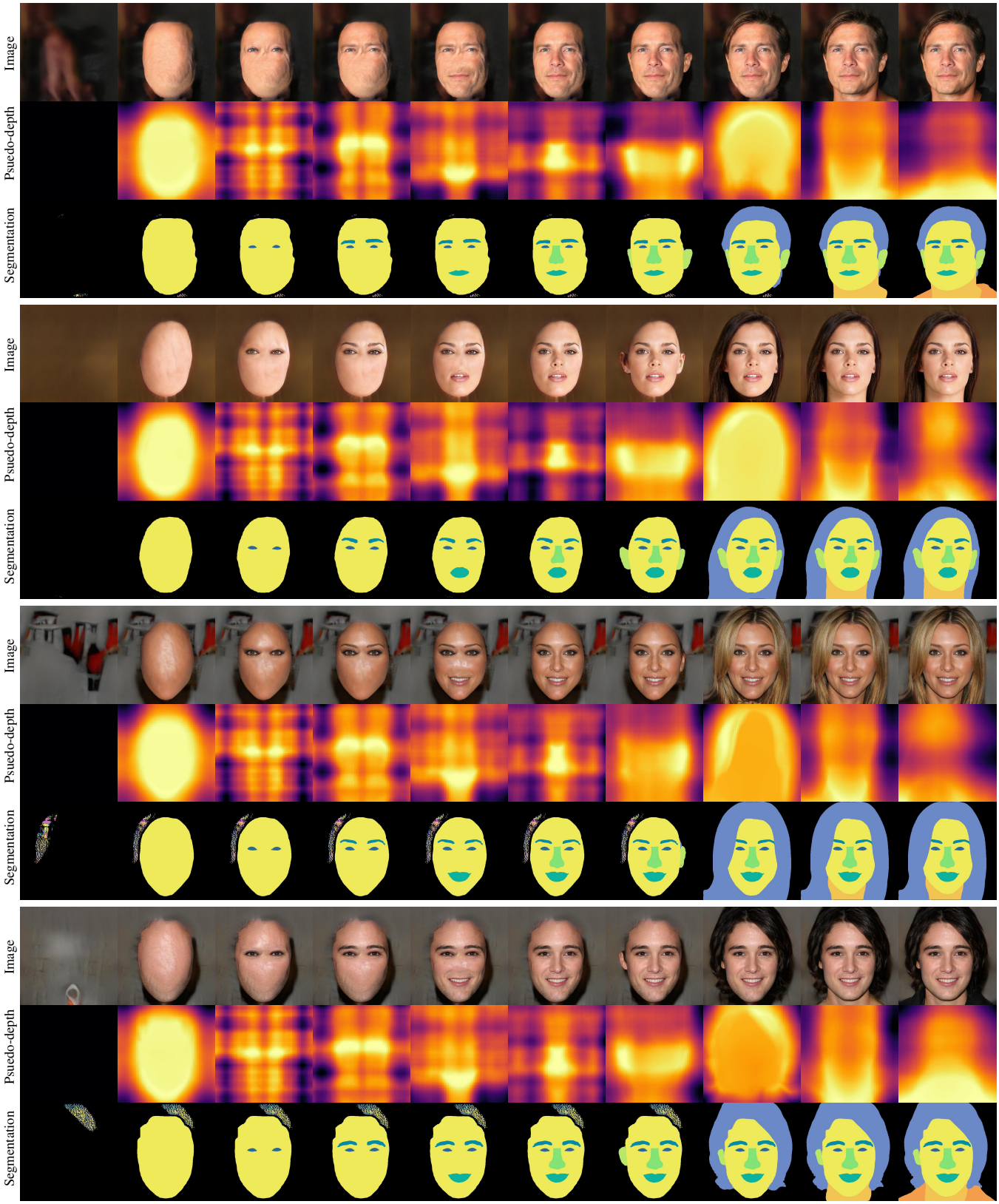


Figure 20. Illustration of compositional synthesis. Starting from background, we gradually add more components into the feature map. The second row of each sample shows the pseudo-depth map of each corresponding component used for fusion. During synthesis, all pseudo-depth maps are fused without an order.

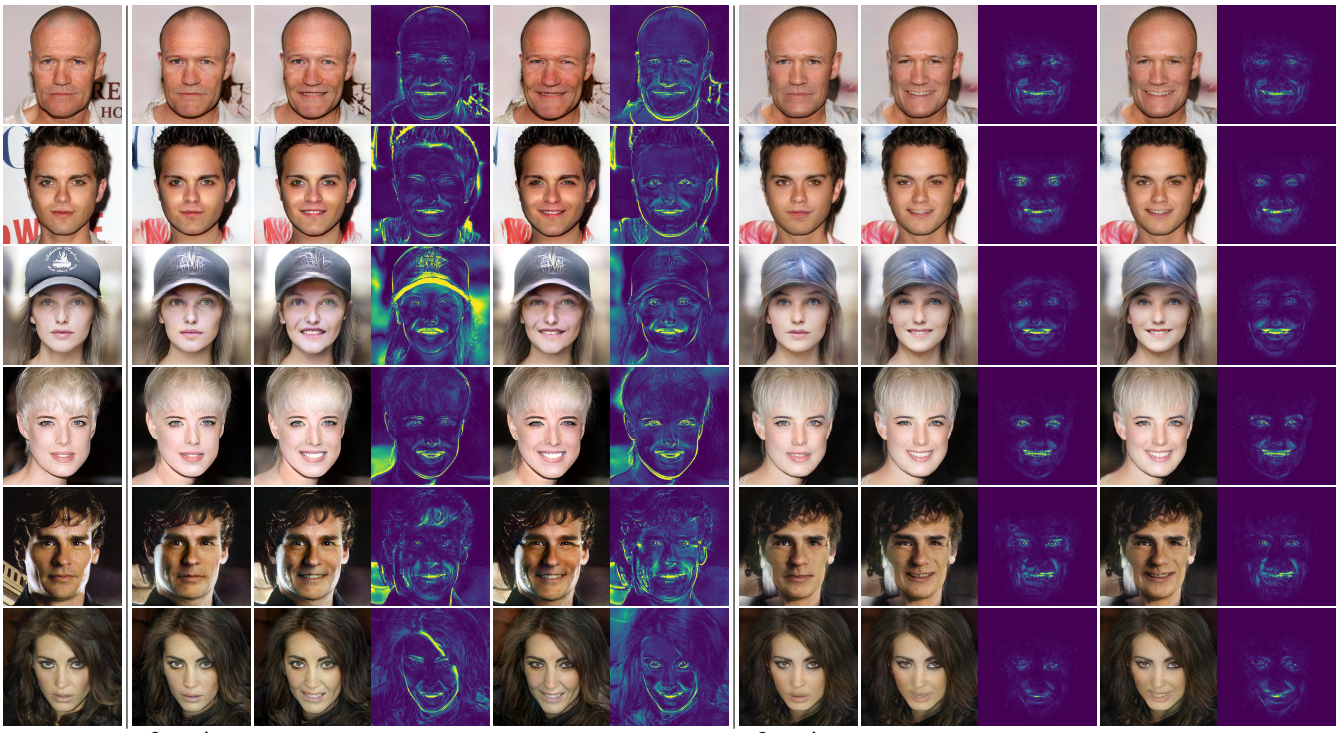


Figure 21. Results of GAN inversion and editing for the **smile** attribute. For each method, we show the inversion result of Restyle encoder, the edited image and the difference map between them.

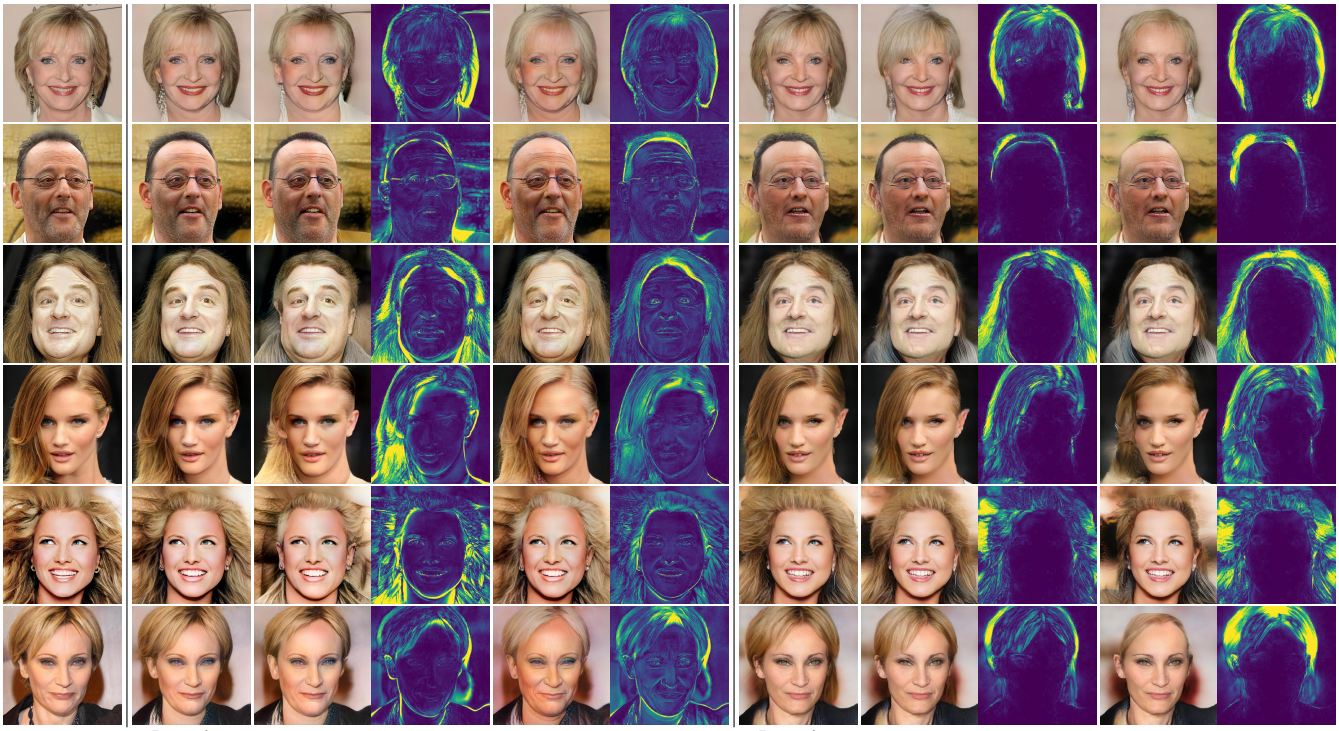


Figure 22. Results of GAN inversion and editing for the **bald** attribute. For each method, we show the inversion result of Restyle encoder, the edited image and the difference map between them.


Figure 23. Results of GAN inversion and editing for the **bangs** attribute. For each method, we show the inversion result of Restyle encoder, the edited image and the difference map between them.

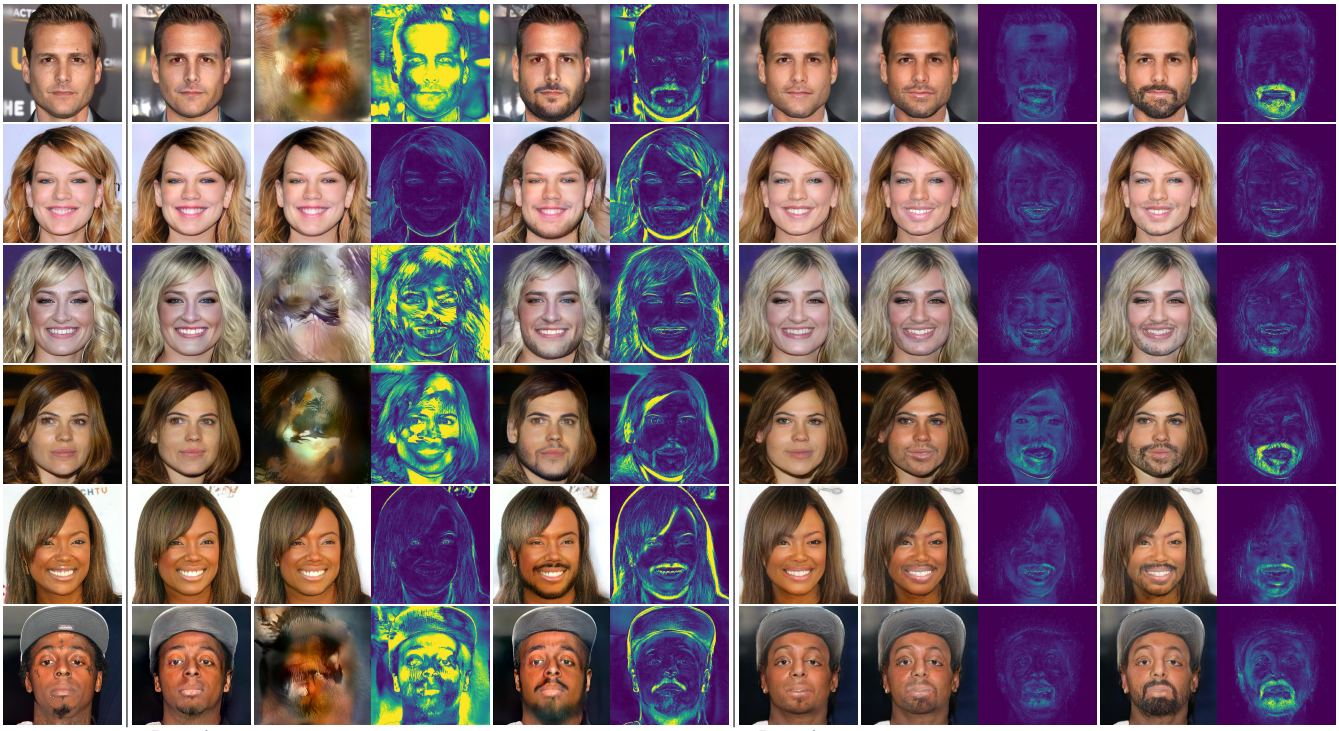


Figure 24. Results of GAN inversion and editing for the **beard** attribute. For each method, we show the inversion result of Restyle encoder, the edited image and the difference map between them.