Gradient Adjusting Networks for Domain Inversion

Erez Sheffi, Michael Rotman, and Lior Wolf(✉)

Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel
wolf@cs.tau.ac.il

Abstract. StyleGAN2 was demonstrated to be a powerful image generation engine that supports semantic editing. However, in order to manipulate a real-world image, one first needs to be able to retrieve its corresponding latent representation in StyleGAN's latent space that is decoded to an image as close as possible to the desired image. For many real-world images, a latent representation does not exist, which necessitates the tuning of the generator network. We present a per-image optimization method that tunes a StyleGAN2 generator such that it achieves a local edit to the generator's weights, resulting in almost perfect inversion, while still allowing image editing, by keeping the rest of the mapping between an input latent representation tensor and an output image relatively intact. The method is based on a one-shot training of a set of shallow update networks (aka. Gradient Modification Modules) that modify the layers of the generator. After training the Gradient Modification Modules, a modified generator is obtained by a single application of these networks to the original parameters, and the previous editing capabilities of the generator are maintained. Our experiments show a sizable gap in performance over the current state of the art in this very active domain. Our code is available at https://github.com/sheffier/gani.

1 Introduction

The ability to distinguish between synthetic images and real ones has become increasingly challenging since the introduction of Generative Adversarial Networks (GAN) [16]. Although the images produced by this framework are often indistinguishable from real ones, one lacks the ability to control the specific outcome. Most relevant to our work is the StyleGAN [23] family of generators, which can produce, for example, realistic faces based on random input vectors. However, for most real-world face images, one cannot find an input vector that would result in exactly the same image.

This is a key limitation of StyleGANs (and other GANs), with far-reaching implications at the application level. It has been repeatedly shown that the StyleGAN latent space displays semantic properties that make it especially suited for image editing applications, see [6] for a survey. However, without the ability to embed real-world images within this space, these capabilities are limited mostly to synthetic images.

R. Gade et al. (Eds.): SCIA 2023, LNCS 13886, pp. 118–137, 2023.
https://doi.org/10.1007/978-3-031-31438-4_9

A set of techniques were, therefore, developed in order to tune the StyleGAN generator G such that it would produce the desired image [4,23,39]. This has to be done carefully, since performing this tuning too aggressively would lead to the loss of the semantic properties. While being a very active research domain, fuelled by many concrete applications, results are still lacking and the generated copy is still distinctively different from the desired image.

Existing methods either (i) optimize one image at a time, or (ii) employ pre-trained feedforward networks to produce a modification of G, given an input image I. The first category is believed to be more accurate, but slower at inference time than the second.

Our work combines elements from both approaches – we propose an optimization procedure for a single image, and our procedure employs trainable networks. The role of these networks is to estimate the change one wishes to apply to the parameters of G. By optimizing the networks to control this change rather than applying it directly, we regularize the change being applied. The per-layer networks we train adapt the parameters of G based on previous parameter variations. Thus, these changes are applied in a very local manner, separately to each layer. Through weight sharing, the capacity of these networks is limited and ensures that this mapping is relatively simple.

In addition to a novel architecture, we also modify the loss term that is used and show that a face-parsing network provides a strong signal, side by side with a face-identification network that is widely used in the relevant literature.

Our results demonstrate: (i) a far more faithful inversion in comparison to the state-of-the-art methods, (ii) a more limited effect on the result of applying G to other vectors in its input space, (iii) a superb ability to support downstream editing applications.

2 Related Work

We describe adversarial image generation methods and ways to control the generated image to fit an input image.

Generative Adversarial Networks. Generative Adversarial Networks (GAN) [16] are a family of generative models composed of two neural networks: a generator and a discriminator. The generator is tasked with learning a mapping from a latent space to a given data distribution, whereas the discriminator aims to distinguish between generated samples and real ones. GANs have been widely applied in many computer vision tasks, such as generating super-resolution images [26,49], image-to-image translation [57], and face generation [22–24].

Once trained, given an input vector, the generator produces a realistic image. Since the mapping between the input vector and the image space is not trivial, it is hard to predict the generated image from the input vector. One way of controlling the synthesis is by feeding additional information to the GAN during the training phase, for instance, by adding additional discrete [32,35], or continuous [12] labels as inputs. A major caveat in this approach is that it requires additional supervision. To bypass this limitation, other approaches constrain the input vector space directly, either by applying tools from information theory [7] or by limiting this space to a non-trivial topology [41]. As a result, the input vector space is disentangled: different entries of the input vector control a different aspect or "generative factor" of the generated image. It has been observed [23,37] that a continuous translation between two vectors in the GAN latent space leads to a continuous change in the generated image. Shen *et al.* [42] has further expanded this observation to face generation, where it was shown that facial attributes lie in different hyperplanes, and are therefore controllable using vector arithmetic in the latent space. Further investigation into the latent space structure has been performed by applying unsupervised methods, such as Principal Component Analysis (PCA) [18] or eigenvalue decomposition [43], or by using semantic labels [2,42]. The existence of an underlying structure, and the presence of a semantic algebra, make it possible to edit the generated images.

StyleGAN. StyleGAN [23] is a style-based generator architecture. Unlike the traditional GAN, which maps a noisy signal directly to images, it splits the noisy signal into two: (i) a style generating signal $z \in \mathbb{R}^{512}$ to a style latent space, $w \in W = \mathbb{R}^{512}$, which is used globally as a set of layer-wise parameters for the GAN (ii) additional noise, which is added to the feature maps after each convolutional layer. This design benefits from both stochastic variation and scale-specific feature synthesis. StyleGAN2 [24] introduced a path-length regularization term that enables smoother style mapping between Z and W, together with a better normalization scheme for the generator.

Image Editing. By understanding the influence of the different entries in the latent space, image synthesis can be controlled [7]. Specifically for face generation, this mapping has been investigated in two directions, unsupervised and supervised. The unsupervised path aims to unveil the domain's structure by applying PCA [18] or eigenvalue decomposition [43]. Other works directly influence this mapping by conditioning it on supervised labels [2,42]. The existence of such an underlying structure, and the understanding of its algebra, enables the semantic modification of generated images, for instance, for facial editing [45]. The ability to control the generated samples is a key aspect of making GANs useful for real-life applications.

Image Inversion. In order to edit a real image using a latent space modification, the originating point in the latent space has to be identified. The solutions to the inverse problem can be divided into three families: (i) optimization-based (ii) encoder-based (iii) generator-modifying. Unlike the last approach, the first two families do not alter any of the generator parameters. In the optimization-based approach, a latent code w^* is evaluated given an input image, I, in an iterative manner [10,29], such that, $I \approx G(w^*, \theta)$, where θ are the generator's parameters.

For the task of face generation, Karras *et al.* [24] proposed an optimization-based inversion scheme for StyleGAN2, where both the latent code w and the injected noise are optimized together, combined with a regularization term that minimizes the auto-correlation of the injected noise at different scales. Abdal *et al.* [1] extended this direction, by expanding latent space W to W^+. To accomplish this, instead of constraining the latent code, $w \in W$, to be identical for all convolutional layers, each layer is now fed with a different set of parameters. This modification extended the dimensionality of the latent code from $\mathbb{R}^{512}$ to $\mathbb{R}^{18\times512}$. To preserve spatial details during inversion, Zhang*et al.* [54] considered a spatially-structured latent space, replacing the original one-dimensional representation, W.

The second family of solutions is encoder-based. These methods utilize an encoder, E, that maps between the image space and the latent space, $w^* = E\left(I\right)$. Unlike the iterative family, these encoders are trained on a set of samples [17, 30]. These image encoders have also been applied for face generation, where they are employed during training, by combining the auto-encoder framework with a GAN [36], or more commonly, on pre-trained generative models, which require far less training data. These approaches focus on mapping an image to an initial latent code, $w \in W$, which may later be fine-tuned using an optimization method [56]. pSp [38] employs a different scheme, in which an additional fine-tuning of w is not performed; instead, a pyramid-based encoder is designed for each style vector of the StyleGAN2 framework. e4e [46] further expands on this idea by limiting the hierarchical structure of the w codes to be of a residual type, so that each style vector is the sum of a basis style vector and a residual part. Further improvement was achieved for the inversion problem by iteratively encoding the image onto the latent space and feeding the generated image back to the encoder as an input [3].

In the last approach, an initial w latent code is evaluated using an encoder. As a second step, the generator is tuned to produce the required image from the w [39]. This tuning procedure has been further improved by employing hypernetworks [4]. In this scenario, instead of directly modifying the generator weights, a set of residual weights is evaluated using an additional neural network. The input to this network is the target image. Therefore, given a new image, a new set of weights is computed.

A faithful edit is also required to preserve the original identity. Liang *et al.* [28] applied Neural Spline Networks to find faithful editing directions. Editing local aspects was accomplished by manipulating specific parts of the feature maps throughout the generation process [48].

3 Method

Our approach adapts a StyleGAN generator for one image at a time by adding a small correction to the generator's parameters. This correction is computed using the novel gradient modification modules, a set of small neural networks that map between the gradients of the loss criteria with respect to the generator's parameters to the parameters' corrections. During training, only the parameters of the gradient modification modules are optimized (the generator parameters are kept frozen) with respect to the loss criteria. The output of these modules is used to update the parameters of StyleGAN's generator, resulting in a new generator that is capable of faithfully generating the input image. Note that the gradient modification modules are not added to StyleGAN and that the structure of StyleGAN is not modified at all.

Given a candidate target image I to be edited, a corresponding latent code, $w \in W^+ = \mathbb{R}^{18\times512}$, is estimated using the off-the-shelf encoder e4e [46]. The latent code, w, is fed into the pre-trained generator to produce a reconstructed image, $G(w) = G(w, \theta)$, where the right-hand side explicitly states the parameters θ of network G. Since w is not an exact solution to the inverse problem, the reconstructed image is usually of poor quality. In our method, w does not change. Instead, we tune the generator parameters θ to improve the generated image, obtaining $G(w, \theta') \sim I$, where θ' are the tuned parameters.

Consider the following image similarity loss function,

$$\mathcal{L}\left(I_1, I_2\right) = \lambda_1 \mathcal{L}_{\text{rec}} + \lambda_2 \mathcal{L}_{lpips} + \lambda_3 \mathcal{L}_{sim} + \lambda_4 \mathcal{L}_{FP}\,, \tag{1}$$

which is the sum of four terms. The first term is a pixel-wise reconstruction loss, the L_2 or the L_1^{smooth} [15] distance between the images I_1 and I_2. The second term, $\mathcal{L}_{lpips}$, is a perceptual similarity loss [55], that relies on feature maps from a pre-trained AlexNet [44] on the ImageNet dataset. $\mathcal{L}_{sim}$ is an identity-preserving similarity loss that is applied on a pair of real and reconstructed images. This loss term accounts for the cosine distance of feature vectors extracted from a pre-trained ArcFace [11] facial recognition network for the facial domain, and a pre-trained MoCo [8] for the non-facial domains, following [4,46]. The last term is a multi-layer face-parsing loss, $\mathcal{L}_{FP}$. Similarly to $\mathcal{L}_{sim}$, it measures the layer-wise aggregated cosine distance of all the feature vectors from the contracting path of the pre-trained facial parsing network P [27], with a U-Net [40] backbone. In total, 5 feature vectors are used for the cosine distance evaluation.

Tuning the G's parameters, θ, involves the estimation of the modified parameters, θ', using a set of feed-forward networks. Let l_i be the ith layer of G, and let θ_i and $\frac{\partial\mathcal{L}}{\partial\theta_i}$ be its learnable parameters and the gradients of the objective function w.r.t to these parameters, respectively. Unlike PTI [39], which applies a regular gradient step to update θ_i, the gradient updates are based on the mapping,

$$\theta_i' = \theta_i \odot (1 + \Delta\theta_i) = \theta_i \odot \left(1 + M_i\left(\frac{\partial\mathcal{L}}{\partial\theta_i}\right)\right)\,, \tag{2}$$

where $\mathcal{M} = \{M_i\}$ is a set of gradient modification modules, each mapping between the original gradients, $\frac{\partial\mathcal{L}}{\partial\theta_i}$, and the parameter correction, $\Delta\theta_i$.

Each module M_i contains a sequential set of $l = 1 \ldots L$ residual blocks [20]. Let y_l be the input to the lth residual block. The output of block l, y_{l+1} is then:

$$r = W_2^l \sigma\left(\text{SN}_2^l\left(W_1^l \sigma\left(\text{SN}_1^l\left(y_l\right)\right)\right)\right) + b^l \tag{3}$$

$$y_{l+1} = y_l + r\,, \tag{4}$$

where σ is the LeakyReLU [31] activation function, with a slope of 0.01 and SN is the Scale Normalization (SN) [34]. The learnable parameters of each block

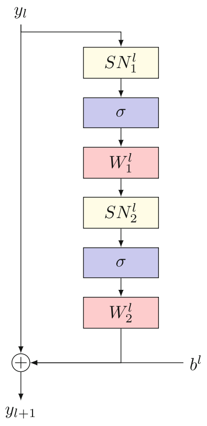

Fig. 1. The structure of one residual block of a gradient modification module, $\mathcal{M}_i$. The input to the first block, y_0 is, $\frac{\partial \mathcal{L}}{\partial \theta_i}$.

reside in two linear operators, W_1^l and W_2^l, in the bias term, b^l, and in the scale coefficient of the SN layer. The parameters of W_1^l and W_2^l are initialized according to [19] from the normal distribution, with an initial standard deviation factored by 0.1, whereas the bias, b^l is sampled uniformly. All the M_i networks assigned to convolutional layers with the number of channels, $c_{in} = c_{out} = 512$, share the same parameters, $\psi_i = \left\{W_1^l, W_2^l, b^l, \text{SN}_1^l, \text{SN}_2^l\right\}_{l=1}^{L}$. During optimization, (only) the parameters $\{\psi_i\}$ of the set $\mathcal{M}$ are optimized to minimize the loss function,

$$\mathcal{L}_{total} = \lambda_e \mathcal{L}\left(I, G(w, \theta')\right) + \lambda_l \mathcal{L}\left(G(f(z), \theta), G(f(z), \theta')\right) \tag{5}$$

where $z \sim \mathcal{N}(0, \mathbb{1}) \in \mathbb{R}^{512}$, and f is the style mapping of StyleGAN [23] so style mapping, $f(z)$ lies in W (and not in W^+). The first term in Eq. (5) is responsible for generating a high-quality image. It is crucial that the optimization procedure should not dramatically alter the mapping $G : w \to I$ as a large variation in the mapping would require the re-identification of the editing directions. The second term in Eq. (5), a localization regularizer [33,39], prevents the generator from drifting by forcing the generator to produce identical images for randomly sampled latent codes. An outline of our method appears in Algorithm 1 (Fig. 1).

4 Experiments

We conduct an extensive set of experiments on various image synthesis datasets. For all image domains, our method utilizes a pre-trained StyleGAN2. For the facial domain, this pre-trained network was optimized for generating facial images distributed according to the FFHQ [23] dataset, whereas the inversion and editing capabilities of our approach are evaluated over images from the test set of the Celeb-HQ [22] dataset. Our method is also evaluated on Church and

Algorithm 1 Method Outline

```
Input: Image $I$, generator $G_\theta$, encoder $E$, $\mathcal{M}$
Output: $G(w, \theta')$
  $w \leftarrow E(I)$                                    ▷ Obtain $w$ from a pre-trained encoder
  for $i \in 1 \ldots n_{\text{iterations}}$ do
      Compute $\frac{\partial \mathcal{L}(I, G(w,\theta))}{\partial \theta}$
      for $M_i \in \mathcal{M}$ do
          $\Delta\theta_i = M_i\left(\frac{\partial \mathcal{L}}{\partial \theta_i}\right)$          ▷ $\mathcal{L}$ in Eq. (1)
      end for
      Sample $z \sim \mathcal{N}(0, \mathbb{1}) \in \mathbb{R}^{512}$            ▷ Localization
      Compute $\frac{\partial \mathcal{L}_{total}}{\partial \psi}$                      ▷ $\mathcal{L}_{total}$ in Eq. (5)
      Update the parameters $\psi$ of $\mathcal{M}$
  end for
```

Table 1. The loss coefficients and number of running iterations that were used by our method for each of the different datasets.

	Iterations	λ_e	λ_l	λ_1	λ_2	λ_3	λ_4
CelebA-HQ	300	1.0	0.2	1.0	0.8	0.1	1.0
AFHQ-Wild	300	1.0	0.2	1.0	0.8	0.1	1.0
Stanford Cars	400	1.0	0.2	1.0	0.8	0.1	–
LSUN Horses	800	1.0	0.2	1.0	0.8	0.1	–
LSUN Church	800	1.0	0.2	1.0	0.8	0.1	–

Horses from the LSUN [53] dataset, on automobiles from Stanford Cars [25], and on wildlife images from AFHQ-WILD [9]. The pre-trained models for these domains were acquired from the e4e [46] and HyperStyle [4] official GitHub repositories. The initial inverted vectors, $w \in W^+$, were also obtained using the e4e encoders in these repositories. Since there is no pre-trained encoder to W^+ space for the AFHQ-WILD dataset, our method is evaluated on this using the W latent space.

For all experiments, the Ranger [51] optimizer was used, with a learning rate of 0.001. The number of iterations and the loss coefficients used for our method appears in Table 1. For the CelebA-HQ and AFHQ-Wild datasets, $\mathcal{L}_{\text{pixel-wise}} = L_2$ and for Stanford Cars, LSUN Church, and LSUN Horses, $\mathcal{L}_{\text{pixel-wise}} = L_1^{\text{smooth}}$ with $\beta = 0.1$. The difference in the number of iterations depends on the quality of the reconstruction of the original generator G from w. Both LSUN Church and LSUN Horses resulted in a poor initial reconstruction and required twice as many iterations.

Table 2 presents four evaluation criteria for reconstruction quality over the CelebA-HQ dataset. The four metrics compare the input image I with the one generated by the inversion method, $G(w, \theta')$. These metrics are the pixel-wise similarity, which is the Euclidean norm L_2, the perceptual similarity, LPIPS [55], the structural similarity score, MS-SSIM [50], and identity similarity, ID [21], between the input image, I, and its reconstruction, $G(w, \theta')$. Our method out-

Table 2. Reconstruction metrics on the CelebA-HQ test set.

Method	↑ID	↑MS-SSIM	↓LPIPS	↓ L_2
StyleGAN2 [24]	0.78	0.90	0.020	0.090
PTI [39]	0.85	0.92	0.090	0.015
IDInvert [56]	0.18	0.68	0.220	0.061
pSp [38]	0.56	0.76	0.170	0.034
e4e [46]	0.50	0.72	0.200	0.052
ReStyle$_{\text{pSp}}$ [3]	0.66	0.79	0.130	0.030
ReStyle$_{\text{e4e}}$ [3]	0.52	0.74	0.190	0.041
NP-GAN-I [14]	–	–	0.283	0.004
HyperStyle [4]	0.76	0.84	0.090	0.019
HFGI [5]	–	–	0.100	0.021
HyperInverter [13]	0.60	0.67	0.105	0.024
Feature-Style Encoder [52]	0.87	–	0.066	0.019
Ours	**0.99**	**0.97**	**0.020**	**0.003**

Table 3. Reconstruction metrics on the Stanford Cars dataset [25] test set.

Method	↑MS-SSIM	↓LPIPS	↓ L_2
StyleGAN2 [24]	0.79	0.16	0.060
PTI [39]	0.93	0.11	0.010
pSp [38]	0.58	0.29	0.100
e4e [46]	0.53	0.32	0.120
ReStyle$_{\text{pSp}}$ [3]	0.66	0.25	0.070
ReStyle$_{\text{e4e}}$ [3]	0.60	0.29	0.090
HyperStyle [4]	0.67	0.27	0.070
NP-GAN-I [14]	–	0.15	**0.006**
Ours	**0.94**	**0.03**	0.010

performs all other methods in all metrics except LPIPS, where the original StyleGAN2 inversion approach matches ours.

Table 3 evaluates the reconstruction quality for the Stanford Cars dataset, Table 4 for the AFHQ-Wild dataset, Table 5 for the LSUN church dataset and Table 6 for the LSUN horses dataset, using three evaluation criteria: the pixel-wise Euclidean norm, L_2, the perceptual similarity, LPIPS, and the MS-SSIM [50] score. Our method outperforms all other methods on the AFHQ-Wild and LSUN church datasets. In the Stanford Cars and the LSUN horses datasets, our approach surpasses other approaches in all evaluation metrics, except for the L_2 metric, where it is second to Near Perfect GAN Inversion (NP-GAN-I) [14].

Inversion Quality. We begin with a qualitative evaluation of reconstructed images. Figure 2 demonstrates the reconstruction of facial images taken from the CelebA-HQ [22] dataset, and Fig. 3 demonstrates the reconstruction of car images taken from the Stanford Cars [25] dataset. We compare the reconstructed images produced by our method with the following previous approaches: pSp [38], e4e [46], ReStyle$_{\text{pSp}}$ [3], ReStyle$_{\text{e4e}}$ [3], HyperStyle [4].

As can be seen in Fig. 2, our method is able to generate almost identical reconstructions. The first and third rows demonstrate the reconstruction of difficult examples. Our method is able to produce near-identical reconstruction, whereas all other methods struggle to achieve inversion of good quality. The second row demonstrates the reconstruction of a relatively easy example. Although all methods are able to produce meaningful reconstruction, only our method is truly able to preserve identity and properly reconstruct fine details (such as gaze, eye color, dimples, etc.).

As can be seen in Fig. 3, there is a large gap between our method and the other ones. Specifically, all baseline methods struggle to reconstruct various elements (fine or coarse). In all three rows, only our method is able to reconstruct the coarse shape of the car properly. In the first row, we can see that only our method is able to reconstruct the following elements properly: (i) the coarse shape of the car, (ii) the shape of the headlight, (iii) the fact that the lights are on, (iv) the absence of a roof, (v) the car's logo. In the second row, only our method is able to reconstruct the following elements properly: (i) shape of the

Table 4. Reconstruction metrics for AFHQ-Wild test set.

Method	↑MS-SSIM	↓LPIPS	↓ L_2
StyleGAN2 [24]	0.82	0.13	0.030
PTI [39]	0.93	0.08	0.010
pSp [38]	0.51	0.35	0.130
e4e [46]	0.47	0.36	0.140
ReStyle$_{\text{pSp}}$ [3]	0.57	0.21	0.050
ReStyle$_{\text{e4e}}$ [3]	0.52	0.25	0.070
HyperStyle [4]	0.56	0.24	0.060
NP-GAN-I [14]	–	0.38	0.014
Ours	**0.96**	**0.03**	**0.006**

Table 5. Reconstruction metrics on the LSUN churches [53] (outdoor) test set.

Method	↑MS-SSIM	↓LPIPS	↓ L_2
StyleGAN2 [24]	0.4797	0.325	0.167
PTI [39]	0.6968	0.097	0.051
pSp [38]	0.4070	0.310	0.130
e4e [24]	0.3481	0.420	0.140
ReStyle [3]	0.3878	0.377	0.127
HFGI [5]	–	0.220	0.090
HyperInverter [52]	0.5762	0.223	0.091
Ours	**0.9479**	**0.015**	**0.014**

headlights, (ii) side-mirrors, (iii) color tone, (iv) reflection. In the third row, only our method is able to reconstruct the following elements properly: (i) wheels, (ii) placement and size of the passenger window, (iii) side-mirror shape and color (Fig. 4).

To test the effect of the number of residual blocks, L, of the gradient modification modules, the image reconstruction performance of our approach was evaluated for 200 randomly sampled images from the CelebA-HQ test set. As Table 7 shows, adding more blocks slightly improves the image quality scores of the MS-SSIM, LPIPS, and L_2 metrics, whereas it hardly degrades the identity preservation score.

Table 6. Reconstruction metrics for LSUN horses.

Method	↑MS-SSIM	↓LPIPS	↓L_2
e4e [24]	0.25	0.431	0.1740
ReStyle [3]	–	0.525	0.159
NP-GAN-I [14]	–	0.141	**0.005**
Ours	**0.94**	**0.016**	0.010

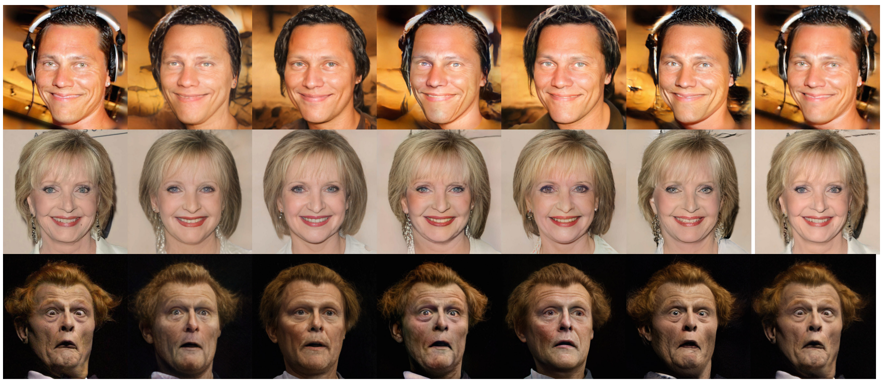

Fig. 2. Visual comparison of image reconstruction quality on CelebA-HQ.

In order to assess the contribution of the normalization scheme utilized by our approach our model was evaluated on the first 200 samples from the test set of CelebA-HQ. First, we test whether the inputs to the module M_i have to be normalized using SN, Instance Normalization (IN) [47], or kept untouched. Additionally, we test which normalization scheme should be applied inside each residual block. Table 8 summarizes the different normalization choices. As can be seen, not applying normalization in the residual block harms all evaluation metrics, whereas normalizing the inputs prior to feeding them into the gradient modification modules, M_i, is not necessary.

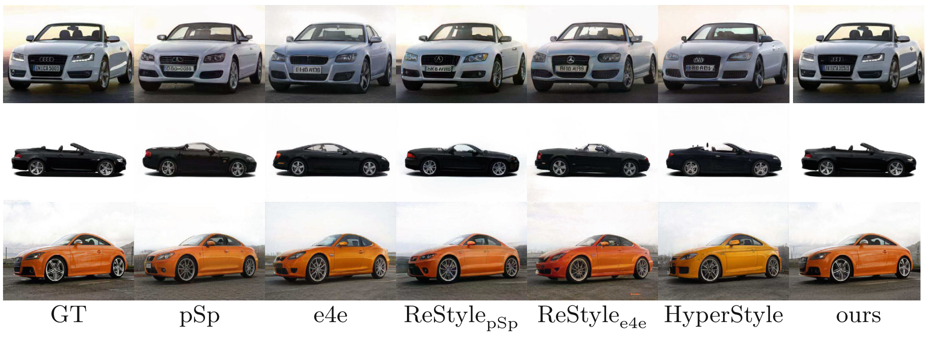

Fig. 3. Visual comparison of image reconstruction quality on the Stanford Cars.

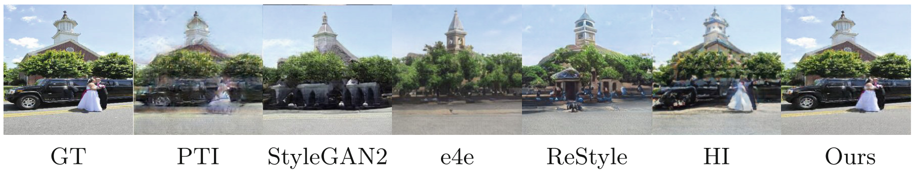

Fig. 4. Visual comparison of image reconstruction quality on the LSUN Church. HI stands for HyperInverter [52]

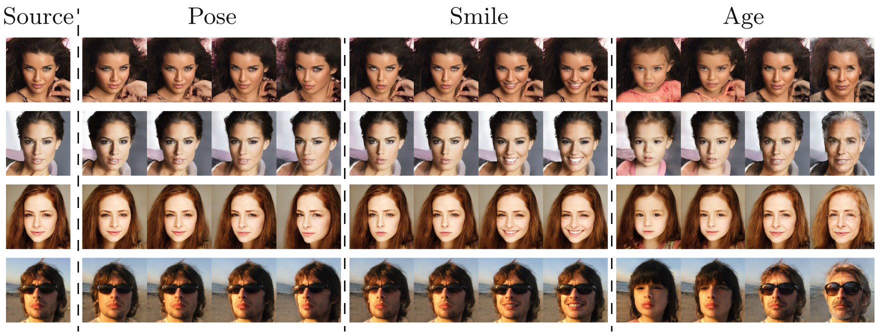

Fig. 5. Pose, smile and age editing using InterFaceGAN [42] on CelebA-HQ.

Editing Quality. The ability of our method to retain the original editing directions of StyleGAN is seen in Fig. 5 for the CelebA-HQ dataset, editing directions and tools were taken from InterFaceGAN [42]. A visual comparison to other approaches for Stanford Cars and LSUN Church datasets appears in Fig. 7 and Fig. 6, respectively. For these datasets, the editing directions were taken from GANSpace [18]. Evidently, unlike other approaches, our method is able to retain the editing direction without harming image quality, for instance, on the automotive images, it is the only approach that modifies the color uniformly, whereas in the outdoor image in Fig. 6, it does not introduce artifacts or change the image's content.

Face-Parsing Loss. Figure 8 shows the benefits of applying the face-parsing loss to images from the CelebA-HQ and AFHQ-Wild datasets. As can be seen, for the facial image, the reconstructed image retains the fine details in the lips when the parsing loss is applied. Furthermore, even though the parsing loss utilizes a pre-trained network, trained on the task of human facial semantic segmentation, it also improves the reconstruction quality for other non-human facial domains.

Localization Regularization. The localization term prevents model drifting, which is often manifested as additional artifacts. As seen in Fig. 9, it is helpful in preventing the diffusion of the skin color to the teeth.

Table 7. The effect of the number of residual blocks, L, on image reconstruction. These results are obtained on a fixed evaluation set consisting of 200 randomly sampled images from the CelebA-HQ test set.

L	# Parameters	↑ID	↑MS-SSIM	↓LPIPS	↓L_2
1	1, 396, 658	**0.99773**	0.9739	0.0201	0.0033
2	2, 793, 316	0.99769	0.9778	0.0177	0.0028
3	4, 189, 974	0.99751	0.9785	0.0170	0.0027
4	5, 586, 632	0.99729	**0.9790**	**0.0167**	**0.0026**

Table 8. Different normalization schemes applied to 200 samples from the test set of the CelebA-HQ dataset. PreNorm stands for normalizing the inputs prior to the application of M_i, and PostNorm stands for the normalization scheme applied in each residual block. SN is the Scale Normalization [34] and IN is the one-dimensional Instance Normalization [47].

	PreNorm	PostNorm	↑ID	↑MS-SSIM	↓LPIPS	↓ L_2
(a)	–	–	0.834	0.765	0.172	0.039
(b)	–	IN	0.996	0.996	0.030	0.005
(c)	–	SN	**0.998**	**0.998**	**0.020**	**0.003**
(d)	IN	–	0.895	0.802	0.151	0.032
(e)	IN	IN	0.997	0.974	0.021	0.003
(f)	IN	SN	0.998	0.973	0.021	0.003
(g)	SN	–	0.997	0.778	0.168	0.037
(h)	SN	IN	0.857	0.972	0.023	0.004
(i)	SN	SN	0.997	0.974	0.021	0.003

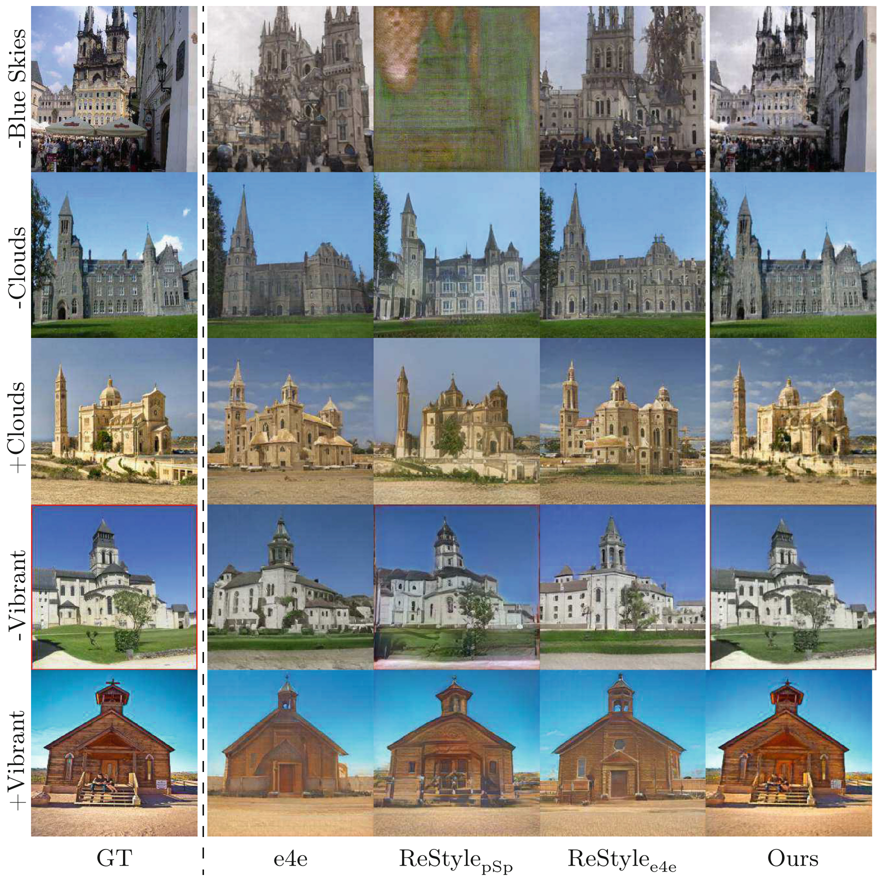

Fig. 6. Editing quality on LSUN Church using GANSpace [18]

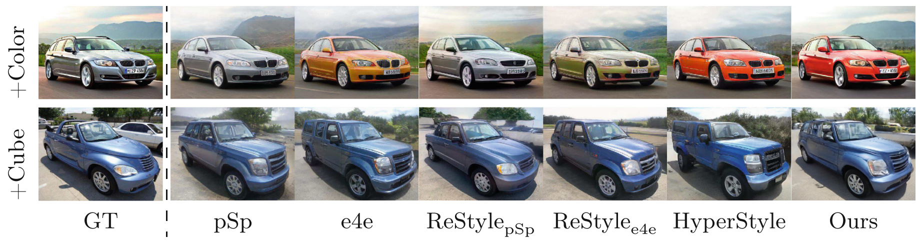

Fig. 7. Editing quality on the Stanford Cars using GANSpace [18]

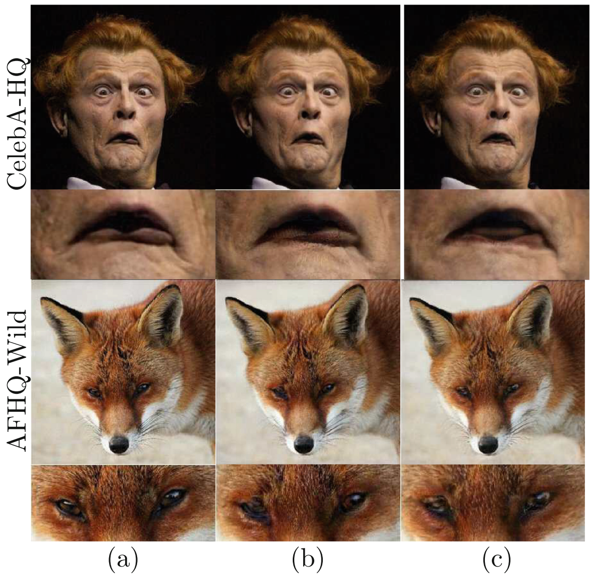

Fig. 8. Visual comparison of the effect of training with $\mathcal{L}_{FP}$. (a) Ground truth (b) with $\mathcal{L}_{FP}$ (c) without $\mathcal{L}_{FP}$.

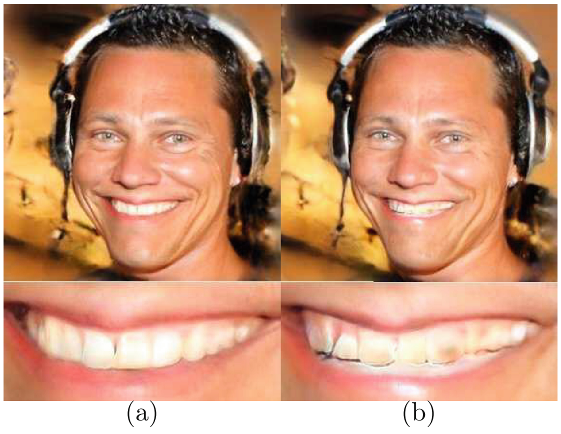

Fig. 9. Image reconstruction. (a) with localization. (b) without localization

5 Conclusions

The proliferation of work devoted to effectively inverting StyleGANs indicates an acute need for such technologies, in order to perform image editing and semantic image manipulations. In this work, we present a novel method that employs learned mappings between the loss gradient of a layer and a suggested shift to the layer's parameters. These shifts are used within an iterative optimization process, in order to fine-tune the StyleGAN generator. The learning of the mapping networks and the optimization of the generator occur concurrently and on a single sample.

We conduct a set of experiments that is considerably more extensive than what has been presented in recent relevant contributions, showing that the modified generator produces the target image more accurately than all other methods in this very active field. Moreover, it supports downstream editing tasks more convincingly than the alternatives.

Acknowledgements. This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research, innovation program (grant ERC CoG 725974).

A Additional Qualitative Results

(See Figs. 10, 11, 12 and 13).

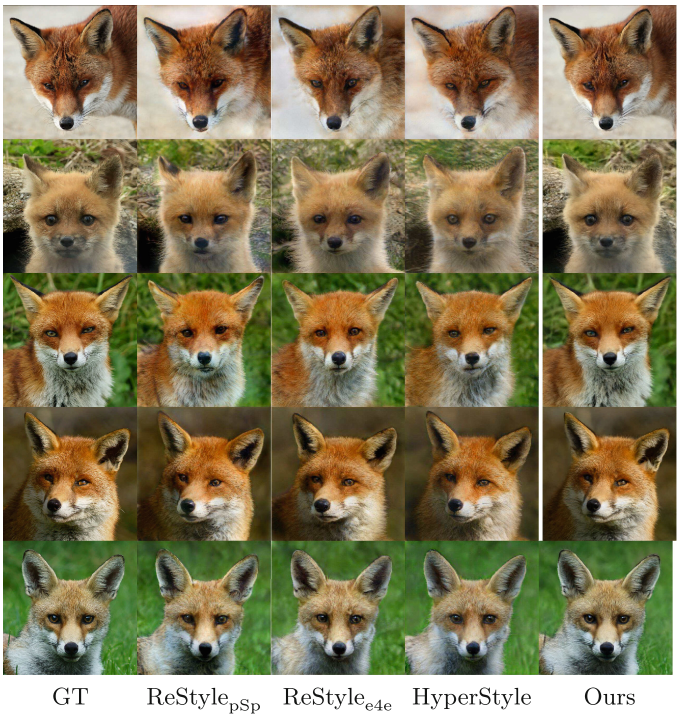

Fig. 10. Visual comparison of image reconstruction quality on the AFHQ-Wild dataset.

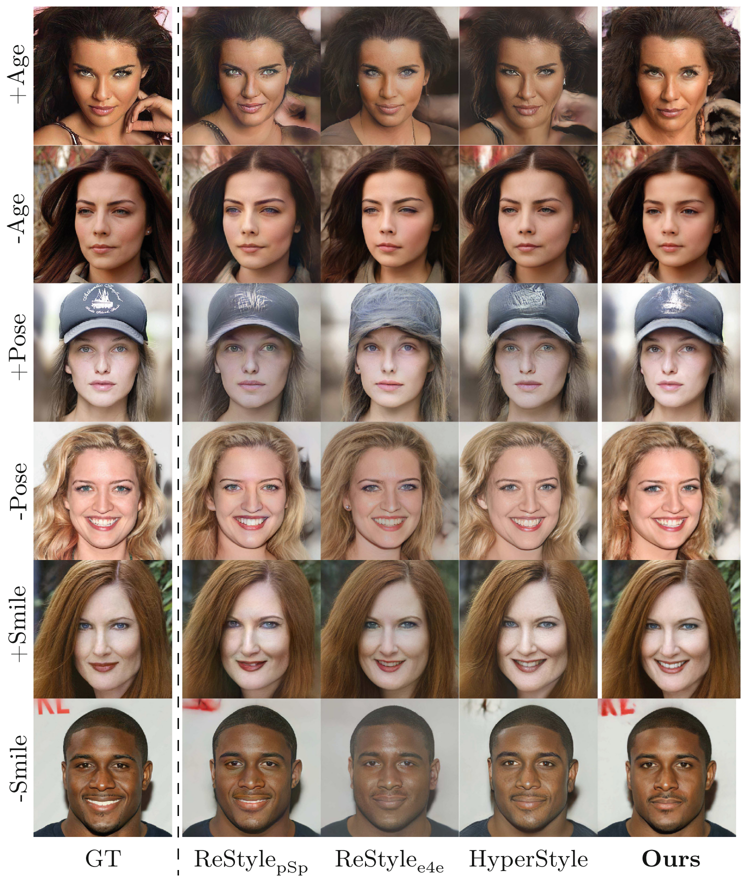

Fig. 11. Additional Pose, smile and age editing using InterFaceGAN [42] of images from the CelebA-HQ dataset.

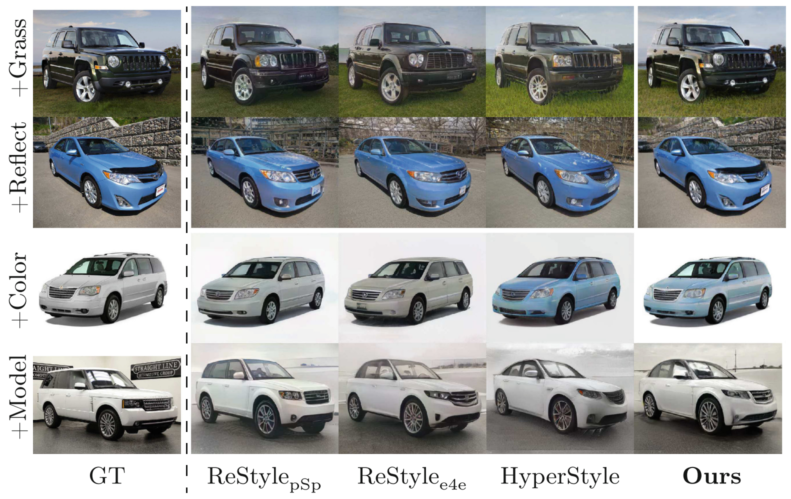

Fig. 12. Additional visual comparison of editing quality (Stanford Cars). Edits were obtained using GANSpace [18]

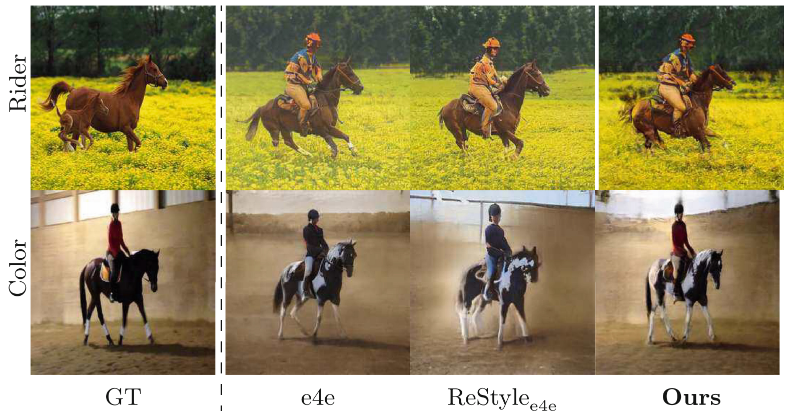

Fig. 13. Visual comparison of editing quality (LSUN Horses). Edits were obtained using GANSpace [18]

References

1. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: how to embed images into the stylegan latent space? In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 4431–4440 (2019)
2. Abdal, R., Zhu, P., Mitra, N.J., Wonka, P.: Styleflow: attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. ACM Trans. Graph. (TOG) **40**(3), 1–21 (2021)

3. Alaluf, Y., Patashnik, O., Cohen-Or, D., et al.: Restyle: a residual-based stylegan encoder via iterative refinement. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021)
4. Alaluf, Y., Tov, O., Mokady, R., Gal, R., Bermano, A.H.: Hyperstyle: stylegan inversion with hypernetworks for real image editing. arXiv preprint arXiv:2111.15666 (2021)
5. Bai, Q., Xu, Y., Zhu, J., Xia, W., Yang, Y., Shen, Y.: High-fidelity gan inversion with padding space. ArXiv abs/2203.11105 (2022)
6. Bermano, A.H., et al.: State-of-the-art in the architecture, methods and applications of stylegan. arXiv preprint arXiv:2202.14020 (2022)
7. Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: Infogan: interpretable representation learning by information maximizing generative adversarial nets. Adv. Neural Inf. Process. Syst. **29** (2016)
8. Chen, X., Fan, H., Girshick, R.B., He, K.: Improved baselines with momentum contrastive learning. ArXiv abs/2003.04297 (2020)
9. Choi, Y., Uh, Y., Yoo, J., Ha, J.W.: Stargan v2: diverse image synthesis for multiple domains. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
10. Creswell, A., Bharath, A.A.: Inverting the generator of a generative adversarial network. IEEE Trans. Neural Netw. Learn. Syst. **30**(7), 1967–1974 (2018)
11. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2019)
12. Ding, X., Wang, Y., Xu, Z., Welch, W.J., Wang, Z.J.: Ccgan: continuous conditional generative adversarial networks for image generation. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=PrzjugOsDeE
13. Dinh, T.M., Tran, A., Nguyen, R.H.M., Hua, B.S.: Hyperinverter: improving stylegan inversion via hypernetwork. ArXiv abs/2112.00719 (2021)
14. Feng, Q., Shah, V., Gadde, R., Perona, P., Martinez, A.: Near perfect gan inversion. ArXiv abs/2202.11833 (2022)
15. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
16. Goodfellow, I., et al.: Generative adversarial nets. Adv. Neural Inf. Process. Syst. **27** (2014)
17. Guan, S., Tai, Y., Ni, B., Zhu, F., Huang, F., Yang, X.: Collaborative learning for faster stylegan embedding. arXiv preprint arXiv:2007.01758 (2020)
18. Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: Ganspace: discovering interpretable gan controls. Adv. Neural Inf. Process. Syst. **33**, 9841–9850 (2020)
19. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
20. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 630–645. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_38
21. Huang, Y., et al.: Curricularface: adaptive curriculum learning loss for deep face recognition. In: proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5901–5910 (2020)
22. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)

23. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4401–4410 (2019)
24. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8110–8119 (2020)
25. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3D object representations for fine-grained categorization. In: 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13), Sydney, Australia (2013)
26. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4681–4690 (2017)
27. Lee, C.H., Liu, Z., Wu, L., Luo, P.: Maskgan: towards diverse and interactive facial image manipulation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
28. Liang, H., Hou, X., Shen, L.: Ssflow: style-guided neural spline flows for face image manipulation. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 79–87 (2021)
29. Ling, H., Kreis, K., Li, D., Kim, S.W., Torralba, A., Fidler, S.: Editgan: high-precision semantic image editing. In: Advances in Neural Information Processing Systems (NeurIPS) (2021)
30. Luo, J., Xu, Y., Tang, C., Lv, J.: Learning inverse mapping by autoencoder based generative adversarial nets. In: International Conference on Neural Information Processing, pp. 207–216. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-319-70096-0_22
31. Maas, A.L., Hannun, A.Y., Ng, A.Y., et al.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of ICML, p. 3. Citeseer (2013)
32. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
33. Mitchell, E., Lin, C., Bosselut, A., Finn, C., Manning, C.D.: Fast model editing at scale. CoRR (2021). https://arxiv.org/pdf/2110.11309.pdf
34. Nguyen, T.Q., Salazar, J.: Transformers without tears: improving the normalization of self-attention. arXiv preprint arXiv:1910.05895 (2019)
35. Odena, A., Olah, C., Shlens, J.: Conditional image synthesis with auxiliary classifier gans. In: International Conference on Machine Learning, pp. 2642–2651. PMLR (2017)
36. Pidhorskyi, S., Adjeroh, D.A., Doretto, G.: Adversarial latent autoencoders. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14104–14113 (2020)
37. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
38. Richardson, E., et al.: Encoding in style: a stylegan encoder for image-to-image translation. arXiv preprint arXiv:2008.00951 (2020)
39. Roich, D., Mokady, R., Bermano, A.H., Cohen-Or, D.: Pivotal tuning for latent-based editing of real images. arXiv preprint arXiv:2106.05744 (2021)
40. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28

41. Rotman, M., Dekel, A., Gur, S., Oz, Y., Wolf, L.: Unsupervised disentanglement with tensor product representations on the torus. In: International Conference on Learning Representations (2022)
42. Shen, Y., Gu, J., Tang, X., Zhou, B.: Interpreting the latent space of gans for semantic face editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9243–9252 (2020)
43. Shen, Y., Zhou, B.: Closed-form factorization of latent semantics in gans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1532–1540 (2021)
44. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
45. Tewari, A., et al.: Stylerig: Rigging stylegan for 3D control over portrait images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6142–6151 (2020)
46. Tov, O., Alaluf, Y., Nitzan, Y., Patashnik, O., Cohen-Or, D.: Designing an encoder for stylegan image manipulation. ACM Trans. Graph. (TOG) **40**(4), 1–14 (2021)
47. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: the missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
48. Wang, R., et al.: Attribute-specific control units in stylegan for fine-grained image manipulation. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 926–934 (2021)
49. Wang, X., et al.: Esrgan: enhanced super-resolution generative adversarial networks. In: Proceedings of the European conference on computer vision (ECCV) workshops (2018)
50. Wang, Z., Simoncelli, E.P., Bovik, A.C.: Multiscale structural similarity for image quality assessment. In: The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, vol. 2, pp. 1398–1402. IEEE (2003)
51. Wright, L.: Ranger - a synergistic optimizer (2019). http://github.com/lessw2020/Ranger-Deep-Learning-Optimizer
52. Yao, X., Newson, A., Gousseau, Y., Hellier, P.: Feature-style encoder for style-based gan inversion. ArXiv abs/2202.02183 (2022)
53. Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)
54. Zhang, L., Bai, X., Gao, Y.: Sals-gan: spatially-adaptive latent space in stylegan for real image embedding. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 5176–5184 (2021)
55. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 586–595 (2018)
56. Zhu, J., Shen, Y., Zhao, D., Zhou, B.: In-domain GAN inversion for real image editing. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12362, pp. 592–608. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58520-4_35
57. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2223–2232 (2017)