Feature-Style Encoder for Style-Based GAN Inversion

Xu Yao[1,2], Alasdair Newson[1], Yann Gousseau[1], Pierre Hellier[2]
[1] LTCI, Télécom Paris, Institut Polytechnique de Paris, France
[2] InterDigital R&I, 975 avenue des Champs Blancs, Cesson-Sévigné, France

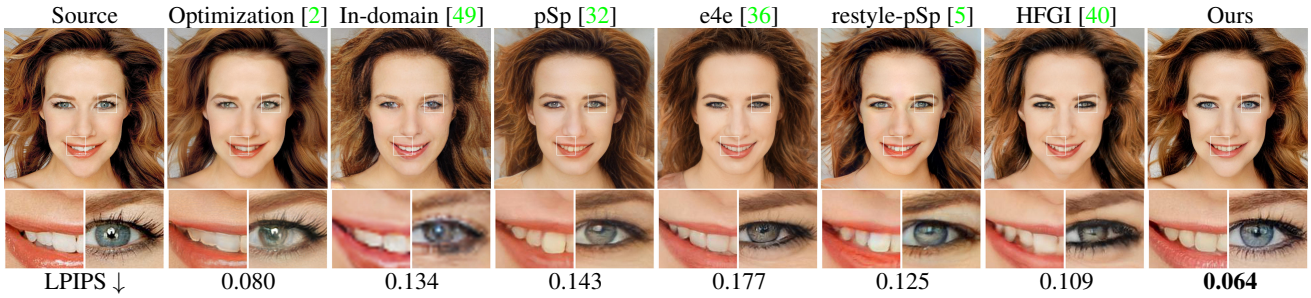


Figure 1. **Inversion of a real image in the latent space of StyleGAN2.** We compare our model against state-of-the-art for the inversion of StyleGAN2 [22] pre-trained on face domain. Our method outperforms the state-of-the-art by up to $20\% - 50\%$ in LPIPS distance [48].

Abstract

We propose a novel architecture for GAN inversion, which we call Feature-Style encoder. The style encoder is key for the manipulation of the obtained latent codes, while the feature encoder is crucial for optimal image reconstruction. Our model achieves accurate inversion of real images from the latent space of a pre-trained style-based GAN model, obtaining better perceptual quality and lower reconstruction error than existing methods. Thanks to its encoder structure, the model allows fast and accurate image editing. Additionally, we demonstrate that the proposed encoder is especially well-suited for inversion and editing on videos. We conduct extensive experiments for several style-based generators pre-trained on different data domains. Our proposed method yields state-of-the-art results for style-based GAN inversion, significantly outperforming competing approaches. Source codes are available at https://github.com/InterDigitalInc/FeatureStyleEncoder.

1. Introduction

The incredible image synthesis power of Generative Adversarial Networks (GANs) [12] has been amply demonstrated by the great quantity of work on this architecture since its creation. However, since a GAN only decodes an image from a probabilistic latent space, a significant research problem is how to *encode* images into the latent space of a pretrained GAN, especially in the case of real (photographic) images, as opposed to *synthetic* images, which are generated by sampling in the latent space. Recent studies [16, 33, 34, 43] have shown that it is possible to control semantic attributes of synthetic images by manipulating the latent space of a pre-trained GAN, however an efficient encoding method, necessary for real images, still remains an open problem, especially in the case of these editing tasks. Thus, an ideal encoder should lead to high perceptual quality on real images, while ensuring that the inverted latent codes are amenable to the same editing that is possible in the case of synthetic images.

Among the many studies on GAN inversion, recent works have been primarily focused on style-based generators [20–22], because of their excellent performance in high quality image synthesis, especially on faces. Unlike traditional generative models which feed the latent code though the input layer only, a style-based generator feeds latent code through adaptive instance normalization at each convolution layer to control the style of the generated image.

To project a given image to the latent space of a style-based GAN model, there are two approaches: optimization and learning an encoder. The most straight-forward is to perform optimization [2] on the latent code by minimizing the image reconstruction error. In order to achieve higher perceptual quality, including the feature maps in the optimization has been proposed [17, 50]. In spite of these achievements, optimization-based methods have significant shortcomings. One major drawback is that the optimization

process is carried out locally with respect to a single image. Thus, the resulting inverted latent codes do not necessarily lie on the original latent space, which makes them difficult to use for editing tasks, as shown by [5, 36].

A better approach is to *learn an encoder* to invert images to latent codes [5, 32, 36, 42, 49]. The inverted latent codes are more regularized and therefore *better suited for editing*. The inversion process is also much faster, which is especially important for computationally intensive tasks such as video editing. However, current encoder-based methods also have several limitations. Firstly, the reconstruction error of the inversion is larger than that of optimization-based methods. Secondly, the reconstructed image is perceptually similar to the input but lacks finer details and appears over smoothed. Thirdly, encoding only the latent code fails to generate correct inversion on outlier data. For instance, given a talking face video, we observe that such methods fail to invert non-frontal poses, thus damaging the consistent reconstruction along the sequence.

To tackle the above mentioned weaknesses, we propose an new inversion architecture. We learn an encoder in the *feature-style* space, which maps an image to a feature code and a latent code. The feature code encodes spatial details, and the latent code is used for editing. This design significantly improves the perceptual quality of the inversion and achieves a balanced trade-off between reconstruction quality and editing capacity. The main contributions of our paper can be summarized as follows:

- We propose a new GAN encoder, which is the first to exploit the idea of encoding feature and style separately, without any optimization step at inference time. Such optimization is both costly and leads to unreliable editing results. Our Feature-Style encoder, on the other hand, significantly improves the perceptual quality of the inversion and improves latent space editing;
- We present a novel training approach, which learns two inversions simultaneously - one which is amenable to editing and one of higher fidelity but less adapted for editing. By training in this manner, our encoder achieves a balanced trade-off between reconstruction quality and editing capacity;
- We conduct extensive experiments to show that our model greatly outperforms state-of-the-art methods on inversion and editing tasks on images and videos. In particular, we improve the perceptual metrics by a very large margin (50%). In addition, we show that the video inversion results of our method is more consistent and stable, which favors further editing on videos.

2. Related works

The goal of our work is to learn an encoder for projecting real images to the latent space of a pre-trained style-based generator. Much of the recent literature on GAN inversion pays particular attention to style-based generators [19–22], as their latent spaces are better disentangled and have improved editing properties.

Style-based Generator. Karras *et al*. proposed the first style-based generator, named StyleGAN [21]. Unlike traditional generative models which feed the latent code though the input layer only, a style-based generator feeds latent code through adaptive instance normalization at each convolution layer to control the style of the generated image. The perceptual quality and variety of the StyleGAN synthetic images surpassed previous image generative models [7, 18]. In StyleGAN2 [22], the image quality was improved further by introducing weight demodulation and path length regularization and redesigning the generator normalization. The StyleGAN2-Ada [19] explored the possibility to train a GAN model with limited data regimes, by using an adaptive discriminator augmentation mechanism that significantly stabilizes training. The third generation, alias-free GAN [20], addressed the aliasing artifacts in the generator, by employing small architectural changes to discard unwanted information and boost the generator to be fully equivariant to translation and rotation.

Latent Space Editing. The motivation of GAN inversion is to achieve real image editing using the latent space of a pretrained GAN model. Various studies show it possible to edit synthetic images by manipulating the corresponding latent code. Local semantic editing can be achieved by optimizing the latent code directly [2, 26]. To explore high level semantic information in the latent space, learning based techniques have been proposed. These techniques include unsupervised exploration [38], learning linear SVM models [33], principle component analysis on the latent codes [16], and k-means clustering of the activation features [10]. To achieve better disentangled editing, [3, 14, 29, 35, 46] proposed to learn neural networks in the latent space. The recent works [34, 39] discovered interpretable transforms by directly decomposing the weights or feature maps of pretrained GANs. Additionally, [6, 25] modify the style-based GAN architecture and retrain it for better disentanglement in image generation. [23, 28, 30] train jointly an encoder and a style-based decoder architecture for image manipulations.

GAN Inversion. The goal of GAN inversion is to encode a real image to the latent space of a pretrained generator, so that the image generated from the inverted latent code is the reconstruction of the input image. Among the rich literature on GAN inversion [44], the approaches addressing style-based generators can be classified into three types: optimization based methods [1, 15, 17, 45, 50], encoder based models [5, 32, 36, 40, 42] and hybrid methods [8, 47, 49]. The optimization based methods update directly the inverted latent code by minimizing the reconstruction error on the input image. For StyleGAN inversion, Abdal *et al*. [1] pro-

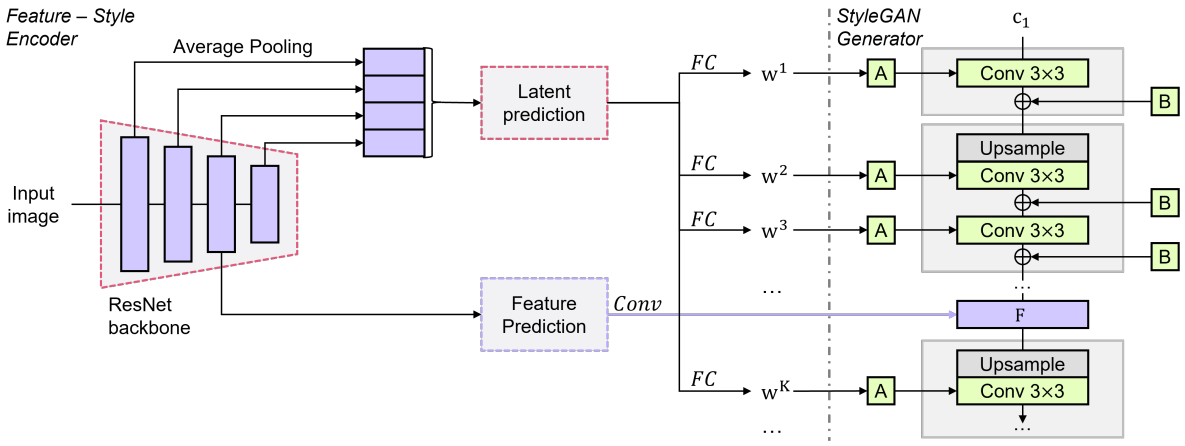


Figure 2. **Feature - Style Inversion Architecture.** Our model consists of a ResNet backbone and two output branches: one for latent code prediction, the other for feature code prediction. The inverted latent code $\mathbf{w}$ is a concatenation of N latent blocks $\{\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^N\}$, each controlling a separate convolution layer in the generator. During generation, we replace the feature maps at the K^{th} convolution layer of the generator with the inverted feature code $\mathbf{F}$, and synthesize the inversion with the latent blocks $\{\mathbf{w}^K, ..., \mathbf{w}^N\}$. K is a fixed parameter, chosen so that reconstruction is accurate and editing can be performed efficiently.

posed to embed the input image in an extended latent space $\mathcal{W}^+$, which offers better flexibility and improves the reconstruction quality. Recently, it was shown that including a feature code in the optimization helps preserve more spatial details and improves the perceptual quality of inversion [17, 50]. Despite the satisfying reconstruction quality, optimization-based methods usually present lower editing quality due to the randomness in the optimization process. To better regularize the inversion, encoder based methods train an encoder to map real images to the latent space of the pretrained generator. Richardson *et al*. [32] proposed the first baseline to learn an encoder for StyleGAN inversion. To improve editing capacity, Tov *et al*. [36] proposed a regularization term which forces the inverted latent code in $\mathcal{W}^+$ to lie closer to the original latent space. A recent concurrent work of Wang *et al*. [40] formulated the inversion task to a data compression problem and proposed an adaptive distortion alignment module to improve the reconstruction quality. Encoder-based methods achieve better editing results but degrade reconstruction quality. On the other hand, hybrid methods take the inverted latent code from a pretrained encoder as initialization and perform optimization on it. Zhu *et al*. [49] proposed to learn a domain-guided encoder and use it as a regularizer for domain-regularized optimization. However, despite the gain in the reconstruction quality, the optimization step makes hybrid methods less suited for video inversion and editing.

3. Method

In this section, we introduce the Feature-Style encoder for real image inversion and editing via the latent space of a pretrained style-based generator. In our model, we use a ResNet backbone with two output branches: one for the inverted latent code, the other for the encoded feature maps. Figure 2 shows the overall architecture, as well as how the latent code and feature maps are plugged into the StyleGAN architecture to generate the reconstructed image.

3.1. Overview

A style-based generator, such as StyleGAN [20–22], consists of a mapping network and a generator $\mathbf{G}$. The mapping network first maps a random latent code $\mathbf{z} \in \mathcal{Z}$ to an intermediate latent code $\mathbf{w} \in \mathcal{W}$, which is further used to scale and bias the feature maps, ie the outputs of a convolutional layer, after each layer in the generator. To project a synthetic image $\mathbf{G}(\mathbf{w})$ to the latent space, it is possible to compute the latent code in the original latent space $\mathcal{W}$ and achieve a satisfying inversion. However, it is much more difficult to project a real image to the original latent space [22], due to the gap between the real data distribution and the synthetic one. An alternative is to project real images to an extended latent space $\mathcal{W}^+$ [1], where $\mathbf{w} \in \mathcal{W}^+$ is a concatenation of N latent blocks $\{\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^N\}$, each controlling a convolution layer in the generator.

In addition to the latent code, StyleGAN1-2 [21, 22] add Gaussian noise after each convolution layer to generate local spatial variations. Abdal *et al*. [2] show that it is possible to reconstruct a real image by jointly optimizing the latent code in $\mathcal{W}^+$ and the noise maps. However, in the original StyleGAN architecture, these noise maps are drawn from a Gaussian distribution, and are therefore not designed to represent geometric elements of the image. Unfortunately, this optimization of noise maps ends up encoding such geometric information in the noise. This is particularly prob-

lematic for latent space editing, since it becomes very difficult to modify geometric information efficiently, which is extremely limiting for an editing algorithm.

On the other hand, without optimization on the noise maps, it is impossible to achieve perfect reconstruction for real images by optimizing the latent code only. To tackle this dilemma, the authors of [17, 50] propose to include the *feature maps* in the optimization process, rather than optimizing noise maps, in order to preserve spatial details. Performing optimization on both the latent code and feature maps yields near perfect reconstruction on real images.

In our work, we aim to have the best of both worlds: we wish to achieve this high reconstruction fidelity, while maintaining the speed and editing capacity of an encoder. Thus, we propose our *Feature-Style encoder*, which projects an image to a latent code $\mathbf{w} \in \mathcal{W}^+$, and a *feature code* $\mathbf{F} \in \mathcal{F} \subset \mathbb{R}^{h \times w \times c}$. This feature code is thus a tensor, and replaces the original GAN's feature map at a fixed layer indexed K of the generator. The parameters (h, w, c) correspond to the spatial size and the number of channels of the tensor, and depend on the layer K. The rest of the layers are controlled by the latent code $\mathbf{w}$. We now present this architecture in more detail.

3.2. Feature-Style encoder

Encoding The basic structure of our Feature-Style encoder is modelled on the classic approach used by most previous works on style-based GAN inversion [5,32,36,40,42], which employ a ResNet backbone. After this, as shown in Figure 2, we have two output branches: a latent prediction branch to encode the latent code $\mathbf{w} \in \mathcal{W}^+$, and a feature prediction branch to encode the feature code $\mathbf{F} \in \mathcal{F}$. The ResNet backbone contains four blocks, each downsampling the input feature maps by a factor of 2. Given an input image, we extract the feature maps after each block. In the latent branch, the four groups of feature maps are passed through an average pooling layer, concatenated and flattened to produce the latent prediction. This is then mapped to the latent code $\mathbf{w} = \{\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^N\}$. Each latent block $\mathbf{w}^i$ is generated from a different mapping network, expressed by a single fully connected layer. In the feature branch, the feature maps extracted after the penultimate block are passed through a convolutional network to encode the feature code $\mathbf{F}$ (see Figure 2). Let $\mathbf{G}^K(\mathbf{w})$ denote the feature maps at the K^{th} convolution layer of the generator. To generate the inversion, we replace $\mathbf{G}^K(\mathbf{w})$ with the feature code $\mathbf{F}$, and use the rest of the latent codes $\{\mathbf{w}^K, ..., \mathbf{w}^N\}$ to generate $\mathbf{G}(\mathbf{w}, \mathbf{F})$. We choose $K = 5$ for a balanced trade-off between the inversion quality and editing capacity, leading to $\mathcal{F} \subset \mathbb{R}^{16 \times 16 \times 512}$.

Editing In a style-based generator, the styles corresponding to coarse layers control high-level semantic attributes, the styles of the middle layers control smaller scale features, whereas the finer styles control micro structures. Given an input latent code $\mathbf{w}$, let us consider that we have a latent code $\tilde{\mathbf{w}}$ corresponding to a desired editing, where $\tilde{\mathbf{w}}$ is obtained from a latent space editing method [16, 33, 34]. To include the edits controlled by $\{\mathbf{w}^1, ..., \mathbf{w}^{K-1}\}$, it is necessary to modify the feature code. Thus we generate two images, $\mathbf{G}(\mathbf{w})$ and $\mathbf{G}(\tilde{\mathbf{w}})$, from $\mathbf{w}$ and $\tilde{\mathbf{w}}$, respectively. During generation, we extract the input features at the K^{th} convolution layer $\mathbf{G}^K(\mathbf{w})$ and $\mathbf{G}^K(\tilde{\mathbf{w}})$, compute the difference between them and add it to the encoded features $\mathbf{F}$. The modified feature $\tilde{\mathbf{F}}$ is determined as:

$$\tilde{\mathbf{F}} = \mathbf{F} + \mathbf{G}^K(\tilde{\mathbf{w}}) - \mathbf{G}^K(\mathbf{w}). \quad (1)$$

Then we generate the edited image $\mathbf{G}(\tilde{\mathbf{w}}, \tilde{\mathbf{F}})$ with $\tilde{\mathbf{w}}$ and the modified feature code $\tilde{\mathbf{F}}$.

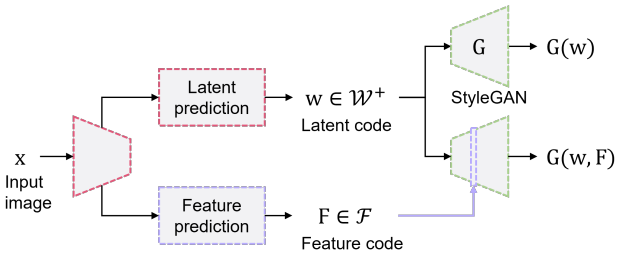


Figure 3. **Training approach.** During training, the encoder learns two inversions simultaneously - one generated with only the latent code, the other generated with both outputs. This design ensures the editing capacity of the learned latent code.

4. Training

4.1. Training data

Previous methods on GAN inversion [5, 32, 36, 40, 42] take only real image datasets as training data. However, compared with the synthetic images, the perceptual quality of the images resulting from the inversion is worse. An intuitive explanation is that there is a difference between the data distributions of real and synthetic images. The encoder is trained to project a given image to the extended latent space $\mathcal{W}^+$. If synthetic images are not viewed by the encoder, the resulting latent code may not perform as well as those of the original latent space. Therefore, we include both synthetic and real images as training data.

In the case of StyleGAN2 [21, 22], the generator accepts two inputs, a latent code and a group of noise maps for local variations. A synthetic image is generated from a random latent code and a group of random noises. During training, if the input is a synthetic image, we pass the ground truth noises into the generator, so that the encoder can focus on the information encoded by the latent code. If the input is a real image, we pass random noises into the generator. Note that in this case even if the latent code is perfectly inverted, local variations exist between the inversion and the input.

4.2. Losses

As shown in Figure 3, our Feature-Style encoder inverts an input image $\mathbf{x}$ to a latent code $\mathbf{w}$, and a feature code $\mathbf{F}$. To ensure the full editing capacity of the latent code, the encoder is trained on two inversions simultaneously - one generated with only the latent code $\tilde{\mathbf{x}}_1 = \mathbf{G}(\mathbf{w})$ and the other generated with both the feature code and the latent code $\tilde{\mathbf{x}}_2 = \mathbf{G}(\mathbf{w}, \mathbf{F})$.

Pixel-wise reconstruction loss In the case of a synthetic image, the reconstruction is measured using mean squared error (MSE) on $\tilde{\mathbf{x}}_1$ only. In this special case, the ground-truth latent code exists and the feature map is irrelevant. For real image input, the ground-truth latent code is unknown, so a per-pixel constraint may be too restrictive. The loss is expressed as:

$$\mathcal{L}_{mse} = ||\mathbf{G}(\mathbf{w}) - \mathbf{x}||_2. \tag{2}$$

Multi-scale perceptual loss (LPIPS) A common problem of the previous GAN inversion methods is the lack of sharpness of the reconstructed image, despite using the per-pixel MSE. To tackle this, we propose a multi-scale loss design which enables the reconstruction of finer details. Given an input image $\mathbf{x}$ and its inversion $\tilde{\mathbf{x}}$, a multi-scale LPIPS loss is defined as:

$$\mathcal{L}_{m\_lpips}(\tilde{\mathbf{x}}) = \sum_{i=0}^{2} ||\mathbf{V}(\lfloor \tilde{\mathbf{x}} \rfloor_i) - \mathbf{V}(\lfloor \mathbf{x} \rfloor_i)||_2, \tag{3}$$

where $\lfloor . \rfloor_i$ refers to downsampling by a scale factor 2^i and $\mathbf{V}$ denotes the feature extractor. This design allows the encoder to capture the perceptual similarities at different scales, which favors the perceptual quality of the inversion. This loss is applied on both inversions.

Feature reconstruction To ensure the possibility of using Eq.(1) to edit the feature code, $\mathbf{F}$ should be similar to the input feature maps at the K^{th} convolution layer in the generator, denoted by $\mathbf{G}^K(\mathbf{w})$. Therefore, we propose a feature reconstruction loss, which favors the encoded features to stay close to the original latent space. This term is defined as:

$$\mathcal{L}_{f\_recon} = ||\mathbf{F} - \mathbf{G}^K(\mathbf{w})||_2. \tag{4}$$

The total loss is defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{mse} + \lambda_1 \mathcal{L}_{m\_lpips} + \lambda_2 \mathcal{L}_{f\_recon}, \tag{5}$$

where $\lambda_1 = 0.2$ and $\lambda_2 = 0.01$ are weights balancing each loss.

Face Inversion For the inversion of a styleGAN model pre-trained on a face dataset, we adopt the multi-layer identity loss and the face parsing loss introduced by [42]. Given an input image $\mathbf{x}$ and its inversion $\tilde{\mathbf{x}}$, the multi-layer identity loss is defined as:

$$\mathcal{L}_{id}(\tilde{\mathbf{x}}) = \sum_{i=1}^{5} (1 - \langle \mathbf{R}_i(\tilde{\mathbf{x}}), \mathbf{R}_i(\mathbf{x}) \rangle), \tag{6}$$

where $\mathbf{R}$ is the pre-trained ArcFace network [11]. The multi-layer face parsing loss is defined as:

$$\mathcal{L}_{parse}(\tilde{\mathbf{x}}) = \sum_{i=1}^{5} (1 - \langle \mathbf{P}_i(\tilde{\mathbf{x}}), \mathbf{P}_i(\mathbf{x}) \rangle), \tag{7}$$

where $\mathbf{P}$ is a pre-trained face parsing model [51]. These two above-mentioned losses are applied on both inversions. Hence the total loss for face inversion is:

$$\mathcal{L}_{face} = \mathcal{L}_{total} + \lambda_3 \mathcal{L}_{id} + \lambda_4 \mathcal{L}_{parsing}, \tag{8}$$

where $\lambda_3 = 0.1$ and $\lambda_4 = 0.1$ are weights balancing the identity preserving and face parsing terms.

5. Experiments

In this section, we present the experimental setup and compare our method with state-of-the-art GAN inversion methods. We conduct the evaluation from two aspects: inversion quality and editing capacity. We also show results on videos as well as ablative studies.

5.1. Experimental setup

We evaluate our framework on several style-based generators pre-trained on various domains. We train the encoder for the inversion of StyleGAN2 [22] on faces and cars, and StyleGAN2-Ada [19] on cats and dogs. For each generator pre-trained on a specific domain, a separate encoder is trained. During the training, we use a batch size of 4, each batch containing two real images and two synthetic images. The model is trained for 12 epochs, using $10K$ iterations per epoch. The learning rate is 10^{-4} for the first 10 epochs and is divided by ten for the last 2 epochs. For the face domain, we minimize Eq.(8), using FFHQ [21] for training, and CelebA-HQ [18] for evaluation. For the car domain, we minimize Eq.(5), using Stanford Cars [24] training set for training, and the corresponding test set for evaluation. For the cat/dog domain, we minimize Eq.(5), using AFHQ Cats/Dogs [9] train set for training, and the corresponding test set for evaluation.

5.2. Inversion

We evaluate our model against the current state-of-the-art encoder-based GAN inversion methods: pSp [32], e4e [36], restyle [5] and a recent preprint work HFGI [40]. We first perform comparisons for the inversion of StyleGAN2 model pre-trained on FFHQ dataset. For each method we use the official implementation [4, 31, 37, 41] to generate the results. Restyle has been implemented on both pSp and e4e. We use restyle-pSp as it has better performance.

Qualitative Results Figure 4 shows the inversion results of the different methods. Overall, visual inspection shows that our method outperforms other models. Firstly, faces

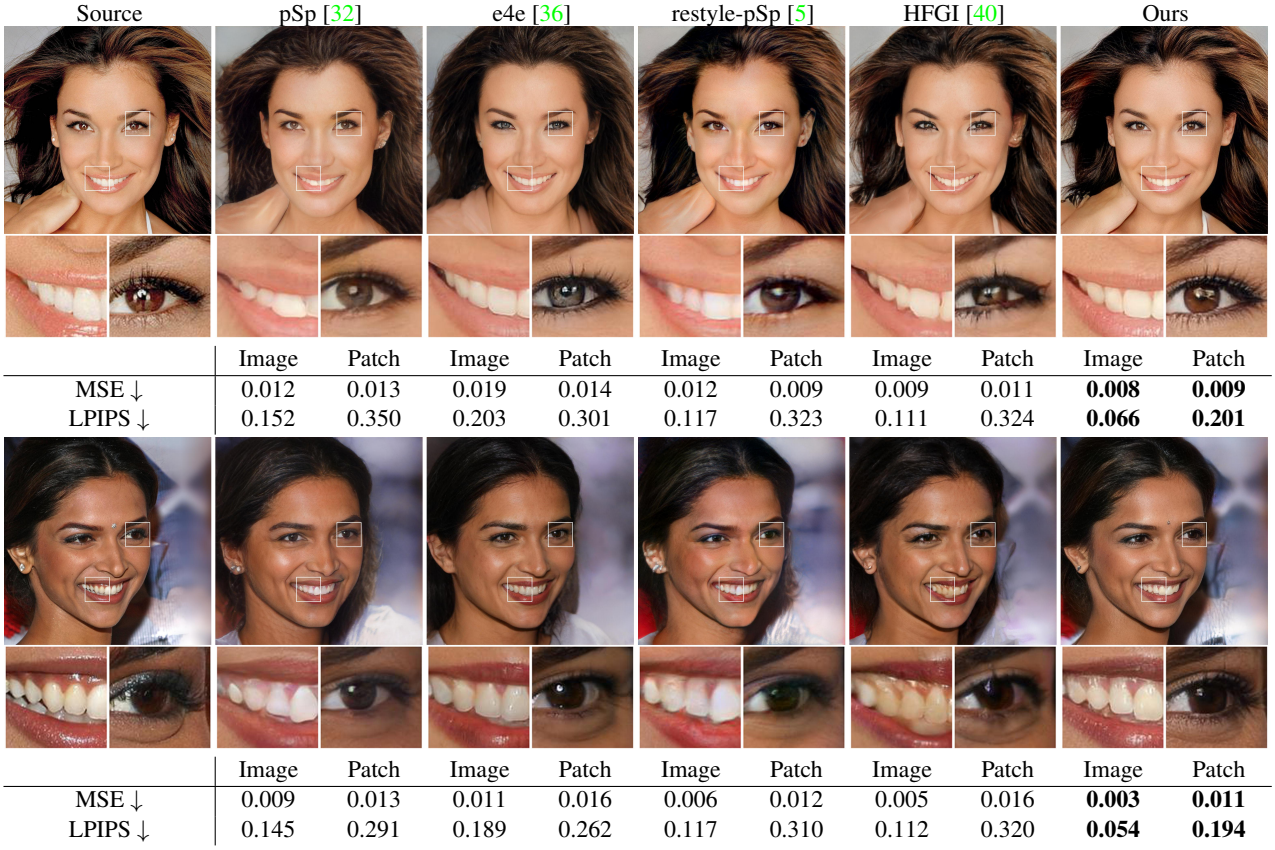


	Image	Patch	Image	Patch	Image	Patch	Image	Patch	Image	Patch
MSE ↓	0.012	0.013	0.019	0.014	0.012	0.009	0.009	0.011	**0.008**	**0.009**
LPIPS ↓	0.152	0.350	0.203	0.301	0.117	0.323	0.111	0.324	**0.066**	**0.201**

	Image	Patch	Image	Patch	Image	Patch	Image	Patch	Image	Patch
MSE ↓	0.009	0.013	0.011	0.016	0.006	0.012	0.005	0.016	**0.003**	**0.011**
LPIPS ↓	0.145	0.291	0.189	0.262	0.117	0.310	0.112	0.320	**0.054**	**0.194**

Figure 4. **Inversion on face domain**. We compare our model against state-of-the-art encoder-based methods [5,32,36,40] for the inversion of StyleGAN2 pre-trained on face domain. Reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness. Pixel-wise reconstruction errors (MSE error, lower is better) and perceptual quality (LPIPS distance, lower is better) confirm this visual conclusion on these examples.

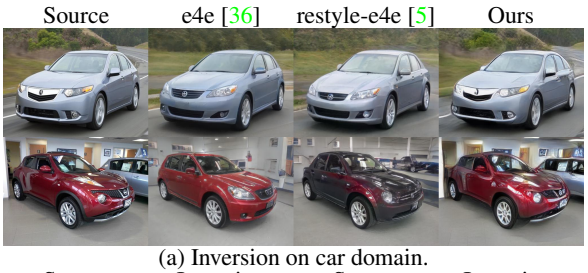
Method	SSIM ↑	PSNR ↑	MSE ↓	LPIPS ↓	ID ↑	FID ↓
IDGI [49]	0.554	22.06	0.034	0.136	0.480	36.83
pSp [32]	0.509	20.37	0.040	0.159	0.654	34.68
e4e [36]	0.479	19.17	0.052	0.196	0.593	36.72
restyle [5]	0.537	21.14	0.034	0.130	0.735	32.56
HFGI [40]	0.595	22.07	0.027	0.117	0.758	26.53
Ours	**0.641**	**23.65**	**0.019**	**0.066**	**0.867**	**19.03**

Table 1. **Quantitative evaluation.** We use *SSIM*, *PSNR* and *MSE* to measure the reconstruction error, and *LPIPS* [48] for the perceptual quality. We also measure the *identity similarity* (ID) between the inversion and the source image, which refers to the cosine similarity between the features in ArcFace [11] of both images. To measure the discrepancy between the real data distribution and the inversion one, we use *FID* [13]. Overall, our method outperforms the state-of-the-art baselines by up to $10\% - 20\%$. In terms of perceptual quality (LPIPS), we improve the result by 50%.

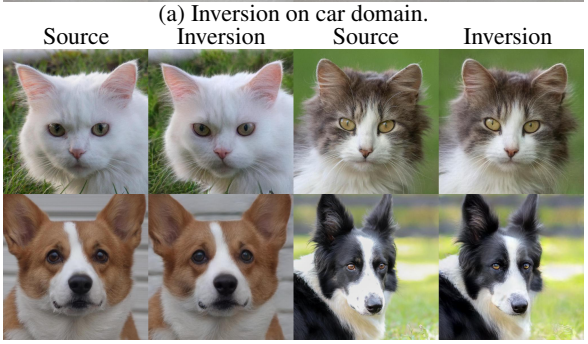
are more faithfully reconstructed globally. Secondly, zoom-in patches show that more details are preserved and that the images produced by our framework are significantly sharper than those of the concurrent methods.

Quantitative Evaluation We evaluate our approach quantitatively against the aforementioned encoder based methods [5, 32, 36, 40] and a hybrid method (in-domain GAN) [49]. We compare each method on the inversion of StyleGAN2 pretrained on FFHQ, using the first $1K$ images of CelebA-HQ as evaluation data. To measure the reconstruction error, we compute *SSIM*, *PSNR* and *MSE*. To measure the perceptual quality, we measure the *LPIPS* [48] distance. Additionally, we measure the *identity similarity* (ID) between the inversion and the source image, which refers to the cosine similarity between the features in ArcFace [11] of the two images. To measure the discrepancy between the real data distribution and the inversion one, we use the Frechet Inception Distance [13] (*FID*). Table 1 presents the quantitative evaluation of all the methods. Our method significantly outperforms the state-of-the-art methods on all the metrics. In terms of perceptual quality (LPIPS), improvement can attain 50%.

Inversion for other domains Figure 5(a) shows the inversion for StyleGAN2 pretrained on the car domain. We train the encoder with Stanford Car dataset [24]. Compared with e4e [36] and restyle-e4e [5], our inversion achieves



(a) Inversion on car domain.



(b) Inversion on cat/dog domain.

Figure 5. **Inversion on other domains.** In (a), we show the inversion results of StyleGAN2 pre-trained on car domain. Our method captures better the details than e4e [36] and restyle-e4e [5]. In (b), we show the inversion results of StyleGAN2-Ada pre-trained on the cat and dog domains, respectively.

a better reconstruction quality, preserving better the details. Figure 5(b) shows the inversion for StyleGAN2-Ada pretrained on AFHQ Cat/Dog dataset [9]. Our encoder achieves nearly perfect inversions. Here we did not compare with [5, 36], as the official pre-trained model is unavailable.

5.3. Editing

In this experiment, we consider the task of real image editing via latent space manipulation. We compare our approach with the state-of-the-art encoder-based GAN inversion methods [5, 32, 36, 40] on the facial image editing via the latent space of StyleGAN2 pretrained on FFHQ dataset. Despite the fact that all the encoders project real images to the latent space of StyleGAN, the latent distribution learned by each encoder can be different. As such, for each inversion model, we generate the inverted latent codes for the first $10K$ images of CelebA-HQ, and apply InterFaceGAN [33] to find the editing directions in the learned latent space. Figure 6 shows facial attribute editing results for all methods. Compared with the state-of-the-art, our method yields visually plausible editing results, while preserving better the identity and sharpness.

To evaluate quantitatively the editing results of each method, we take the first $1K$ images of CelebA-HQ as testing data. For each method, we project each image to the latent space and apply InterFaceGAN [33] to generate the editing result on the following facial attributes: 'gender', 'makeup', 'smiling' and 'eyeglasses'. Then we compute

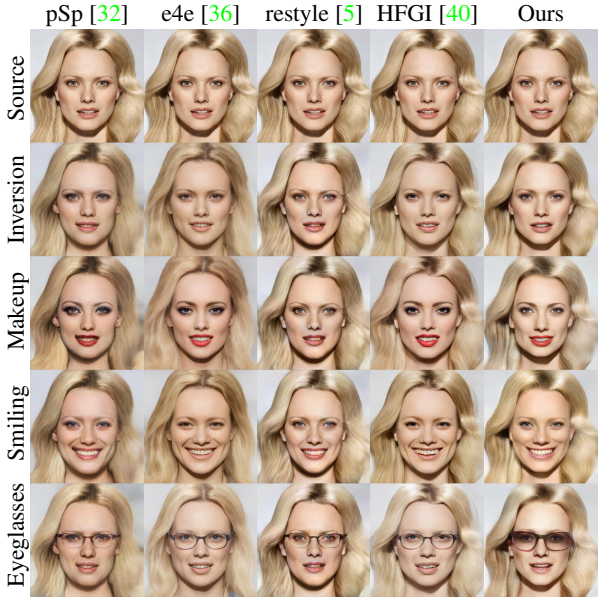


Figure 6. **Latent space editing**. For each method, we apply InterFaceGAN [33] to perform latent editing for facial attribute manipulation. Our method yields plausible editing results, while at the same time preserving better the identity and the sharpness.

Method	pSp [32]	e4e [36]	restyle [5]	HFGI [40]	Ours
FID↓	39.66	37.79	32.84	30.68	**27.45**

Table 2. **FID measured on edited images.** For each inversion method, we encode the first $1K$ images of CelebA-HQ to the latent space and perform latent editing on four facial attributes. The FID is calculated between the real images and all the edited images. Our method outperforms the other approaches.

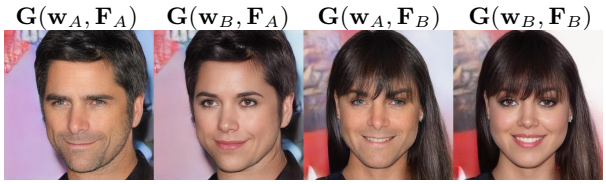


Figure 7. **Style mixing.** The first and last column show the inversions of two images $\mathbf{x}_A$ and $\mathbf{x}_B$, denoted by $\mathbf{G}(\mathbf{w}_A, \mathbf{F}_A)$ and $\mathbf{G}(\mathbf{w}_B, \mathbf{F}_B)$, respectively. The second column is generated from the feature code of $\mathbf{x}_A$ and the latent code of $\mathbf{x}_B$, denoted by $\mathbf{G}(\mathbf{w}_B, \mathbf{F}_A)$, and vice versa for the third column, denoted by $\mathbf{G}(\mathbf{w}_A, \mathbf{F}_B)$. The feature code encodes the geometric structures such as pose and facial shape, whereas the latent code controls the appearance styles like eye color and makeup.

the FID between the real images and all the edited images. Table 2 shows that the discrepancy between the data distribution of our editing results and that of the real images is the smallest by a large margin.

Additionally, we show style mixing results in Figure 7, generated from the latent code of one image with the feature code of another image. From this experiment we observe

Figure 8. **Video inversion**. For each method, we show the inversion results of multiple consecutive images extracted from a video sequence. Our inversion method preserves better the identity along the video and yields a better reconstruction for the extreme poses.

Method	SSIM ↑	PSNR ↑	MSE ↓	LPIPS ↓	ID ↑
pSp [32]	0.736	22.30	0.025	0.196	0.687
e4e [36]	0.713	20.57	0.037	0.220	0.620
restyle-pSp [5]	0.761	23.17	0.021	0.189	0.781
HFGI [40]	0.783	24.04	0.017	0.182	0.810
Ours	**0.818**	**26.64**	**0.009**	**0.122**	**0.895**

Table 3. **Quantitative evaluation on video inversion.** We sample randomly 120 videos from RAVDESS dataset [27], perform the inversion using each method and compute the quantitative metrics. Our method outperforms the competing approaches on both the reconstruction error and the perceptual quality.

that the geometric structures such as pose and facial shape are encoded by the feature code, while the appearance styles like eye color and makeup are encoded by the latent code.

5.4. Video inversion

In this section, we discuss the possibility of integrating our proposed encoder into a video editing pipeline. We compare the inversion quality and stability of different encoders on videos. Figure 8 shows a qualitative inversion result on consecutive images extracted from a video sequence. The last two frames in the original sequence are extreme poses. As can be observed, other methods fail to invert non-frontal poses, thus damaging the consistent reconstruction along the sequence. Our approach yields consistent inversion of high fidelity, which favors further editing on videos.

Configurations	SSIM ↑	PSNR ↑	LPIPS ↓	ID ↑	FID ↓
(A) w/o $\mathcal{L}_{m\_lpips}$	0.647	24.01	0.056	0.874	22.63
(B) w/o feature input	0.489	19.44	0.192	0.635	35.28
(C) w/o synthetic data	0.644	23.67	0.065	0.873	20.45
(D) our baseline	0.641	23.65	0.066	0.867	19.03

Table 4. **Ablative study on experimental setup.** We conduct experiments on three different configurations: (1) w/o multi-scale setting in the perceptual loss, (2) w/o feature prediction branch, (3) w/o synthetic training data. Our baseline achieves better perceptual quality and comparable performance on the distortion metrics.

We evaluate our encoder quantitatively against the state-of-the-art for video inversion on RAVDESS [27], a dataset of talking face videos. From which we sample randomly 120 videos as evaluation data. For each method, we perform the inversion on each video and compute the quantitative metrics on the inversion results. As shown in Table 3, our approach outperforms the competing approaches on both the reconstruction error and the perceptual quality.

5.5. Ablation study

We conduct an ablation study on the experimental setup for the inversion of StyleGAN2 pretrained on FFHQ. Specifically, we compare the quantitative metrics of several ablative configurations. As shown in Table 4, in configuration (A), we replace the multi-scale perceptual loss by a common LPIPS loss, and change the corresponding weight λ_1 to scale it to a similar magnitude as before. In (B), we discard the feature prediction branch and generate the inversion with only the latent code. In (C), we use only real images as training data.

For (A), we observe a comparable result on the distortion metrics, but a much higher FID compared to the baseline. Including the proposed multi-scale perceptual loss greatly improves the perceptual quality and presents comparable performance on other distortion metrics. (B) confirms that the feature code helps to generate an inversion with better fidelity. For (C), we observe similar performance on the distortion metrics but a higher FID value. This demonstrates that including synthetic data in the training helps improving the perceptual quality of the inversion results.

6. Conclusion

We have proposed a new *Feature-Style encoder* architecture for style-based GAN inversion. For the first time, we achieve projection of a given image to the Feature-Style latent space of a style-based generator, without out any optimization step at inference time. Our approach significantly improves the perceptual quality of the inversion, outperforming the strong and competitive state-of-the-art methods. Additionally, we show that our proposed encoder is more suited for the inversion and editing of videos.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4432–4441, 2019. 2, 3

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8296–8305, 2020. 1, 2, 3, 11, 13, 14, 15, 16, 17

[3] Rameen Abdal, Peihao Zhu, Niloy Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *arXiv e-prints*, pages arXiv–2008, 2020. 2

[4] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Official implementation of restyle: A residual-based stylegan encoder via iterative refinement. https://github.com/yuval-alaluf/restyle-encoder, 2021. 5

[5] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6711–6720, 2021. 1, 2, 4, 5, 6, 7, 8, 11, 13, 14, 15, 16, 17

[6] Yazeed Alharbi and Peter Wonka. Disentangled image generation through structured noise injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5134–5142, 2020. 2

[7] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 2

[8] Lucy Chai, Jun-Yan Zhu, Eli Shechtman, Phillip Isola, and Richard Zhang. Ensembling with deep generative views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14997–15007, 2021. 2

[9] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8188–8197, 2020. 5, 7

[10] Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5771–5780, 2020. 2, 12

[11] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019. 5, 6, 11

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 1

[13] 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. *Advances in neural information processing systems*, 30, 2017. 6

[14] Xianxu Hou, Xiaokang Zhang, Hanbang Liang, Linlin Shen, Zhihui Lai, and Jun Wan. Guidedstyle: Attribute knowledge guided style manipulation for semantic face editing. *Neural Networks*, 2021. 2

[15] Minyoung Huh, Richard Zhang, Jun-Yan Zhu, Sylvain Paris, and Aaron Hertzmann. Transforming and projecting images into class-conditional generative networks. In *European Conference on Computer Vision*, pages 17–34. Springer, 2020. 2

[16] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. In *Proc. NeurIPS*, 2020. 1, 2, 4

[17] Kyoungkook Kang, Seongtae Kim, and Sunghyun Cho. Gan inversion for out-of-range images with geometric transformations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13941–13949, 2021. 1, 2, 3, 4

[18] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 2, 5

[19] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Proc. NeurIPS*, 2020. 2, 5

[20] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Proc. NeurIPS*, 2021. 1, 2, 3, 11, 18

[21] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 1, 2, 3, 4, 5, 11

[22] 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, 3, 4, 5

[23] Hyunsu Kim, Yunjey Choi, Junho Kim, Sungjoo Yoo, and Youngjung Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 852–861, 2021. 2

[24] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013. 5, 6

[25] Gihyun Kwon and Jong Chul Ye. Diagonal attention and style-based gan for content-style disentanglement in image generation and translation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13980–13989, 2021. 2

[26] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision

semantic image editing. *arXiv preprint arXiv:2111.03186*, 2021. 2

[27] Steven R Livingstone and Frank A Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391, 2018. 8, 11

[28] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7198–7211. Curran Associates, Inc., 2020. 2

[29] 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*, pages 2085–2094, 2021. 2

[30] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14104–14113, 2020. 2

[31] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Official implementation of encoding in style: a stylegan encoder for image-to-image translation. https://github.com/eladrich/pixel2style2pixel, 2020. 5

[32] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2287–2296, 2021. 1, 2, 3, 4, 5, 6, 7, 8, 11, 13, 14, 15, 16, 17

[33] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020. 1, 2, 4, 7, 11, 19

[34] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1532–1540, 2021. 1, 2, 4, 11, 19, 20, 21, 22

[35] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6142–6151, 2020. 2

[36] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 1, 2, 3, 4, 5, 6, 7, 8, 11, 13, 14, 15, 16, 17

[37] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Official implementation of designing an encoder for stylegan image manipulation. https://github.com/omertov/encoder4editing, 2021. 5

[38] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *International Conference on Machine Learning*, pages 9786–9796. PMLR, 2020. 2

[39] Binxu Wang and Carlos R Ponce. The geometry of deep generative image models and its applications. *arXiv preprint arXiv:2101.06006*, 2021. 2

[40] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. High-fidelity gan inversion for image attribute editing. *arXiv preprint arXiv:2109.06590*, 2021. 1, 2, 3, 4, 5, 6, 7, 8, 11, 13, 14, 15, 16, 17

[41] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. Official implementation of high-fidelity gan inversion for image attribute editing. https://github.com/Tengfei-Wang/HFGI, 2021. 5

[42] Tianyi Wei, Dongdong Chen, Wenbo Zhou, Jing Liao, Weiming Zhang, Lu Yuan, Gang Hua, and Nenghai Yu. A simple baseline for stylegan inversion. *arXiv preprint arXiv:2104.07661*, 2021. 2, 4, 5

[43] Zongze Wu, Dani Lischinski, and Eli Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12863–12872, 2021. 1

[44] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *arXiv preprint arXiv:2101.05278*, 2021. 2

[45] Yangyang Xu, Yong Du, Wenpeng Xiao, Xuemiao Xu, and Shengfeng He. From continuity to editability: Inverting gans with consecutive images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13910–13918, 2021. 2

[46] Xu Yao, Alasdair Newson, Yann Gousseau, and Pierre Hellier. A latent transformer for disentangled face editing in images and videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13789–13798, 2021. 2

[47] Cheng Yu and Wenmin Wang. Adaptable gan encoders for image reconstruction via multi-type latent vectors with two-scale attentions. *arXiv preprint arXiv:2108.10201*, 2021. 2

[48] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 1, 6

[49] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *European Conference on Computer Vision*, pages 592–608. Springer, 2020. 1, 2, 3, 6, 11, 13, 14, 15, 16, 17

[50] Peihao Zhu, Rameen Abdal, John Femiani, and Peter Wonka. Barbershop: Gan-based image compositing using segmentation masks. *arXiv preprint arXiv:2106.01505*, 2021. 1, 2, 3, 4

[51] zllrunning. Face parsing network pre-trained on celebamask-hq dataset. https://github.com/zllrunning/face-parsing.PyTorch, 2019. 5

A. Inversion

We show additional visual results for the inversion of StyleGAN2 pre-trained on face domain in Figs. 10 to 14. We compare our model against an optimization based method [2], state-of-the-art encoder based methods [5, 32, 36, 40] and a hybrid method [49]. As can be observed, reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness.

Alias-free GAN We show preliminary inversion results of the third generation on StyleGAN - recently released (one month ago) Alias-free GAN [20] pretrained on FFHQ [21]. Compared with StyleGAN2, the architecture of Alias-free GAN has several important changes. First, the input tensor passed into the generator is no longer constant, but synthesized from the latent code. The spatial size of the input tensor is increased from 4×4 to 36×36. Additionally, the noise inputs are discarded. As shown in Figure 15, despite the architectural changes, our proposed encoder still yields satisfying inversion results.

B. Editing

Latent Space Editing We show additional facial attribute editing results in Figure 16. The latent editing directions are computed using InterFaceGAN [33], except the last attribute 'pose', computed with SeFa [34].

Editing on Other Domains In the main paper, we have presented the inversion results for StyleGAN2 pretrained on the car domain and StyleGAN2-Ada pretrained on cat and dog domain. Here we present additional editing results. Figure 17 shows editing results on car domain. Figure 18 and Figure 19 show the editing results on cat and dog domain, respectively. All the latent editing directions are computed with SeFa. Our model yields satisfying editing results on other domains.

Style Mixing We show additional style mixing results in Figure 20. The style mixing is generated from the latent code of one image with the feature code of another image. These additional results confirm that the geometric structures such as pose and facial shape are encoded by the feature code, while the appearance styles like eye color and makeup are encoded by the latent code.

C. Video results

Inversion Consistency Additionally, to evaluate the consistency of the inversion, we propose a new metric *Identity Consistency*, which refers to the averaged identity similarity between the reconstructed frame $\tilde{\mathbf{x}}_i$ and frame $\tilde{\mathbf{x}}_0$ along a video sequence:

$$IC = \frac{1}{N}\sum_{i=1}^{N}\langle \mathbf{R}(\tilde{\mathbf{x}}_i), \mathbf{R}(\tilde{\mathbf{x}}_0)\rangle, \tag{9}$$

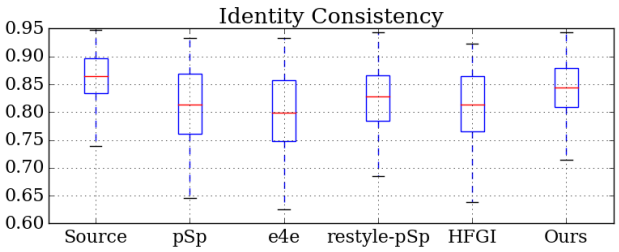


Figure 9. **Identity consistency of video inversion**. For each method, we compute the proposed metric *identity consistency* for each inverted video and plot the results in a box-plot. Our averaged identity consistency is the closest to that of the source videos.

where $\mathbf{R}$ is the pre-trained ArcFace [11] network. We compute this metric for our encoder and state-of-the-art methods for video inversion on RAVDESS [27]. From this dataset we sample randomly 120 videos as evaluation data. For each method, we perform the inversion and compute this metric on each inverted video and present the results with a box-plot. Figure 9 shows that the averaged identity consistency of our inversion is the closest to that of the source, which proves the stability of our inversion.

D. Ablation study

Visual Results In the main paper, we have conducted an ablation study on the experimental setup using quantitative metrics. We have compared the following configurations:

(A) w/o multi-scale setting in the perceptual loss

(B) w/o feature prediction branch

(C) w/o synthetic training data

(D) our baseline

We show additional qualitative results of these ablative configurations in Figure 21. Compared with our baseline, the inversion of configuration (A) is less sharp and reconstructs less well the details. Configuration (B) fails to achieve a plausible reconstruction. The inversion of configuration (C) is globally plausible, yet less reliable in details. For instance, the teeth are less photo-realistic compared with our baseline. This qualitative comparison confirms the quantitative evaluation in the main paper.

Choice of K We provide an additional ablation study on the choice of feature code insertion layer K. We compared $K = 4$, $K = 6$ and $K = 7$ with our baseline $K = 5$. A different model is trained for each configuration. Figure 22 shows the qualitative results of inversion and style mixing. We observe that using $K = 4$ yields good style mixing effects but lower reconstruction quality. While choosing $K = 6$ or 7 generates perfection reconstruction, the style mixing effects is less obvious. Using $K = 6$ or 7 encodes nearly all the information in the feature code, thus limited

in editing. Our choice of $K = 5$ holds a balanced trade-off between editing capacity and reconstruction quality.

E. Limitations

In this section, we discuss the limitations of our proposed method. Our encoder learns to project an image to a feature code and a latent code. To perform latent space editing, it is necessary to modify the feature code using equation (1) in the main paper. We have noticed that in some cases, this may be not sufficient. For instance in Figure 16, the editing results of attribute 'gender' is less satisfying when the original person has long hairs. In the future, it could be helpful to study further improvements for the feature code editing, *e.g.*, by including masks for interested area, or modifying only the relevant channels to achieve local editing [10].

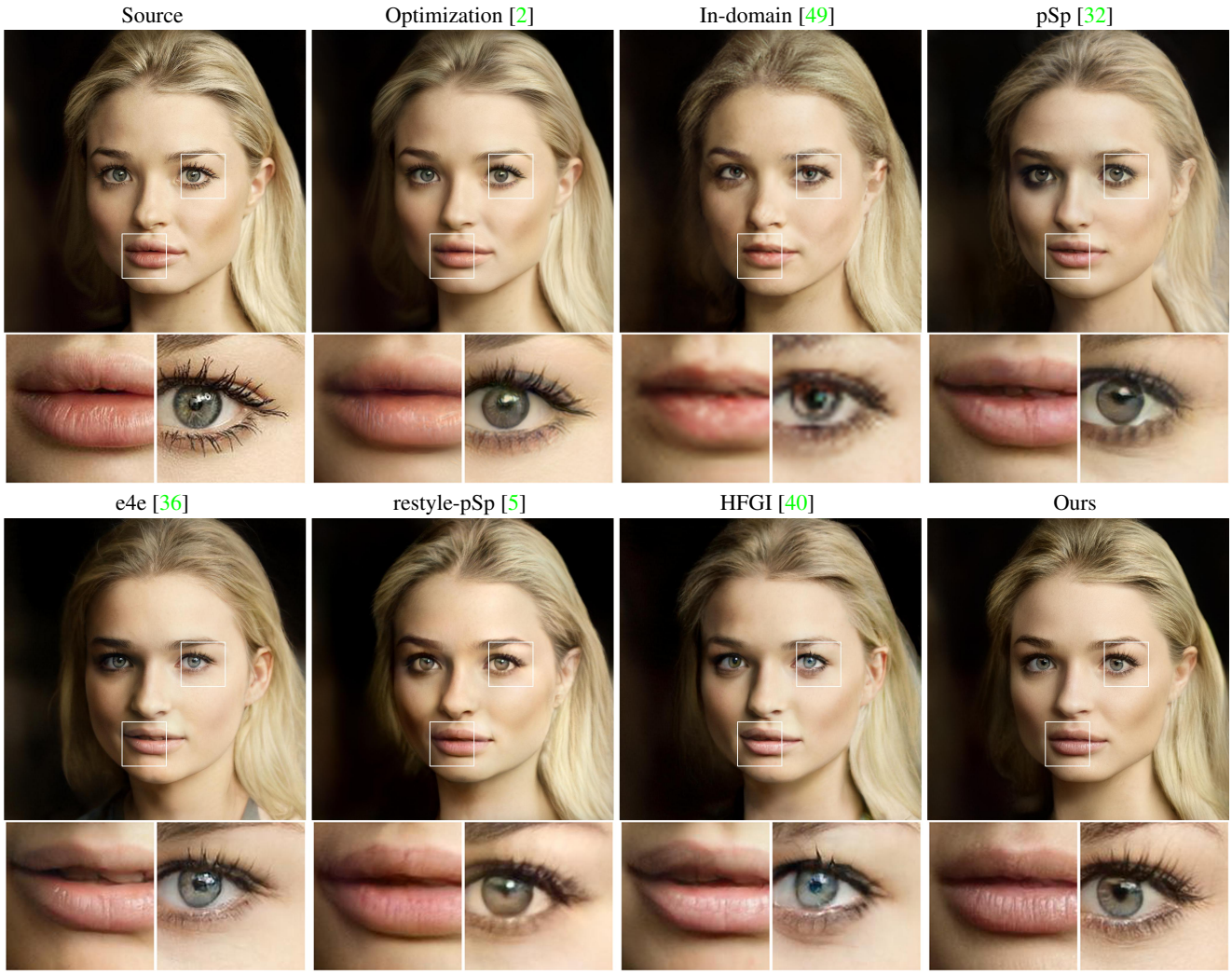


Figure 10. **Inversion on face domain**. We compare our model against state-of-the-art GAN inversion methods [2, 5, 32, 36, 40, 49] for the inversion of StyleGAN2 pre-trained on face domain. Reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness.

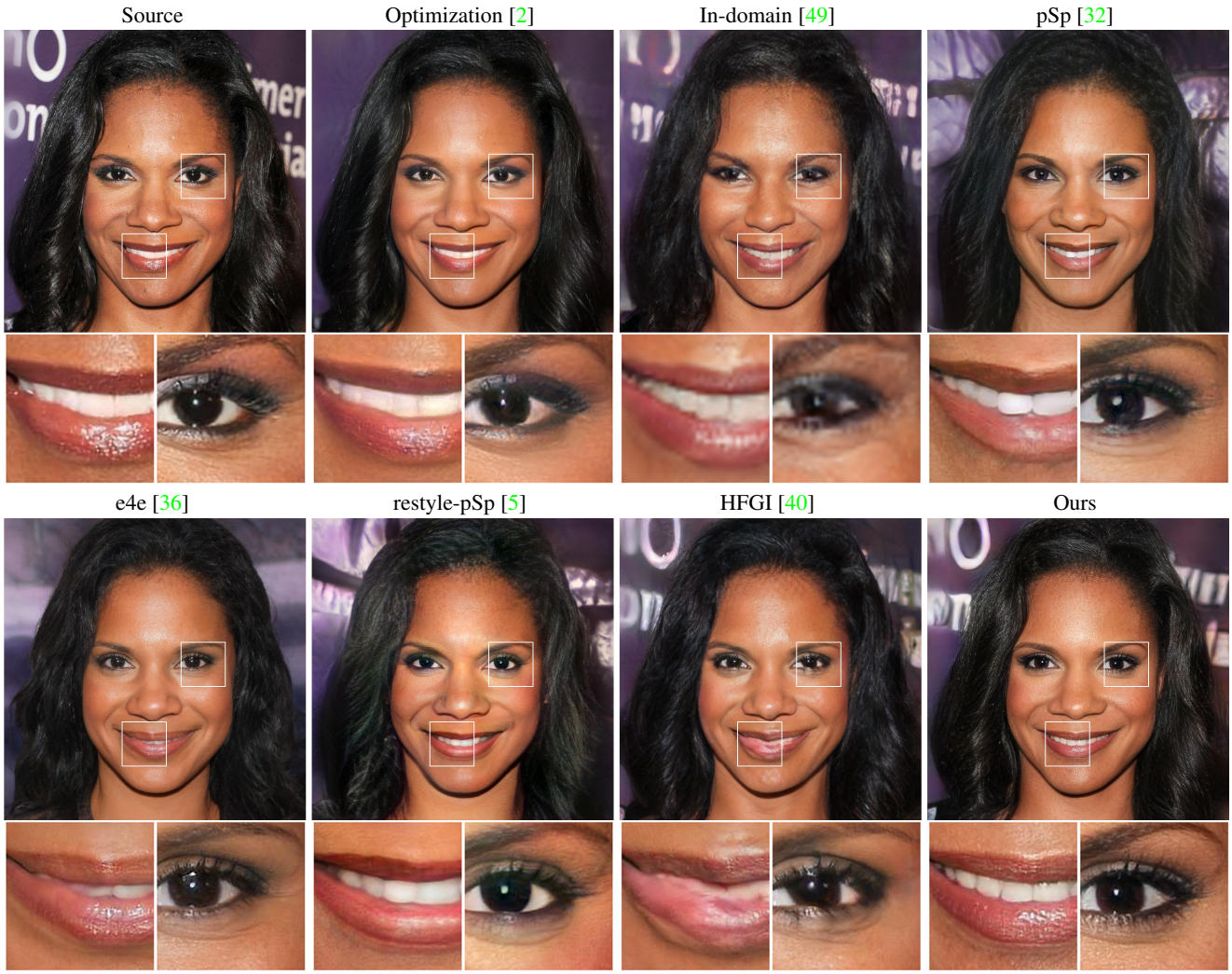


Figure 11. **Inversion on face domain**. We compare our model against state-of-the-art GAN inversion methods [2, 5, 32, 36, 40, 49] for the inversion of StyleGAN2 pre-trained on face domain. Reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness.

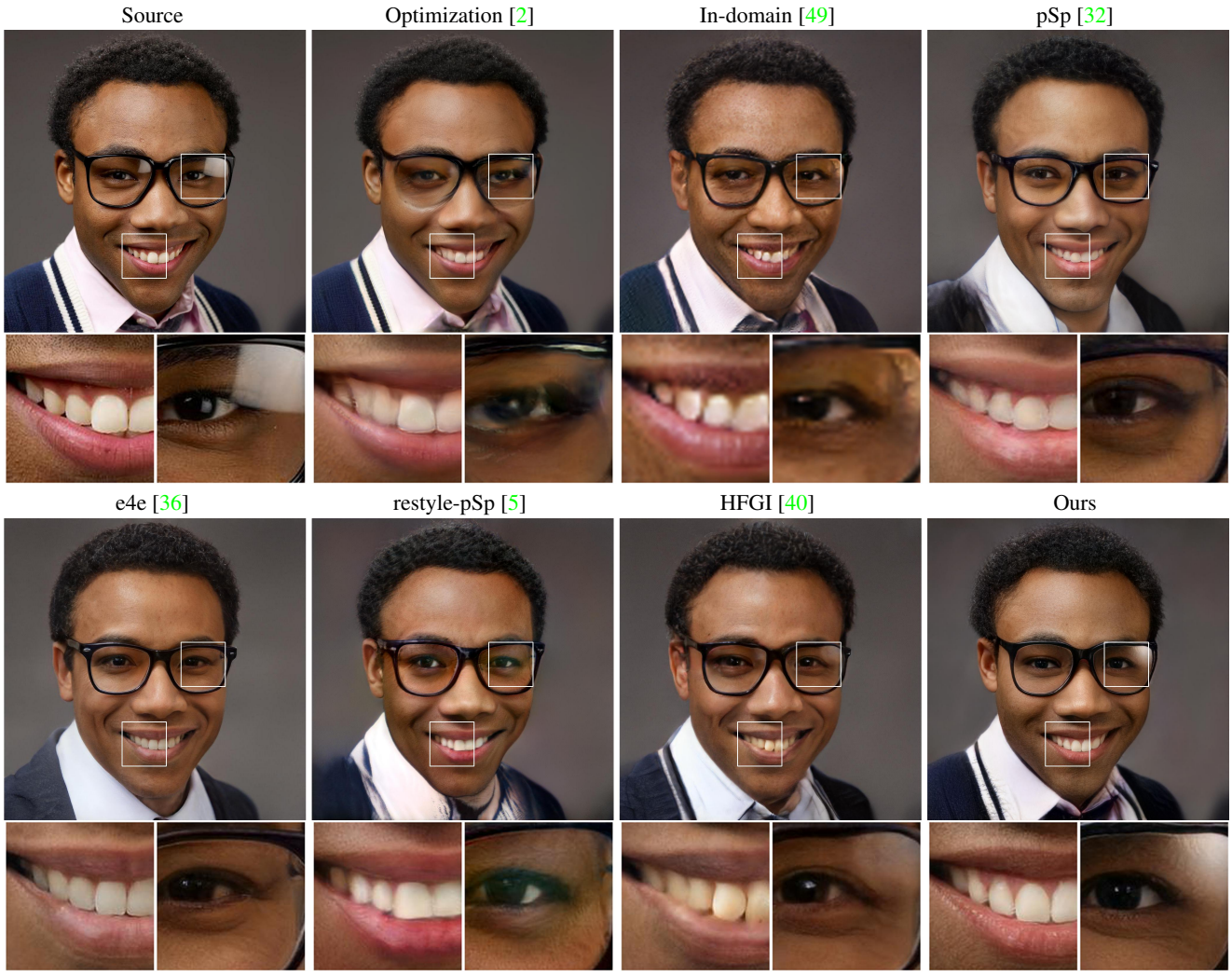


Figure 12. **Inversion on face domain**. We compare our model against state-of-the-art GAN inversion methods [2, 5, 32, 36, 40, 49] for the inversion of StyleGAN2 pre-trained on face domain. Reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness.

Figure 13. **Inversion on face domain**. We compare our model against state-of-the-art GAN inversion methods [2, 5, 32, 36, 40, 49] for the inversion of StyleGAN2 pre-trained on face domain. Reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness.

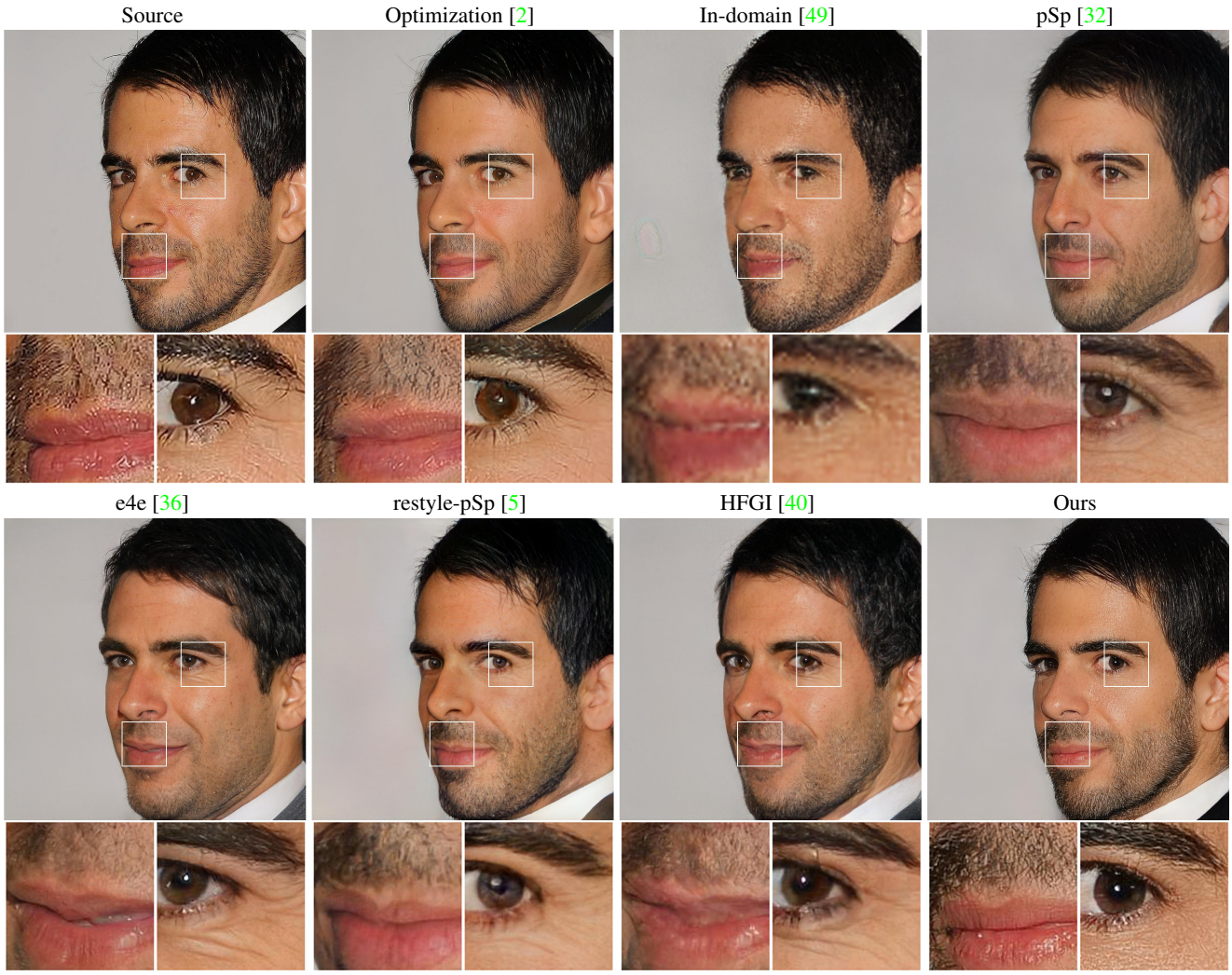


Figure 14. **Inversion on face domain**. We compare our model against state-of-the-art GAN inversion methods [2, 5, 32, 36, 40, 49] for the inversion of StyleGAN2 pre-trained on face domain. Reconstructions using our framework are visually more faithful and zoom-in patches show that they exhibit much more details and sharpness.

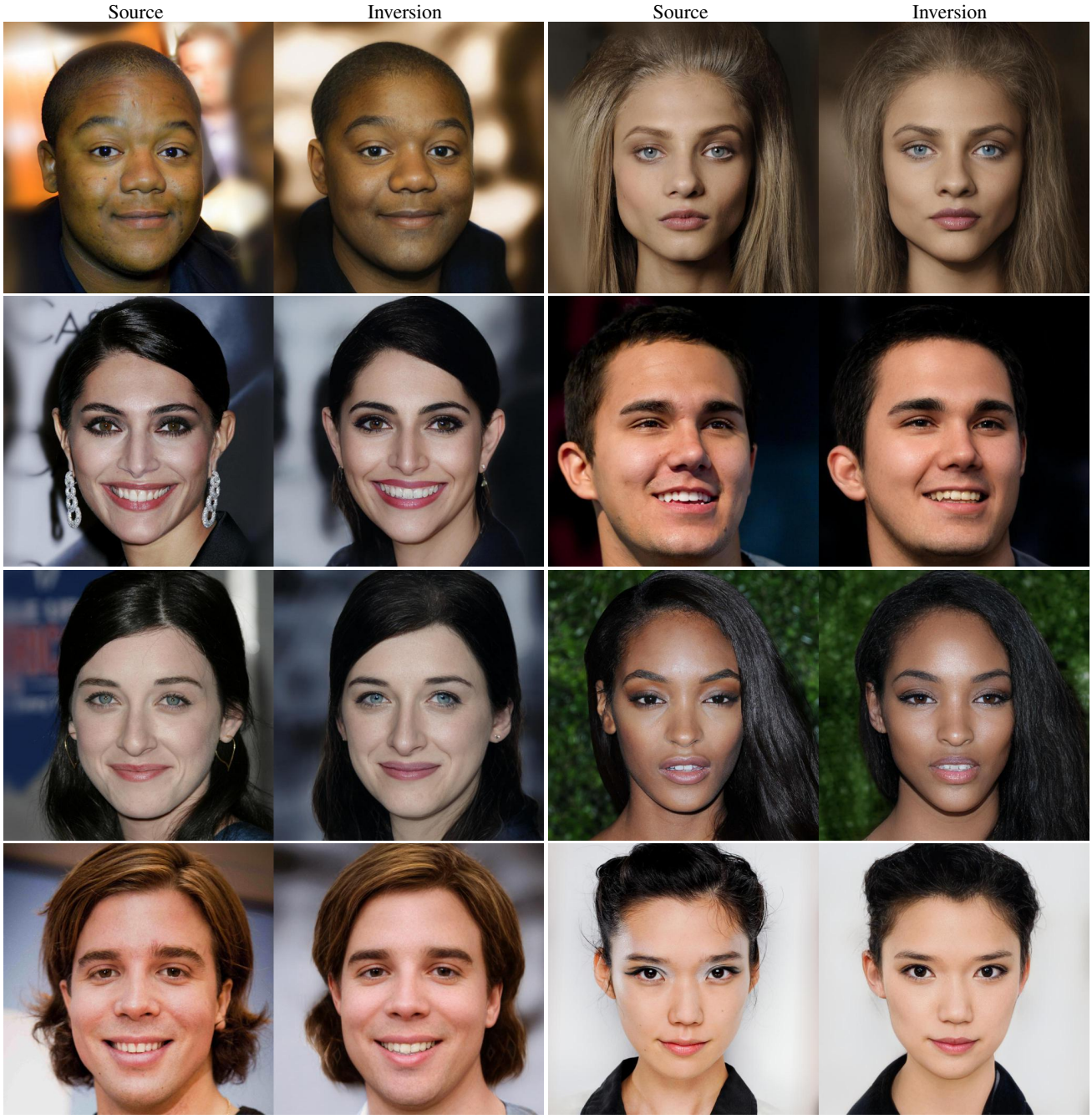


Figure 15. **Inversion of Alias-free GAN.** We show preliminary inversion results of the 3rd generation of StyleGAN - recent released (one month ago) Alias-free GAN [20] pretrained on face domain. Compared with StyleGAN2, the architecture of Alias-free GAN has several important changes. Despite the architectural changes, our proposed encoder still yields satisfying inversion results.

Figure 16. **Editing on face domain**. We show additional facial attribute editing results. The latent editing directions are computed using InterFaceGAN [33], except the last attribute 'pose', computed with SeFa [34].

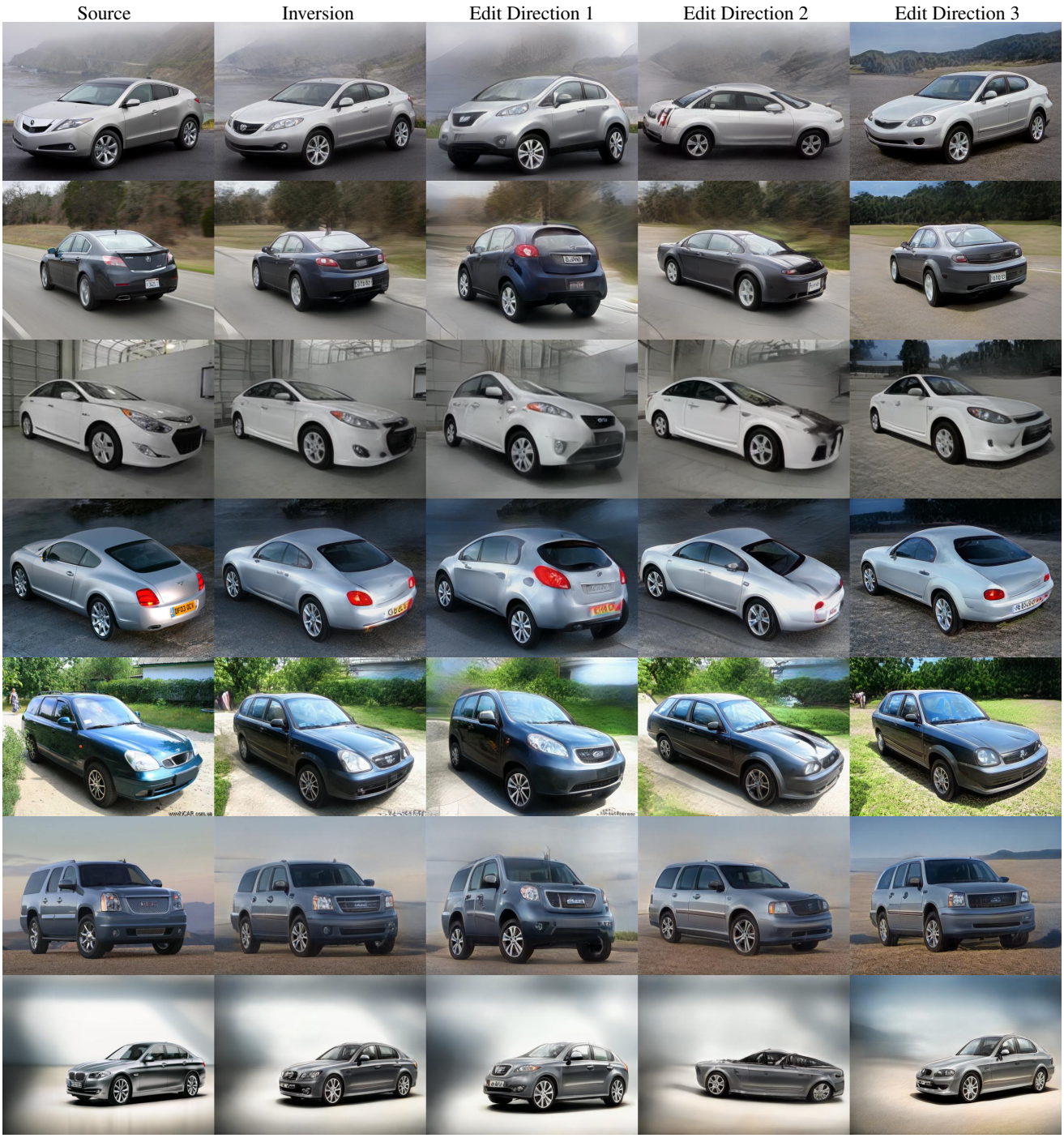


Figure 17. **Editing on car domain**. We show latent space editing results on car domain. We compute the latent editing directions with SeFa [34]. The first column is the source image, second column is our inversion result, the third to last column correspond to the semantic directions found with SeFa [34]. Our model yields satisfying editing results on car domain.

Figure 18. **Editing on cat domain**. We show latent space editing results on cat domain. We compute the latent editing directions with SeFa [34]. The first column is the source image, second column is our inversion result, the third to last column correspond to the semantic directions found with SeFa [34]. Our model yields satisfying editing results on cat domain.

Figure 19. **Editing on dog domain**. We show latent space editing results on dog domain. We compute the latent editing directions with SeFa [34]. The first column is the source image, second column is our inversion result, the third to last column correspond to the semantic directions found with SeFa [34]. Our model yields satisfying editing results on dog domain.

Figure 20. **Style mixing of feature code and latent code.** The first and last column show the inversions of two images $\mathbf{x}_A$ and $\mathbf{x}_B$, denoted by $\mathbf{G}(\mathbf{w}_A, \mathbf{F}_A)$ and $\mathbf{G}(\mathbf{w}_B, \mathbf{F}_B)$, respectively. The second column is generated from the feature code of $\mathbf{x}_A$ and the latent code of $\mathbf{x}_B$, denoted by $\mathbf{G}(\mathbf{w}_B, \mathbf{F}_A)$, and vice versa for the third column, denoted by $\mathbf{G}(\mathbf{w}_A, \mathbf{F}_B)$. The feature code encodes the geometric structures such as pose and facial shape, whereas the latent code controls the appearance styles like eye color and makeup.

Figure 21. **Visual results of ablation study**. We show the qualitative results of the different ablative configurations. Compared with the inversion result of our baseline, that of configuration (A) is less sharp and reconstructs less well the details. Configuration (B) fails to achieve a plausible reconstruction. The result of configuration (C) is globally plausible, yet less reliable in the details. For instance, the teeth are less photo-realistic compared with our baseline. This experiment confirms the quantitative evaluation in the main paper.

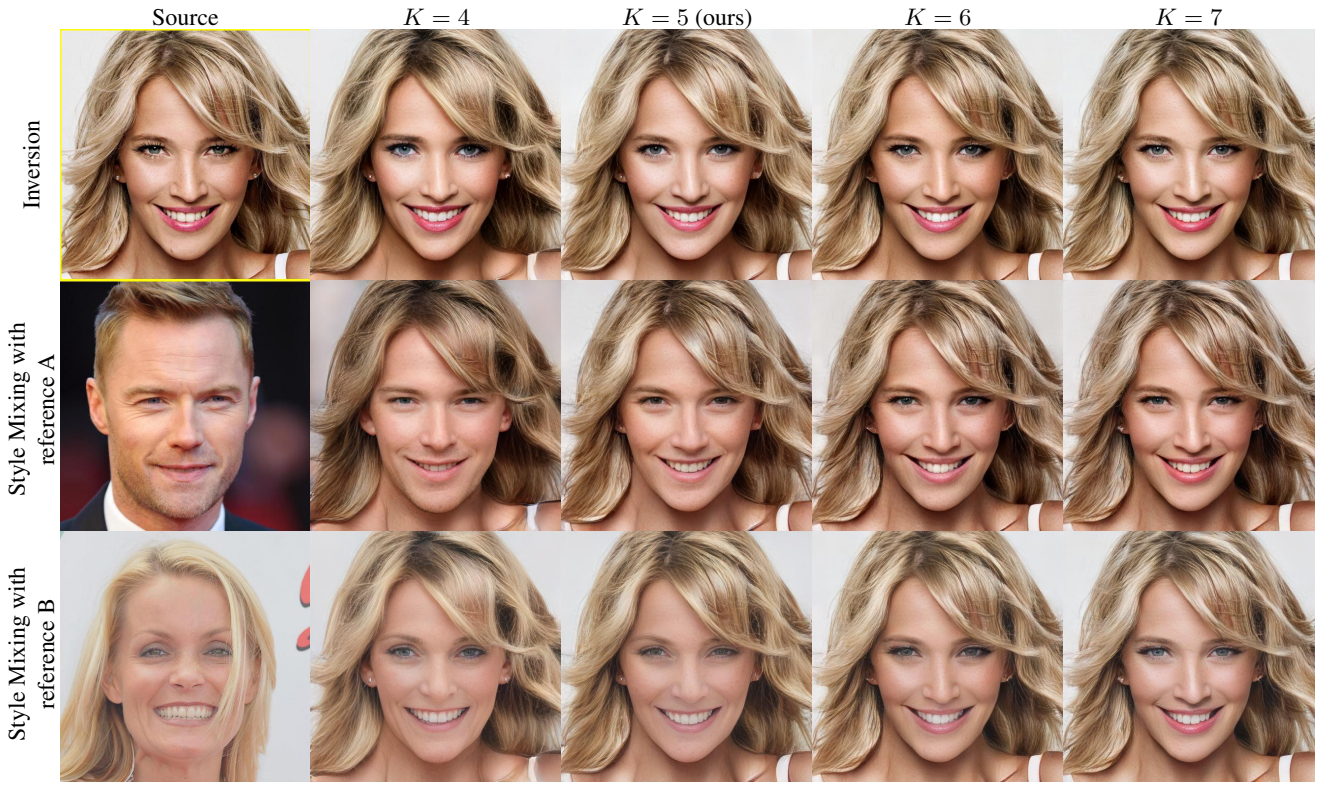


Figure 22. **Choice of feature code insertion layer K.** The first column shows the source image (yellow frame) and two reference images for style mixing. In the second to last column, the first row is the inversion results of each configuration, the second and third rows are the style mixing results, generated from the feature code of the source image and the latent code of the reference image. Choosing $K = 4$ yields good style mixing effects but lower reconstruction quality. Choosing $K = 6$ or 7 encodes nearly all the information in the feature code, which is limiting for editing. Our choice of $K = 5$ holds a balanced trade-off between editing capacity and reconstruction quality.