StyleFlow: Attribute-conditioned Exploration of StyleGAN-Generated Images using Conditional Continuous Normalizing Flows

RAMEEN ABDAL and PEIHAO ZHU, KAUST
NILOY J. MITRA, University College London and Adobe Research
PETER WONKA, KAUST

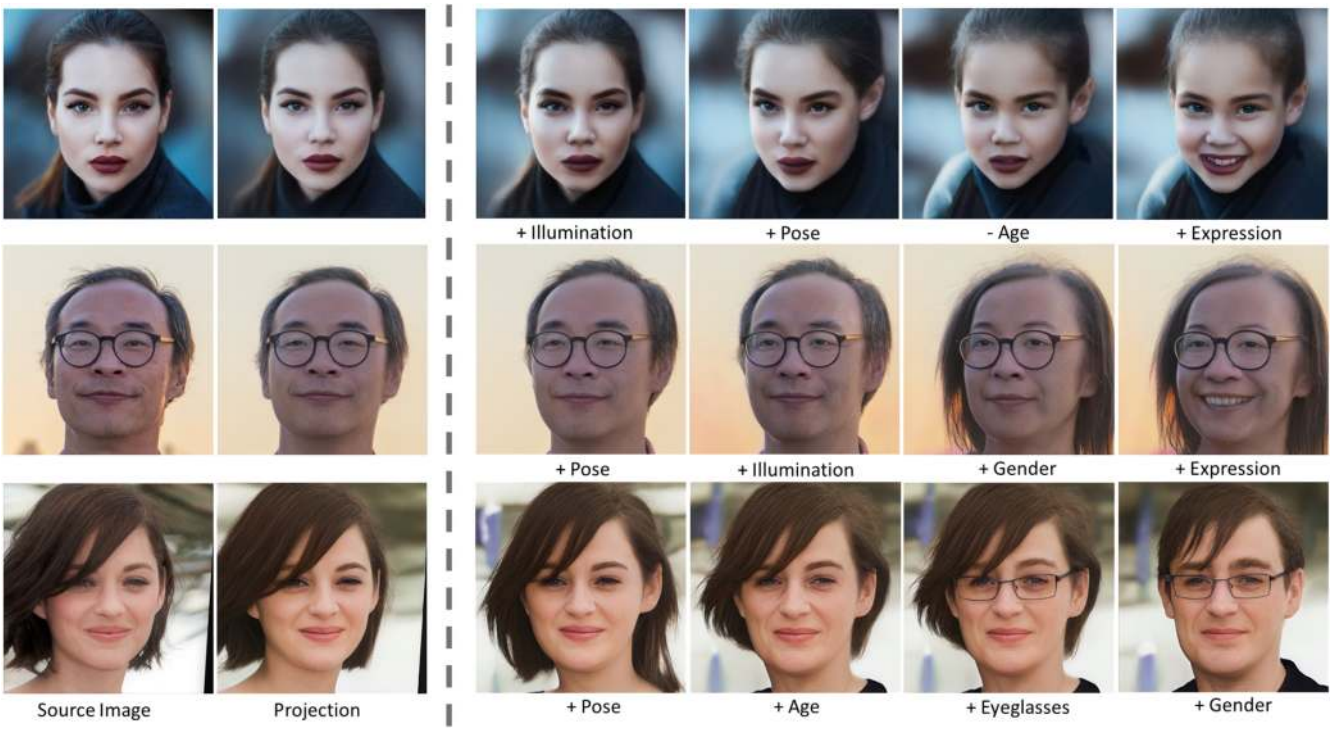

Fig. 1. We present *StyleFlow* to enable attribute-conditioned semantic edits on projected real images and StyleGAN generated images. For each of these examples, the user sequentially changes (camera) pose, illumination, expression, eyeglasses, gender, and age of a real image. Please judge, where applicable, the extent of identity preservation of the respective person under the applied edits. In this figure, all the source images are real images.

High-quality, diverse, and photorealistic images can now be generated by unconditional GANs (e.g., StyleGAN). However, limited options exist to control the generation process using (semantic) attributes while still preserving the quality of the output. Further, due to the entangled nature of the GAN latent space, performing edits along one attribute can easily result in unwanted changes along other attributes. In this article, in the context of *conditional exploration* of entangled latent spaces, we investigate the two sub-problems of attribute-conditioned sampling and attribute-controlled editing. We present StyleFlow as a simple, effective, and robust solution to both the sub-problems by formulating conditional exploration as an instance of conditional continuous normalizing flows in the GAN latent space conditioned by attribute features. We evaluate our method using the face and the car latent space of StyleGAN, and demonstrate fine-grained disentangled edits along various attributes on both real photographs and StyleGAN generated images. For example, for faces, we vary camera pose, illumination variation, expression, facial hair, gender, and age. Finally, via extensive qualitative and quantitative comparisons, we demonstrate the superiority of StyleFlow over prior and several concurrent works. Project Page and Video: https://rameenabdal.github.io/StyleFlow.

CCS Concepts: • **Computing methodologies** → **Image manipulation**;

This work was supported by the KAUST Office of Sponsored Research (OSR) under Award No. OSR-CRG2018-3730, the ERC Grant SmartGeometry, and the Visual Computing Center at KAUST. We also thank Adobe Research for their support.
Authors' addresses: R. Abdal, P. Zhu, and P. Wonka, KAUST; emails: {rameen.abdal, peihao.zhu}@kaust.edu.sa, pwonka@gmail.com; N. J. Mitra, University College London and Adobe Research; email: nimitra@adobe.com.

0730-0301/2021/04-ART21 $15.00
https://doi.org/10.1145/3447648

Additional Key Words and Phrases: Generative adversarial networks, image editing

ACM Reference format:
Rameen Abdal, Peihao Zhu, Niloy J. Mitra, and Peter Wonka. 2021. StyleFlow: Attribute-conditioned Exploration of StyleGAN-Generated Images using Conditional Continuous Normalizing Flows. *ACM Trans. Graph.* 40, 3, Article 21 (April 2021), 21 pages.
https://doi.org/10.1145/3447648

1 INTRODUCTION

A longstanding goal of Computer Graphics has been to generate high-quality realistic images that can be controlled using user-specified attributes. One broad philosophy has been to create detailed 3D models, decorate them with custom materials and texture properties, and render them using realistic camera and illumination models. Such an approach, once realized, provides users with significant control over a range of attributes including object properties, camera position, and illumination. However, it is still hard to achieve photorealism over a range of attribute specifications.

Alternatively, in a recent development, **generative adversarial networks (GANs)** have opened up an entirely different image generation paradigm. Tremendous interest in this area has resulted in rapid improvements both in speed and accuracy of the generated results. For example, StyleGAN [Karras et al. 2019b], one of the most celebrated GAN frameworks, can produce high-resolution images with unmatched photorealism. However, there exist limited options for the user to control the generation process with adjustable attribute specifications. For example, starting from a real face photograph, how can we edit the image to change the (camera) pose, illumination, or the person's expression?

One successful line of research that supports some of the above-mentioned edit controls is *conditional generation* using GANs. In such a workflow, attributes have to be specified directly at (GAN) training time. While the resultant conditional GANs [Choi et al. 2019; Park et al. 2019; Xiao et al. 2018] provide a level of (semantic) attribute control, the resultant image qualities can be blurry and fail to reach the quality produced by uncontrolled GANs like StyleGAN. Further, other attributes, which were not specified at training time, can change across the generated images, and hence result in loss of (object) identity due to edits.

We take a different approach. We treat the attribute-based editing problem as a *conditional exploration* problem in an unsupervised GAN, rather than conditional generation requiring attribute-based retraining. We explore the problem using StyleGAN, as one of the leading uncontrolled GAN setups, and treat attribute-based user manipulations as finding corresponding *non-linear paths* in the StyleGAN's latent space. Specifically, we explore two problems: (i) *attribute-conditioned sampling*, where the goal is to sample a diverse set of images all meeting user-specified attribute(s); and (ii) *attribute-controlled editing*, where the user wants to edit, possibly sequentially, a particular image with target attribute specifications. The paths inferred by StyleFlow are conditioned on the input image, and hence can be adapted to the uniqueness of individual faces.

Technically, we solve the problem by proposing a novel normalizing flow-based technique to conditionally sample from the GAN latent space. First, assuming access to an attribute evaluation function (e.g., an attribute classification network), we generate sample pairs linking StyleGAN latent variables with attributes of the corresponding images. In our implementation, we consider a range of attributes including camera, illumination, expression, gender, and age for human faces; and camera, type, and color for cars. We then enable adaptive latent space vector manipulation by casting the conditional sampling problem in terms of conditional normalizing flows using the attributes for conditioning. Note that, unlike in conditional GANs, this does *not* require attribute information during GAN training. This results in a simple yet robust attribute-based image editing framework.

Fig. 2. Given an input StyleGAN image (left), vector arithmetic in the latent space [Radford et al. 2016] (here computed by Image2StyleGAN [Abdal et al. 2019]) can produce expression edits (middle) but at the cost of changing identity of the face due to the entangled latent space. In contrast, our proposed StyleFlow (right), by extracting non-linear paths in the latent space, enables expression edits while retaining identity.

We evaluate our method mainly in the context of human faces (both generated as well as real images projected into the relevant latent space) and present a range of high-quality *identity-preserving* edits at an unmatched quality (contrast against Figure 2 produced using latent space arithmetic and Figure 3 showing fine-grained edits). As a further proof of robustness, we demonstrate sequential edits (see Figure 1 and Figure 7) to the images without forcing the latent vectors out of the distribution, as guaranteed by our formulation of the problem with normalizing flows. We compare against concurrently proposed manipulation techniques and demonstrate superior identity preservation, both quantitatively and qualitatively.

In summary, we

(1) are the first to propose flow models for exploring the StyleGAN latent space;
(2) enable non-linear exploration of a GAN latent space and discover edit directions that are conditioned on the target images; and
(3) facilitate edits that are high quality, qualitatively and quantitatively, on both sampled and real images compared to previous and concurrent methods.

Project code and user interface (see Supplementary Material) will be released for research use.

2 RELATED WORK

Generative Adversarial Network Architecture. GANs have been introduced by Goodfellow et al. [2014] and sparked a huge amount of follow-up work. One direction of research has been the improvement of the GAN architecture, loss functions, and training

Fig. 3. Disentangled edits performed by our StyleFlow framework on real images. From top to bottom, changing camera view, expression, and age, respectively. Please note the non-local changes. For example, in the second row, the eyes and cheeks respond under increasing smile.

dynamics. As current state-of-the-art, we consider a sequence of architecture versions by Karras and his co-authors, Progressive-GAN [Karras et al. 2017], StyleGAN [Karras et al. 2019b], and StyleGAN2 [Karras et al. 2019a]. These architectures are especially strong in human face synthesis. Another strong system is BigGAN [Brock et al. 2018], which produces excellent results on ImageNet [Deng et al. 2009]. We build our work on StyleGAN2, as it is easier to work with. A detailed review of the history of GAN architectures, or a discussion on loss functions, and so on, is beyond the scope of the article. We also note that in addition to GANs, there are other generative models, such as **Variational Autoencoders (VAEs)** [Kingma and Welling 2013] or pixelCNN [van den Oord et al. 2016]. While these methods have some advantages, GANs currently produce the highest quality output for the applications we consider by a large margin.

Conditional GANs. A significant invention for image manipulations are conditional **GANs (CGANs)**. To add conditional information as input, CGANs [Mirza and Osindero 2014] learn a mapping $G : x, z \rightarrow y$ from an observed input x and a randomly sampled vector z to an output image y. One important class of CGANs use images as conditioning information, such as pix2pix [Isola et al. 2017], BicycleGAN [Zhu et al. 2017b], pix2pixHD [Wang et al. 2018], SPADE [Park et al. 2019], MaskGAN [Lee et al. 2019], and SEAN [Zhu et al. 2019]. Other notable works [Nie et al. 2020; Siarohin et al. 2020; Zakharov et al. 2019] produce high-quality image animations and manipulations. CGANs can be trained even with unpaired training data using cycle-consistency loss [Kim et al. 2017; Yi et al. 2017; Zhu et al. 2017a]. They can be used as a building block for image editing, for example, by using a generator G to translate a line drawing or semantic label map to a realistic-looking output image. CGANs have given rise to many application-specific modifications and refinements.

Applications of Conditional GANs. CGANs are an excellent tool for semantic image manipulations. In the context of faces, StarGAN [Choi et al. 2018, 2019] proposes a GAN architecture that considers face attributes such as hair color, gender, and age. Another great idea is to use GANs conditioned on sketches and color information to fill in regions in a face image. This strategy was used by FaceShop [Portenier et al. 2018] and SC-FEGAN [Jo and Park 2019]. One interesting aspect of these papers is that they use masks to restrict the generation of content to a predefined region. Therefore, some of the components of these systems borrow from the inpainting literature. We just mention DeepFillv2 [Yu et al. 2018] as a representative state-of-the-art technique. An early paper in computer graphics showed how conditioning on sketches can produce good results for terrain modeling [Guérin et al. 2017]. Specialized image manipulation techniques, such as makeup transfer PSGAN [Jiang et al. 2019] or hair editing [Tan et al. 2020] are also very useful, as faces are an important class of images. A very challenging style transfer technique is the transformation of input photographs to obtain caricatures [Cao et al. 2019]. This problem is quite difficult, because the input and output are geometrically deformed. Overall, these methods are not directly comparable to our work, because the input and problem statements are slightly different.

Image Editing by Manipulating Latent Codes. A competing approach to conditional GANs is the idea to manipulate latent codes of a pretrained GAN. Radford et al. [2016] observed that interesting semantic editing operations can be achieved by first computing difference vectors between two latent codes (e.g., a latent code for a person with beard and a latent code for a person without beard) and then adding this difference vector to latent codes of other people (e.g., to obtain an editing operation that adds a beard). Our technique falls into this category of methods that do not design a separate architecture, but manipulate latent codes of a pretrained GAN. This approach became very popular recently and all noteworthy competing papers are only published on arXiv at the point of submission. We would like to note that these papers were developed independently of our work. Nevertheless, we believe it will be useful for the reader to judge our work in competition with these recent papers [Härkönen et al. 2020; Nitzan et al. 2020; Tewari et al. 2020a], because they provide better results than other work. A great idea is proposed by the StyleRig [Tewari et al. 2020a] paper, where they want to transfer face rigging information from an existing model as a method to control face manipulations in the StyleGAN latent space. While the detailed control of the face

ultimately did not work, they have very nice results for the transfer of overall pose (rotation) and illumination. Based on earlier work, some authors worked on the hypothesis that the StyleGAN latent space is actually linear and they propose linear manipulations in that space. Two noteworthy efforts are InterFaceGAN [Shen et al. 2019] and GANSpace [Härkönen et al. 2020]. The former tries to find the latent space vectors that correspond to meaningful edits. The latter takes a data driven approach and uses PCA to learn the most important directions. Upon analyzing these directions the authors discover that the directions often correspond to meaningful semantic edits. Our results confirm that the assumption of a linear latent space is a useful simplification that produces good results. However, we are still able to produce significantly better disentangled results with a non-linear model of the latent space conditioned on the input image.

Embedding Images into the GAN Latent Space. We would also like to mention techniques that try to embed images into the latent space of a GAN. Generally, there are three main techniques. The first technique is to build an encoder network [Richardson et al. 2020] that maps an image into the latent space. The second technique is to use an optimization algorithm to iteratively improve a latent code so it produces the output image [Abdal et al. 2019, 2020; Karras et al. 2019a]. There is the idea to combine the two techniques and first use the encoder network to obtain an approximate embedding and then refine it with an optimization algorithm [Zhu et al. 2020a, 2016]. Finally, other methods [Zhu et al. 2020b] use VAEs to create inverse mappings. We will use the optimization-based technique of Karras et al. [2019a]. In addition, embedding can itself be used for GAN-supported image modifications. We will compare to one recent approach in our work [Abdal et al. 2019].

Neural Rendering. Neural rendering refers to the idea to generate images from a scene description using a neural network. We refer the reader to a recent state-of-the-art report [Tewari et al. 2020b] that summarizes recent techniques. Current methods tackle specific sub-problems such as novel view synthesis [Hedman et al. 2018; Sitzmann et al. 2019; Thies et al. 2020], relighting under novel lighting conditions [Guo et al. 2019; Xu et al. 2018], animating faces [Fried et al. 2019; Kim et al. 2018; Thies et al. 2019], and animating bodies [Aberman et al. 2019; Martin-Brualla et al. 2018; Shysheya et al. 2019] in novel poses. While these techniques share some similar goals in terms of user interaction, the overall problem setting is sufficiently different from our work, so it is hard to compare to these works directly.

3 OVERVIEW

We support two tasks: first, *attribute-conditioned sampling*, wherein we want to sample high-quality realistic images with target attributes; and, second, *attribute-controlled editing*, wherein we want to edit given images such that the edited images have the target attributes while best preserving the identity of the source images.

For generating realistic images, we use StyleGAN. In our implementation, we support sampling from both StyleGAN [Karras et al. 2019b] and StyleGAN2 [Karras et al. 2019a]. We recall that StyleGAN maps latent space samples $\mathbf{z}_s \in \mathbb{R}^{512}$ to intermediate vectors $\mathbf{w} \in \mathbb{R}^{512}$ in W space by learning a non-linear mapping $f : \mathbf{z}_s \rightarrow \mathbf{w}$, such that the $\mathbf{w}$-s can then be decoded to images $I(\mathbf{w}) = I(f(\mathbf{z}_s)) \in \mathbb{R}^{3\times1024\times1024}$. In the uncontrolled setup, $\mathbf{z}_s$ is sampled from a multi-dimensional normal distribution. Note that the $\mathbf{w}$ vector is used to control the normalization at 18 different locations of the StyleGAN2 generator network. The idea of the extended latent space $W+$ is to not repeat the same vector $\mathbf{w}$ 18 times, but use 18 different vectors. Hence, a vector $\mathbf{w} \in W+$ has dimensions 18×512. We evaluate with both of these latent spaces in our article. Specifically, for training the StyleFlow network, we use W; while, for restricting edits and editing real images, we use $W+$.

To measure attributes of any image, we assume access to a class-specific attribute function $\mathcal{A}$, typically realized as a classifier network, which returns a vector of attributes $\mathbf{a} := \mathcal{A}(I)$ for any given image I belonging to the class under consideration. The attributes are represented as an l-dimensional vector (e.g., $l = 17$ for human faces in our tests).

Solving the first task amounts to sampling $\mathbf{z}$ from a multi-dimensional normal distribution and using a learned mapping function of the form $\Phi(\mathbf{z}, \mathbf{a})$, where $\mathbf{a}$ denotes the target attributes, to produce suitable intermediate weights. These weights, when decoded via StyleGAN, produce attribute-conditioned image samples of the form $I(\Phi(\mathbf{z}, \mathbf{a}))$ matching the target attribute.

Specifically, using a zero-mean multi-dimensional normal distribution with identity as variance, we can conditionally sample as,

$$\mathbf{z} \sim N(\mathbf{0}, \mathcal{I}) \quad \text{and} \quad \mathbf{w} = \Phi(\mathbf{z}, \mathbf{a}) \tag{1}$$

and in the process satisfy $\mathcal{A}(I(\Phi(\mathbf{z}, \mathbf{a})) = \mathbf{a}$. In Section 5, we describe how to train and use a neural network to model such a function $\Phi(\mathbf{z}, \mathbf{a})$ using forward inference on a conditional **continuous normalizing flow (CNF)**. In other words, the normalizing flow maps the samples from an n-dimensional prior distribution, in this case a normal distribution, to a latent distribution conditioned on the target attributes.

Solving the second task is more complex. Given an image I_0, we first project it to the StyleGAN space to obtain $\mathbf{w}_0$ such that $I(\mathbf{w}_0) \approx I_0$ using [Abdal et al. 2019; Karras et al. 2019a]. Recall that our goal is to edit the current image attributes $\mathbf{a}_0 = \mathcal{A}(I(\mathbf{w}_0))$ to user specified attributes $\mathbf{a}_t$, whereby the user has indicated changes to one or multiple of the original attributes while best preserving the original image identity. We then recover latent variables $\mathbf{z}_0$ that lead to intermediate weights $\mathbf{w}_0$ using an inverse lookup $\mathbf{z}_0 = \Psi(\mathbf{w}_0, \mathbf{a}_0)$. We realize the inverse map using a reverse inference of the CNF network described earlier, i.e., $\Psi(\mathbf{w}_0, \mathbf{a}_0) := \Phi^{-1}(\mathbf{w}_0, \mathbf{a}_0)$. Let G be the StyleGAN generator. Finally, we perform a forward inference, using the same CNF, to get the edited image I_t that preserves the identity of the source image as,

$$\begin{aligned} I_t &= G(\Phi(\mathbf{z}_0, \mathbf{a}_t)) = G(\Phi(\Phi^{-1}(\mathbf{w}_0, \mathbf{a}_0), \mathbf{a}_t)) \\ &= G(\Phi(\Phi^{-1}(\mathbf{w}_0, \mathcal{A}(I(\mathbf{w}_0))), \mathbf{a}_t)). \end{aligned} \tag{2}$$

We first summarize normalizing flows in Section 4, and then, in Section 5, we describe how the invertible CNF can be used to compute the exact likelihood of the samples from the latent distribution of a generative model.

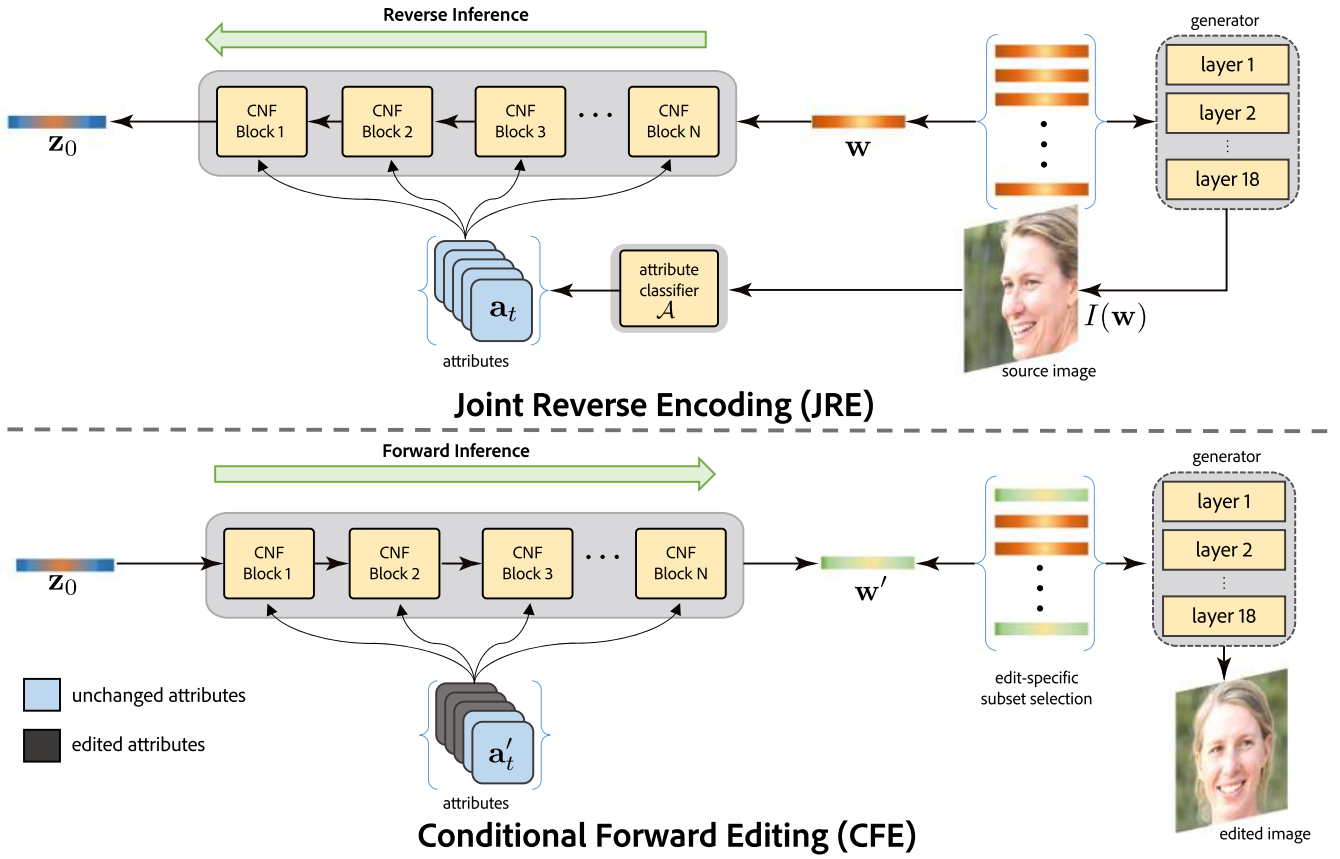

Fig. 4. Attribute-conditioned editing using StyleFlow. Starting from a source image, we support attribute-conditioned editing by using a reverse inference followed by a forward inference through a sequence of CNF blocks (see Figure 5). Here, $\mathbf{z}$ denotes the variable of the prior distribution and $\mathbf{w}$ denotes the intermediate weight vector of the StyleGAN. Also note that the reverse and the forward inferences are both solved by an ODE solver by evaluating CNF functions over the time variable. Please refer to the text for details.

4 NORMALIZING FLOWS

A normalizing flow, often realized as a sequence of invertible transformations, allows to map an unknown distribution to a known distribution (e.g., normal or uniform distribution). This inverse mapping, from an initial density to a final density, and vice versa, can be simply seen as a chain of recursive change of variables.

4.1 Discrete Normalizing Flows

Let $\phi : \mathbb{R}^d \to \mathbb{R}^d$ be a bijective map such that there exists an invertible map g with $g := \phi^{-1}$. Let the transformation of the random variable be from $\mathbf{z} \sim p_z(\mathbf{z})$ to $\mathbf{w}$ such that $\mathbf{w} = \phi(\mathbf{z})$. By the change of variable rule, the output probability density of variable $\mathbf{w}$ can be obtained as,

$$p_w(\mathbf{w}) = p_z(\mathbf{z}) \left| \det \frac{\partial \phi}{\partial \mathbf{z}} \right|^{-1} \tag{3}$$

where $\phi^{-1}(\mathbf{w}) = \mathbf{z}$ or $g(\mathbf{w}) = \mathbf{z}$. The same rule applies for a successive transformation of the variable $\mathbf{z}$. Specifically, let the transformation be represented by $\mathbf{w} = \phi_K(\phi_{K-1}(\dots\phi_1(\mathbf{z}_0)))$, i.e., $\mathbf{z}_0 \to \dots \mathbf{z}_{K-1} \to \mathbf{z}_K = \mathbf{w}$, and, since ϕ^{-1} exists the inverse mapping is expressed as $\mathbf{z}_0 = \phi_1^{-1}(\phi_2^{-1}(\dots\phi_K^{-1}(\mathbf{w})))$. Therefore, applying the change of variable rule, we obtain the modified output log probability density,

$$\log p_w(\mathbf{w}) = \log p_z(\mathbf{z}_0) - \sum_{n=1}^{K} \log \det \left| \frac{\partial \phi_n}{\partial \mathbf{z}_n} \right|, \tag{4}$$

where $\mathbf{z}_{n+1} = \phi_n(\mathbf{z}_n)$ and $\mathbf{z}_K = \mathbf{w}$.

In the special case of planar flows, the function ϕ can be modeled by a neural network [Rezende and Mohamed 2015] where the flow takes the form,

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \mathbf{u}_n h\left(\mathbf{w}_n^\top \mathbf{z}_n + b\right), \tag{5}$$

where $\mathbf{u}_n \in \mathbb{R}^d, \mathbf{w}_n \in \mathbb{R}^d, b \in \mathbb{R}$ are the learnable parameters, $h()$ is a smooth element-wise non-linear activation with derivative $h'()$. The probability density obtained by sampling $p_z(\mathbf{z}_0)$ and applying a sequence of planar transforms to produce variable $\mathbf{w} = \mathbf{z}_K$ takes the form,

$$\log p_w(\mathbf{w}) = \log p_z(\mathbf{z}_0) - \sum_{n=1}^{K} \log \left| 1 + \mathbf{u}_n^\top \xi(\mathbf{z}_{n-1}) \right|, \tag{6}$$

where $\xi(\mathbf{z}) = h'(\mathbf{w}^\top \mathbf{z} + b)\mathbf{w}$.

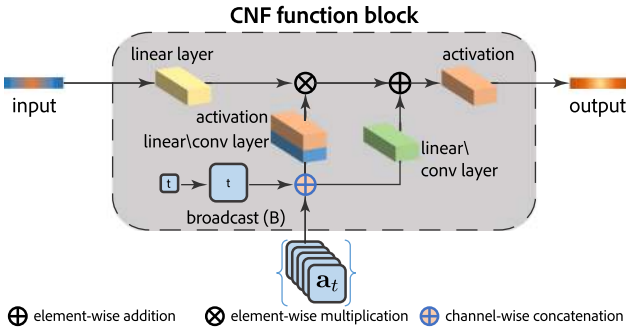

Fig. 5. Conditional continuous normalizing flow (CNF) function block realized as a neural network block. Note that the learned function, conditioned on attribute vector $\mathbf{a}_t$, can be used for both forward and backward inference.

4.2 Continuous Normalizing Flows (CNF)

The normalizing flows can be generalized into a continuous formulation [Chen et al. 2018; Grathwohl et al. 2018] using neural ODE [Chen et al. 2018], which adopts adjoint sensitivity method [Pontryagin 2018] to compute the gradients with respect to the parameters in an ODE black box solver. In continuous Normalizing flows, differential equations are expressed in the form: $\frac{dz}{dt} = \phi(\mathbf{z}(t), t; \theta)$ where z is the variable of a given distribution, t is the time variable, and θ are the parameters of an arbitrary neural network. Specifically, the differential equation takes the form,

$$z(t_1) = z\,(t_0) + \int_{t_0}^{t_1} \phi(z(t), t; \theta) dt.$$

Finally, the change in the log density can be expressed as,

$$\log p\,(\mathbf{z}\,(t_1)) = \log p\,(\mathbf{z}\,(t_0)) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial \phi}{\partial \mathbf{z}(t)}\right) dt. \tag{7}$$

We choose **Normalizing Flows (NF)**, as they produce tractable distributions where both sampling and density evaluation is efficient and exact [Kobyzev et al. 2020]. We decided not to use DNF networks, as it is difficult to ensure a reversible mapping. Another related problem is the expressiveness and versatility of such networks due to the fixed number of invertible functions to choose from. Finally, the Jacobian determinant computation is costly, so a workaround is to constrain the network architecture, which is also undesirable [Grathwohl et al. 2018].

In our StyleFlow framework, we use CNFs for our formulation. In contrast, the main benefit of the continuous formulation is that it is invertible by definition and the determinant is replaced by a matrix trace, which is considerably easier to compute. Hence, it allows to choose from a wide variety of architectures. Additionally, FFJORD [Grathwohl et al. 2018] also claims that CNFs can potentially learn a less entangled internal representation compared to DNFs (see Figure 2 in FFJORD). We would also like to point out that in this work, we interpret time as a "virtual" concept related to how CNFs are evaluated. Instead of evaluating the results by sequentially passing through the network layers, as in DNF, in CNFs the ODEs are used to evaluate the function through time. Hence, for both our conditional sampling and editing tasks, we condition based on the target attributes and use CNFs to continuously evolve the image samples. The CNFs are expected to retarget the probability densities, as described next.

5 METHOD

We consider the latent vectors $\mathbf{w} \in \mathbb{R}^{512}$ sampled from the W space of the StyleGAN1/2. The prior distribution is represented by $p_z(\mathbf{z})$, where $\mathbf{z} \in \mathbb{R}^{512}$. Our aim is to model a conditional mapping between the two domains. Moreover, it is imperative to be able to learn a semantic mapping between the domains so editing applications are realizable. Figure 4 shows the overall StyleFlow framework. We explain our method in the following subsections.

5.1 Dataset Preparation

A general workflow for the preparation of the dataset is as follows: We first sample 10K samples from the Gaussian $\mathbb{Z}$ space of the StyleGAN1/2 [Karras et al. 2019a, 2019b]. Then, we infer the corresponding $\mathbf{w}$ codes in the disentangled W space of the models. We use the vectors $\mathbf{w}$ of W space truncated by a factor of 0.7, as suggested by StyleGAN. Otherwise, there will be outlier faces of low quality. We generate corresponding images $I(\mathbf{w})$ via StyleGAN1/2 generator and hence create a mapping between the W space and the image space I. To have conditional control over the image features, we use a face classifier network $\mathcal{A}$ to map the images I to the attribute A_t domain. We use this dataset for the final training of the flow network using triplets $w \in W$, $i \in I$ and $a_t \in A_t$. Note that the attributes a_t do not depend on the variable t. To prepare the A_t domain on the training dataset, we use a state-of-the-art Microsoft Face API [Microsoft 2020], which we found to be robust for the face attribute classification. The API provides a diverse set of attributes given a face image. The main attributes that we focus on in our work are gender, pitch, yaw, eyeglasses, age, facial hair, expression, and baldness/hair length. For the lighting attribute, we use the predictions from the DPR model [Zhou et al. 2019] that outputs a nine-dimensional vector per image measuring the first nine spherical harmonics of the lighting. Thus, for faces, we have $a_t \in \mathbb{R}^{17}$. More details are provided in Table 1 of the Supplementary Material.

5.2 Attribute-translation Model

We use a series of the gate-bias modulation [Grathwohl et al. 2018; Yang et al. 2019] (called "ConcatSquash") networks to model the function ϕ of the conditional continuous normalizing flows. We make this design choice, as the gate part can learn the per-dimension scaling factor given an input latent code and the bias part of the network can learn to translate the code towards a particular edit direction that is suitable for our formulation of adaptive identity aware edits. This model builds on top of FFJORD [Grathwohl et al. 2018] as a general framework where the attributes can be 2D or 3D tensors like an image for an Image2Image translation task. Figure 5 shows the function block used in the CNF block. To include the condition information into the network, we transform the time variable t with a broadcast operation B to match the spatial dimensions of the attribute space. Then, we apply channel-wise concatenation to the resultant variable with the attribute variable a_t, and finally the new variable a_t^+ is fed to the network as a conditional attribute variable. Note that at inference time, we use

Fig. 6. Attribute-conditioned sampling using StyleFlow by resampling z_0 given the attributes. Here, we show sampling results for attribute specifications of *females with glasses in a target pose* (top); *50-year old males with facial hair* (middle); and *smiling 5-year old children in a target pose* (bottom).

linear interpolation in the attribute domain to smoothly translate between the two edits to get the final image.

For stable training, we used four stacked CNF functions (i.e., gate-bias functions) and two Moving Batch norm functions [Yang et al. 2019] (one before and one after the CNF functions), where each function outputs a vector of the same dimension (i.e., 512). We observed that using only two to three CNF function block models overfit on the data. The matrix trace is computed by 10 evaluations of Hutchinson's trace estimator. Depending on the properties of the extended a_t^+ tensor, we can use the convolutional or linear neural network to transform the tensors to make them the same shape as the input. We make use of linear layers in this work, but we expect extensions to this work where the flows can be conditioned on images or segmentation maps. Then, we perform gate-bias modulation on the input tensor. Note here, we use sigmoid non-linearity before the gate tensor operation [Grathwohl et al. 2018]. The final output tensor is passed through a $Tanh$ non-linearity before passing to the next stage of the normalizing flow.

An important insight of our work is that flow networks trained on one attribute at a time learn entangled vector fields, and hence resultant edits can produce unwanted changes along other attributes. Instead, we propose to use joint attribute learning for training the flow network. We concatenate all the attributes to a single tensor before feeding it to the network. In Section 7, we show that the joint attribute training produces an improvement in the editing quality with a more disentangled representation. We hypothesize that the training on single condition tends to over-fit the model on the training data. Further, in absence of measures along other attribute axis, the conditional CNF remains oblivious of variations along those other attribute directions. Therefore, the flow changes multiple features at a time. Joint attribute training tends to learn stable conditional vector fields for each attribute.

5.3 Training Dynamics

The goal during the training is to maximize the likelihood of the data $\mathbf{w}$ given a set of attributes a_t. So, the objective can be written as $max_\theta \prod_{w,a_t} p(w|a_t, \theta)$. Here, we assume the standard Gaussian prior with z as the variable. Also, let $\mathcal{N}$ represent the Gaussian probability density function. Algorithm 1 shows the training algorithm [Chen et al. 2018; Grathwohl et al. 2018] of the proposed joint conditional continuous normalizing flows. Other details are as follows: Epochs: 10; Batch size: 5; Training speed: 1.07–2.5 iter/sec depending on the number of CNF functions (see Table 4); GPU: Nvidia Titan XP; Parameters: 1,128,449; Final Log-Likelihood: −4,327,872; Total inference time: 1.31 sec (flow prediction time CFE (see Section 6.2): 0.61 sec, JRE (see Section 6.2), UI overhead and StyleGAN image generation : 0.70 sec). In practice, the total training time varies between 17–18 hours.

For faster inference, we also train a model with six stacked CNF functions, which improves the inference time (CFE) to 0.15–0.21 sec (total inference time 0.53 sec) at the cost of slight decrease in the quality of disentanglement. Note that this configuration might underfit the data (see Table 4). We use the adjoint method to compute the gradients and solve the ODE using the "dopri5" ODE solver [Grathwohl et al. 2018]. The tolerances are set to default 1×10^{-5}. We use the Adam optimizer with an initial learning rate of 1×10^{-3}. Other parameters (β_1, β_2) of the Adam optimizer are set to default values.

ALGORITHM 1: Flow training Algorithm

Input: Paired latent-attribute data ($\{w, a_t\}$); Neural network ϕ; Integration times t_0 and t_1; ODE solver with adjoint sensitivity method; Number of training steps N_t; Optimizer F'; Learning rate η.

Initialization:

$$\begin{bmatrix} \mathbf{z}(t_1) \\ \log p(\mathbf{w}|a_t) - \log p(\mathbf{z}(t_1)) \end{bmatrix} = \begin{bmatrix} \mathbf{w} \\ 0 \end{bmatrix}; \; a_t^+ = B(t) \,||\, a_t,$$

where B expands the variable t such that spatial dimension of a_t is equal to t and $||$ is the concatenation operation.

for $i = [1 : N_t]$ **do**

$$\begin{bmatrix} \mathbf{z_0} \\ \Delta_{\mathbf{logp}} \end{bmatrix} = \int_{t_1}^{t_0} \begin{bmatrix} \phi(\mathbf{z}(t), a_t^+; \theta) \\ -\mathrm{Tr}\left(\frac{\partial \phi}{\partial \mathbf{z}(t)}\right) \end{bmatrix} dt$$

$\mathcal{L} = \log \mathcal{N}(z_0; 0, I) - \Delta_{logp}$

$\theta \leftarrow \theta - \eta F'(\nabla_\theta \mathcal{L}, \theta)$;

end

6 ATTRIBUTE-CONDITIONED SAMPLING AND EDITING

A natural benefit of the training formulation of the framework is the sampling. In particular, the mapping learned between the two domains is used to produce a vector z given a w vector and vice versa. Moreover, we can manipulate the vectors in the respective domains and the changes translate to the other domain semantically from the editing point of view. Please refer to the supplementary video for interaction sessions.

6.1 Conditional Sampling

Once the network is trained, we are able to conditionally sample the $w \in W$ with the Gaussian prior modelled by the continuous normalizing flows. Formally, we set the attribute variable a_t to a desired set of values and then sample multiple $z \sim p(z)$. These vectors are then passed through the (trained) conditional CNF network. The learned vector field translates the vectors to produce the latent vectors w, which are then finally fed to the StyleGAN1/2 generator. In Section 7, we show the results of sampling given a set of attributes. We notice that the quality of the samples is very high as well as unedited attributes remain largely fixed. The conditional sampling is an important result of our work and validates the fact that the network is able to learn the underlying semantic representations, which is further used to perform semantic edits to the images.

6.2 Semantic Editing

Here, we show the procedure to semantically edit the images using the proposed framework. Note here the vector manipulations are adaptive and are obtained by solving a vector field by an ODE solver. Unlike the previous methods [Abdal et al. 2019; Härkönen et al. 2020; Shen et al. 2019] the semantic edits performed on the latent vectors w forces the resultant vector to remain in the distribution of W space ($p(w)$). This enables us to do stable sequential edits that, to the best of our knowledge, are difficult to obtain with the previous methods. We show the results of the edits in Section 7. Figure 4 illustrates the semantic editing procedure in StyleFlow. In the following, we will discuss the procedure and components of the editing framework.

6.2.1 Joint Reverse Encoding (JRE). The first step of the semantic editing operation in the StyleFlow framework is the Joint Reverse Encoding. Here, we jointly encode the variables w and a_t. Specifically, given a $w \in W$, we infer the source image $i \in I$. Note that we can also start with a real image and use the projection methods [Abdal et al. 2019; Karras et al. 2019a; Zhu et al. 2020a] to infer the corresponding w. Such procedures may render the vectors outside the original $p(w)$ distribution and hence makes the editing a challenging task. Later, we show that StyleFlow is a general framework that also works on real images. We pass the image I through the face classifier API [Microsoft 2020] and the lighting prediction DPR network [Zhou et al. 2019] to infer the attributes. Then, we use reverse inference given a set w and a_t to infer the corresponding z_0.

6.2.2 Conditional Forward Editing (CFE). The second step is the Conditional Forward Editing, where we fix the z_0 (this vector encodes the projection of the given image on the learned Z space), and to translate the semantic manipulation to the image I, we change the set of desired conditions, e.g., we change the age attribute from 20 yo to 60 yo (year old). Then, with the given vector z_0 and the new set of (target) attributes a_t', we do a forward inference using the flow model. Finally, we process the resulting vector w' to produce an editing vector.

6.2.3 Edit-specific Subset Selection. This is the third step of the StyleFlow editing framework. Studying the structure of the StyleGAN1/2, we choose to apply the given vector w' at the different indices of the $W+$[Abdal et al. 2019, 2020; Karras et al. 2019b] ($\mathbb{R}^{18\times512}$) space, depending on the nature of the edit, e.g., we would expect the lighting change to be in the later layers of the StyleGAN where mostly the color/style information is present. Empirically, we found the following indices of the edits to work best: Light (7−11), Expression (4−5), Yaw (0−3), Pitch (0−3), Age (4−7), Gender (0−7), Remove Glasses (0−2), Add Glasses (2−3), Baldness/Hair length (0−5), and Facial hair (5−9). The final step is the inference of the image from the modified latents. We refer to the framework without the Edit Specific Subset Selection Block as V1 and the final framework is referred to as V2. In Section 7, we show the importance of this module in improving the editing quality.

We support two options to edit: in the faster and approximate version, we do not reproject every time an edit is performed; and in the slower and accurate version, all the vectors (18 w-s) are reprojected in one pass after an edit. As the w code is manipulated every time by the w-s based on the edit-specific subset selection, some of these subsets overlap with others in a sequential edit and may make the edit unstable. In the fast option, occasionally there can be sudden jumps in the output image. Encoding these vectors back to space ensures that the flow network is aware of the changes made to the W/W+ space (identity-aware) of the StyleGAN1/2 and hence make the edits stable. Note that, since the previous and concurrent methods are not identity-aware, this problem also adds to the reason for failed sequential edits. So, in our work the vectors of resultant W+ space are re-mapped to a new set of z_0s using JRE followed by CFE and edit specific subset selection to perform a subsequent edit.

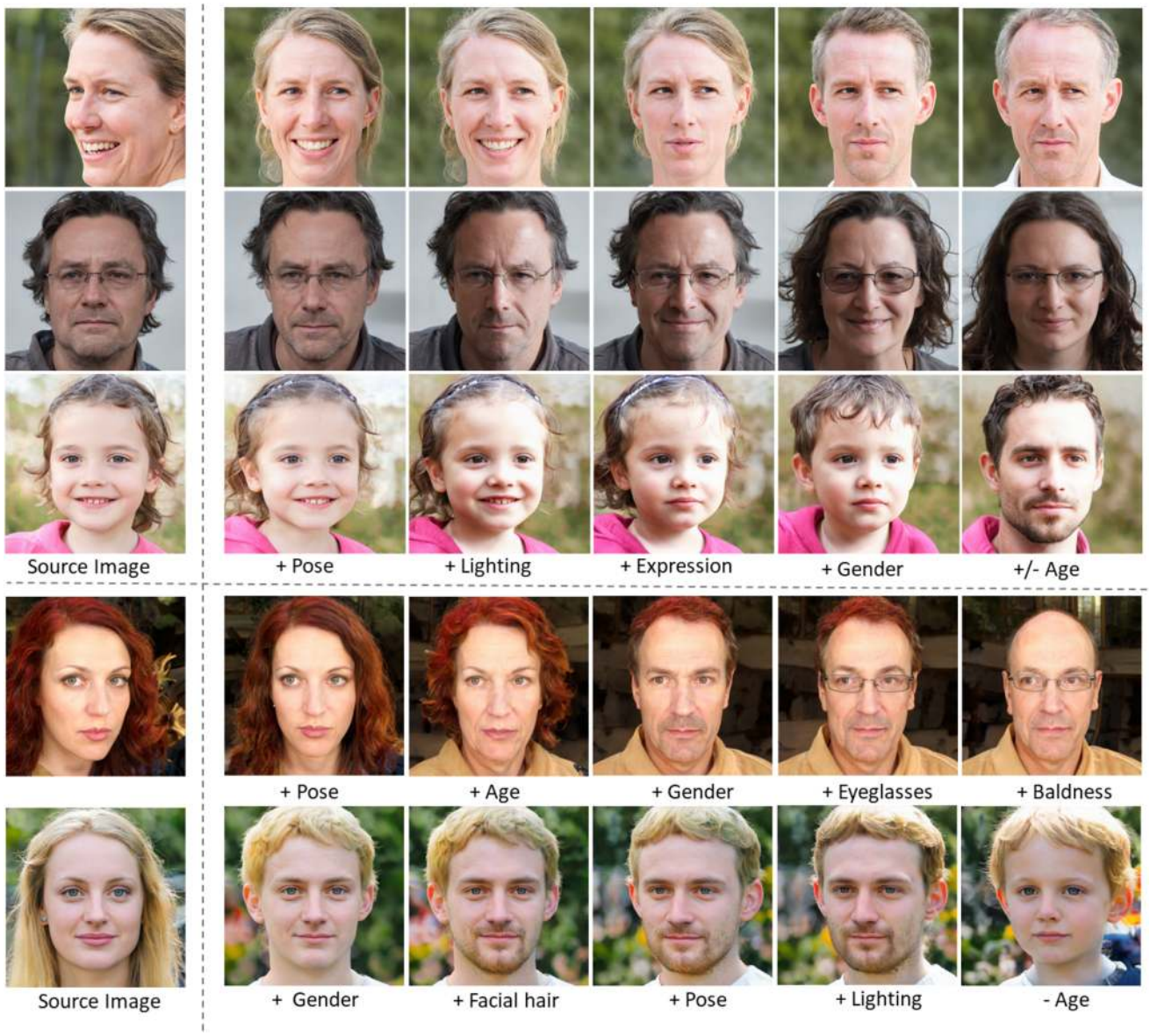

Fig. 7. Sequential edits using StyleFlow on synthetically generated faces with "+"/"-" denoting corresponding attribute was increased/decreased. Please notice the quality of preservation of attributes that are not being edited, demonstrating the disentanglement of the various attributes.

7 RESULTS

In this section, we discuss the results produced by our StyleFlow framework and compare them, both quantitatively and qualitatively, with other methods.

7.1 Datasets

We evaluate our results on two datasets—FFHQ [Karras et al. 2019b] and LSUN-Car [Yu et al. 2015]. **Flickr-Faces-HQ (FFHQ)** is a $1,024 \times 1,024$ resolution high-quality image dataset of human faces consisting of 70,000 images, which are diverse in terms of ethnicity, age, and accessories. LSUN-Car is a 512×384 resolution image dataset of cars consisting of 16,185 images, which are diverse in terms of car pose, color, and types. We use StyleGAN pretrained on these datasets to evaluate our results.

7.2 Evaluation Metrics

We evaluate the results of our framework and competing methods using three different types of metrics, namely, FID/LPIPS, face identity, and edit consistency scores.

7.2.1 FID and LPIPS Score. To measure the diversity and quality of the output samples, we use the FID score and LPIPS [Zhang et al. 2018] between the test images and the edited images. We evaluate the results with 1K generated samples from the StyleGAN2 framework.

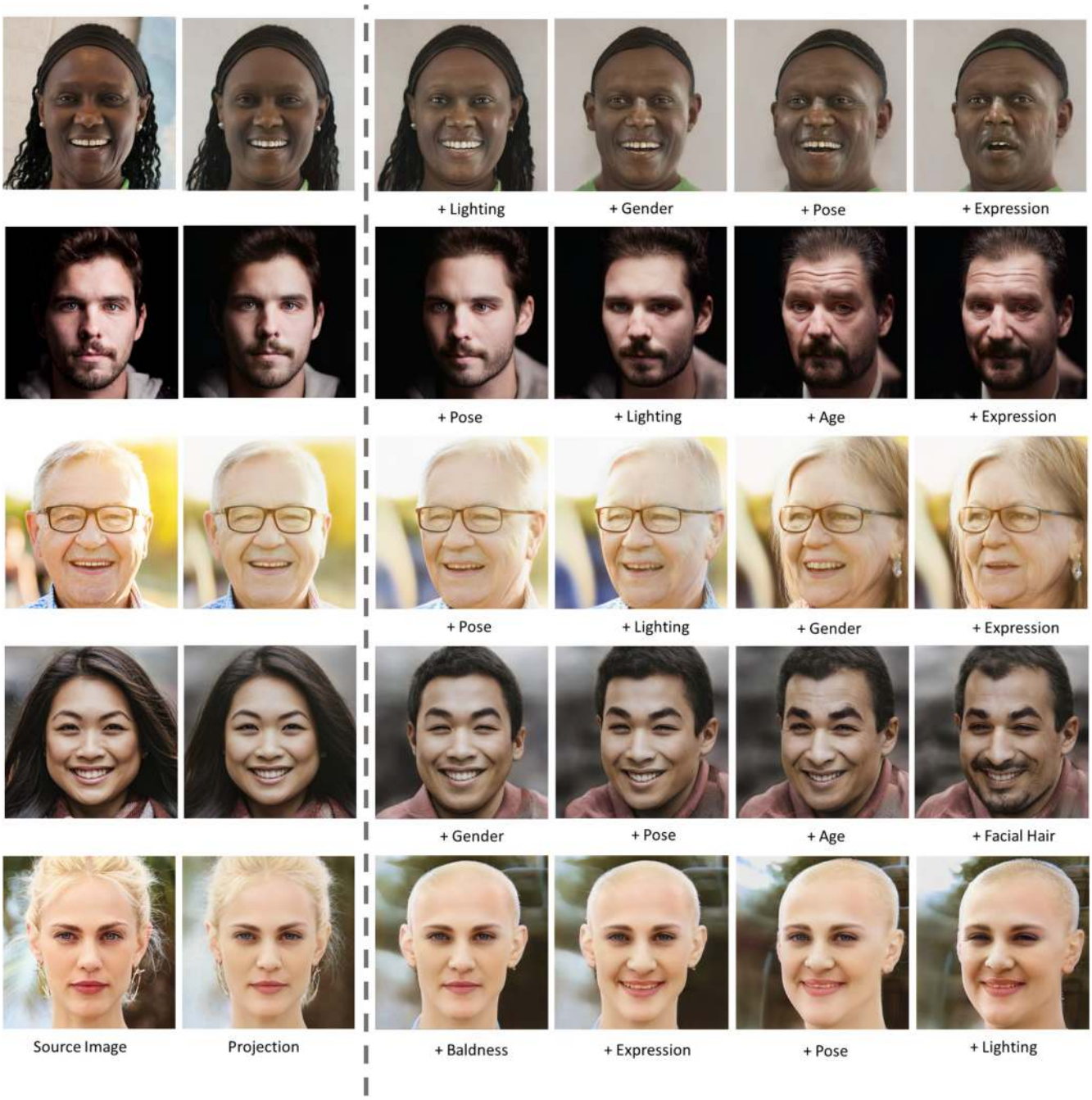

Fig. 8. Sequential edits using StyleFlow on real images. We show different permutations of the edits to demonstrate that a particular edit can appear anywhere in the sequence.

7.2.2 Face Identity Score. To measure the quality of the edit and quantify the identity preservation of the edits, we evaluate the edited images using a face identity score. We take a state-of-the-art classifier model for face recognition [Geitgey 2020] to output embeddings of the images. Given a pair of images, before and after edits, we calculate the Euclidean distance and the cosine similarity between the embeddings. Note that we use a different classifier from the attribute estimator used in training our StyleFlow. We choose three major edits for this purpose: light, pose, and expression.

7.2.3 Edit Consistency Score. To measure the consistency of the applied edit across the images, we evaluate over different edit permutations. For example, in a sequential editing setup, if the pose edit is applied, then it should not be affected by where in

Fig. 9. Non-sequential edits using StyleFlow on real images. Note that the method is able to handle extreme pose (first and second rows), asymmetrical expressions (fourth row), and age diversity (first and last rows) well compared to the concurrent methods.

the sequence it is applied. In principle, in the above case, different permutations of edits should lead to the same attributes when classified with an attribute classifier. Say, *ep* refers to an expression edit followed by a pose edit, while *pl* refers to a pose edit followed by a lighting edit. We expect the pose attribute to be the same when evaluated on the final image. We measure this using the score $|\mathcal{A}_p(E_p(E_e(I)) - \mathcal{A}_p(E_l(E_p(I))|$, where E_x denotes conditional edit along attribute specification x and $\mathcal{A}_p$ denotes pose attribute vector regressed by the attribute classifier.

7.3 Compared Methods

We compare to a simple version of vector arithmetic as demonstrated in Image2StyleGAN. Additionally, we compare to three concurrent works: InterfaceGAN, GANSpace, and StyleRig.

InterfaceGAN and Image2StyleGAN were retrained using StyleGAN2. GANSpace and StyleFlow are naturally implemented in StyleGAN2. However, StyleRig uses StyleGAN1.

(i) Image2StyleGAN: For Image2StyleGAN, we embedded paired images of expression (IMPA-FACES3D [IMPA-FACE3D 2012]), lighting pairs from DPR, and the rotation pairs using StyleFlow outputs. The lighting part of both Image2StyleGAN and InterfaceGAN is trained for right-to-left illumination change.

(ii) InterfaceGAN [Shen et al. 2019]: We retrained InterfaceGAN on the same data as StyleFlow. The images were segregated based on the attributes to create the binary data. For the magnitude of edits, note that it is difficult for competing methods to produce the same magnitude of changes when a given vector is applied for different faces. For example, in Figure 1 Supplementary Material, top row, the target rotation is a complete flip, for InterfaceGAN, the learned vector is translated till it matches the extreme pose. Hence, instead of evaluating the results by considering the attributes as binary, we use three continuous metrics, as previously described.

(iii) GANSpace [Härkönen et al. 2020]: We used the code provided by the authors and use the version using layer subsets. Note that to match the edits for the generated images, we used the sigmas −15 for the expression (index 46), −3 for the pose (index 1) and 10 for the LR lighting direction (index 12) from the official GANSpace open source implementation. For real images, we use the following sigmas: Expression −15, Pose 2 and LR lighting direction 7 to match the edits.

(iv) StyleRig [Tewari et al. 2020a]: The comparison results used in the paper were prepared by the authors of StyleRig. We would like to reiterate that the last three competing methods were only available on arXiv at the time of submission and were independently developed.

7.4 Qualitative Comparison of Edits

We show sequential edits on real images projected to StyleGAN2 by re-implementing the Image2StyleGAN W+ projection algorithm in Figures 1 and 8. In addition to the sequential edits, we show non-sequential edits in Figure 9. Note that we demonstrate results on images with diverse pose, lighting, expressions, and age attributes. For example, in row 4 in Figure 9, even though the expression is asymmetrical, StyleFlow handles the edits well.

Figure 7 shows the results of the sequential edits on generated images using our framework. Here, we consider the sequential edits of Pose → Lighting → Expression → Gender → Age. To show that different permutations of the edits can be performed without affecting the performance, Figure 7 and Figure 8 show the results of a random sequence of edits performed on a source image. Here, we consider multiple edits of gender, facial hair, pose, lighting, age, expression, eyeglasses, and baldness/hair length. Note the quality of the edits. Also, notice that the order of the edits does not affect the quality of the images. We can handle extreme pose changes while smoothly transferring the edits as the attributes change. Global features such as background, cloths, and skin tone are largely preserved. Moreover, we also show high-quality results for attribute transfer in Figure 10.

Other approaches that directly manipulate the latent space, e.g., adding offset vectors, are not able to achieve the same quality because vector manipulations often move the final latent into a region outside the distribution [Karras et al. 2019a]. StyleFlow deviates from these methods. First, attribute-guided edits amount to non-linear curves in the StyleGAN latent space. Second, the above curves (or even their linear approximations; see Figure 11) are conditioned on the current identity. Note that this is in contrast to edit vectors being independent of current identity as in GANSpace and InterfaceGAN (see Figures 14 and 15 for comparison).

We also compare with the StyleRig method. StyleRig has not been designed to work with real images, while StyleFlow can be applied to projected real images (see Figure 1, Figure 15, Figure 14, and Figure 9). StyleRig also only supports a subset of edits. Additionally, while the sequential edits are important in practice, StyleRig does not work well in this setting. Figures 14 and 15 compare with results kindly produced by the StyleRig authors on our test scenario (four representative images taken from real image dataset). We show that the quality of the edits by our method on attribute transfer are of similar or higher visual quality.

7.5 Quantitative Comparison of Edits

First, we would like to analyze how much the edits depend on the initial latent vector. For an edit $w \rightarrow w'$, we can compute the difference vector between the final latent vector w' and the initial latent vector w. The linear models Image2StyleGAN and InterfaceGAN make the assumption that the difference vectors are independent of the starting latent vector w if the same edit is applied. We perform the following test to evaluate how much we deviate from this assumption. Given many edits of the same type (e.g., changing a neutral expression to smiling by translating the attribute from 0 to 1), we compute their difference vectors $w' - w$. Then, given a set of pairs of these vectors, we compare the magnitude and the angles between the vectors. By sampling multiple such edits, the mean of the magnitudes (norm) of these difference vectors is computed to be 12.5, indicating the adaptive nature of the edits. The angles between the vectors are observed to vary up to 36°. This shows that the edits depend on the initial latent w allowing the resultant vector to follow the original posterior distribution. Hence, we condition the edits w.r.t. the source models to produce higher-quality edits.

To assess the *non-linearity* of the edit path, we compare the interpolation in the attribute domain (a_t) to the interpolation in the latent domain (w), i.e., we linearly change the variable of the attribute that is fed to the flow model versus linear interpolation of the vector w to w'. We sample 20 points along the interpolation paths of both scenarios and then compare the latents produced by both the methods. We compute the mean of the norm of these difference vectors along the path. Sampling multiple edits produced by StyleFlow, we conclude that, on average, the linear interpolation in the w domain differs from the attribute domain a_t by a factor of 1.5. In Figure 11, we compare the results of the non-linear path edits with the linear interpolation visually. Note, here the final w' is obtained by subjecting StyleFlow to extreme edits. We notice that the non-linear path is able to retain hair style, clothes, and head coverings for a larger extent along the path, validating the improvement in the disentanglement.

We perform sequential edits to the generated images as shown in Supplementary Material, Figure 1, i.e., "Pose," "Light," and

Fig. 10. Attribute-conditioned edits where source images are edited using multiple attributes from the target images. Upper set uses pose, expression, and illumination from the respective target images; bottom set uses eyeglass, facial hair, age, and gender from the respective target images.

"Expression." In Table 1, we show that the FID score for our method is lower (better) than the FID score of other methods.

Identity preservation results are presented in Table 2. The metrics show that our method outperforms others across all metrics and all edits. One exception are expression edits for which GANSpace also performs well. Also, we evaluate the scores when all the edits are applied sequentially. Here, our method also shows superiority in quantitative evaluation. Moreover, we also compute the accuracy based on the final decision of the classifier of the two embeddings being the same face.

In Table 3, we show the cyclic edit consistency evaluation of our method and compare it with other methods. As shown in Table 3, our editing method remains consistent under different permutations. We used mean (absolute) error across the respective attributes. Note that for GANSpace, since only four disentangled edits match our attributes, i.e., gender, expression, pose, and lighting, we are restricted in the comparison. In case of the expression, we notice that the attribute classifier outputs binary values, so the consistency scores for every method is zero. We make the same observation for gender change.

7.6 Choice of Encoding and Subset Selection

We evaluate the two design choices of joint attribute encoding and edit specific subset selection. As explained, we perform joint attribute encoding to ensure the face identity is preserved during the **Conditional Forward Editing (CFE)**. Figure 12 shows the variation of the proposed approach when attributes are trained jointly versus separately. The results show that in case of the joint encoding of the attributes, the identity of the face and the unedited attributes such as hair style, age, and background are better preserved. We evaluate the effectiveness of the edit-specific subset selection block in Figure 13. We notice that the edit done with the V2 framework performs high-quality edits producing images with comparable the skin tone, background, and clothes with respect to the source image. We also provide an ablation study evaluating the architecture with regards to different numbers of CNF function blocks in Table 4.

7.7 User Study

To evaluate the visual quality and the identity preservation of the images after the performed edits, we set up a user study. In a pairwise test setup, we asked, "Which of the two edited images better preserves the identity of the person in the original image?" and "Choose which edited image among the two is more realistic and of higher visual quality?" We compare four different methods, i.e., InterfaceGAN, GANSpace, StyleRig, and Styleflow. We consider five types of edits common to all the methods: expression, pose, lighting, and sequential Edits in the form of lighting + pose and lighting + pose + expression. We collected 18 diverse real images for the evaluation (4 different examples shown in Figures 14 and 15). Since we perform pairwise comparisons (one edit at a time), our evaluation contains 1,080 different comparisons per task. We divide these comparisons into 18 comparisons of 60 image pairs per task. We asked 25 people to perform both the tasks. Figure 16 shows the results of the user study comparing two methods at a time and aggregating all the edit scores. The results show that StyleFlow outperforms others in terms of visual quality and identity preservation. Table 5 shows quantitative results of identity preservation on the real dataset. While StyleRig has a better evaluation of the initial projected image, after editing, our method is better on all edit types, except for expression edits. We also encourage the reader to visually evaluate the 18 example edits included in Supplementary Material.

7.8 Qualitative Editing Results on Cars

We also show the editing results of our framework with StyleGAN2 trained on the LSUN-Car dataset, Figure 17 shows qualitative results of our framework. We show sequential edits including SUV/Hatchback conversion, rotation, and color change. We use a

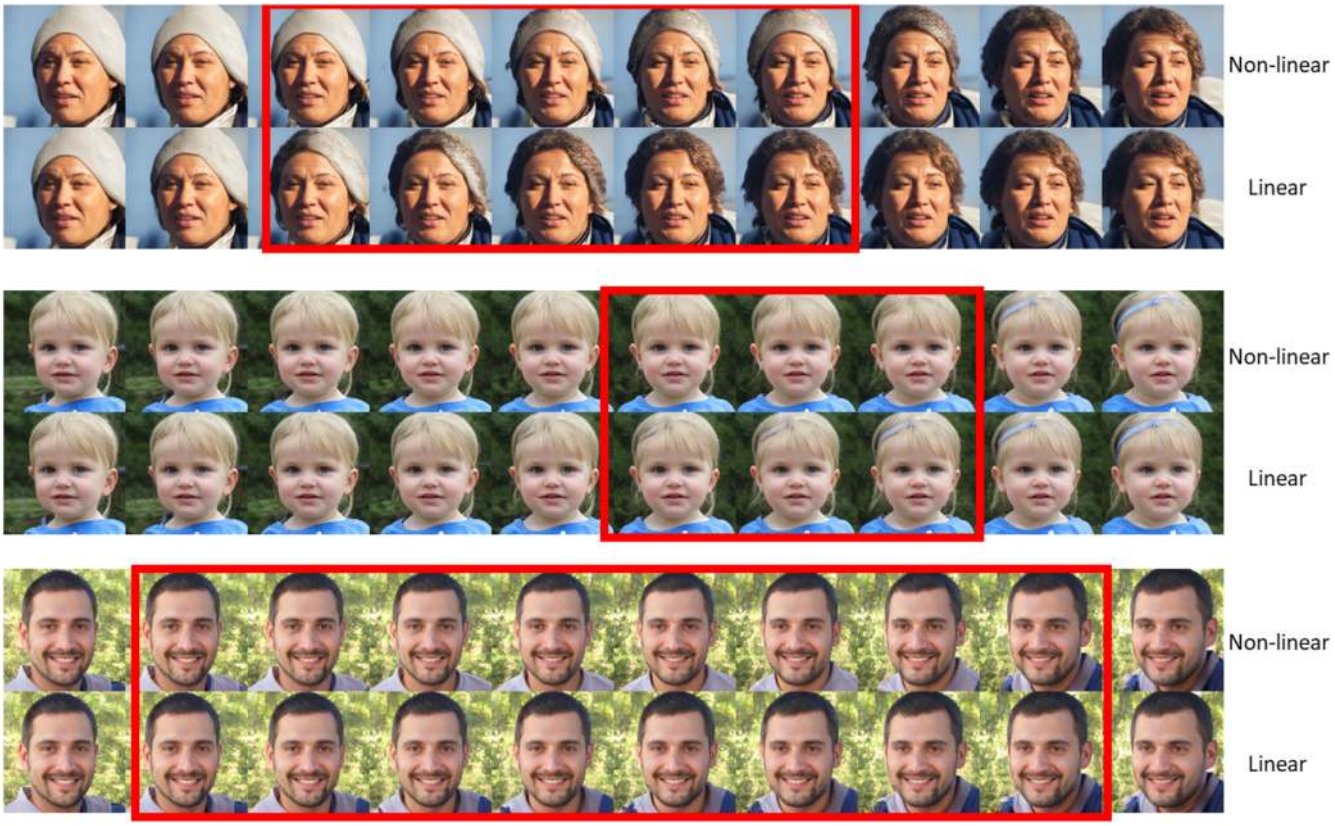

Fig. 11. Exploration of the non-linear vs. linear edit paths produced by interpolating the StyleFlow variables a_t vs w. The top row in every example is a non-linear path and the bottom row represents a linear path. The red box shows the region with the largest differences.

Table 1. Using FID (Fréchet Inception Distance) Score and LPIPS [Zhang et al. 2018] to Compare Realism of Sequential Edits Using Different Methods

Sequential edit	*FID* ⇓	*LPIPS* ⇓
Image2StyleGAN	82.49	0.64
InterfaceGAN	67.08	0.65
GANSpace	64.69	0.62
Our(V2)	**53.15**	**0.57**

Table 2. Identity Preservation Achieved by Different Methods as Evaluated by a SOTA Face Classifier [Geitgey 2020]

Edit	Metric	*I2S*	*IG*	*GS*	Ours(V1)	Ours(V2)
Light	C_s ⇑	0.910	0.945	0.942	0.958	**0.963**
	E_d ⇓	0.633	0.508	0.524	0.438	**0.394**
Pose	C_s ⇑	0.877	0.940	0.939	0.952	**0.966**
	E_d ⇓	0.748	0.532	0.526	0.466	**0.400**
Expression	C_s ⇑	0.941	0.946	**0.973**	0.951	0.967
	E_d ⇓	0.534	0.509	**0.359**	0.472	0.388
All	C_s ⇑	0.870	0.895	0.902	0.923	**0.941**
	E_d ⇓	0.774	0.690	0.681	0.564	**0.529**
	Acc ⇑	0.000	0.300	0.550	0.900	**0.950**

Please refer to the text for details. *Notations: I2S - Image2StyleGAN; IG - InterfaceGAN; GS - GANSpace; C_s - Cosine Similarity; E_d - Euclidean Distance; Acc: Accuracy.*

fine-tuned ResNet-152 [He et al. 2016] model trained on Stanford Cars [Krause et al. 2013] to create the attributes. For car manipulation, we used a car recognition model [Spectrico 2020] with 95% classification accuracy, we report the accuracy for the Hatchback/SUV conversion as 80% and the color as 100%. For the rotation, there is no precise model in the literature to evaluate the scores quantitatively, hence, we only show multiple visual examples in the supplementary video.

7.9 Attribute-conditioned Sampling

We show the results of the conditional sampling of StyleGAN2 in Figure 6. In the first row, for instance, we sample females of different age groups with glasses and fixed pose. Note that, during the sampling operation, we resample z to infer vectors in w and keep a set of attributes fixed. Apart from the quality of the samples, we find the diversity of the samples to be also high.

7.10 StyleFlow Editing Interface

To enable interactive editing, we developed an image editing interface (see Figure 18) that allows a user to select a given real or generated image and perform various edits with the help of interactive sliders. For sequential edits, the checkpoint images are

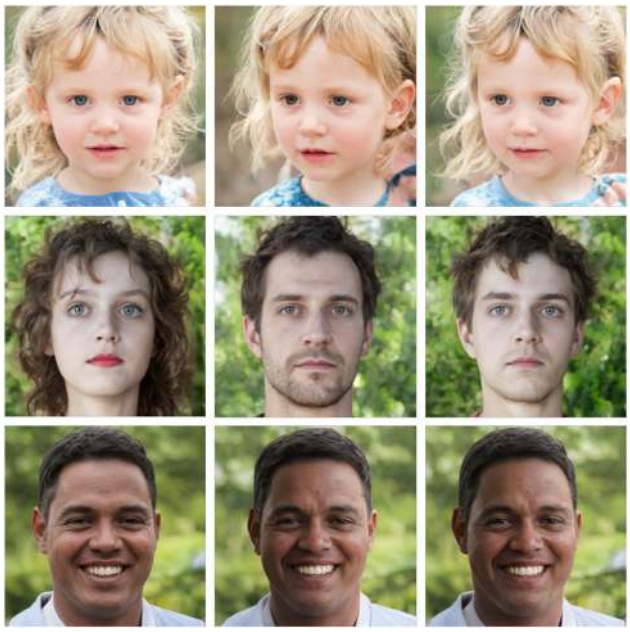

Fig. 12. Effect of training conditional CNFs with single attribute versus simultaneously along multiple attributes. In this example, from top-to-bottom, the target edits: pose change only, gender change only, illumination change only. In the single-attribute case (second column), the edits result in changes along other attributes: additional identity changes, additional age changes, and additional pose changes, respectively. In contrast, in the multi-attribute case (right column), other attributes are better preserved.

Fig. 13. Importance of edit-specific subset selection. (Left) Input image; (middle) changes performed without edit-specific subset selection block; (right) changes performed with edit-specific subset selection block. As seen, the subset selection results in better preservation of the background. Please refer to the text for details.

saved in a panel so a user can revisit the changes made during the interactive session.

8 DISCUSSION

Here, we describe different aspects of the different editing methods and how the design choices effect the results.

Table 3. Edit Consistency Evaluation (Mean Absolute Error) to Compare Different Methods under Permutation of Edit Operations

Edit	*I2S*	*IG*	*GS*	Ours(V2)
Pose ($ep-pl$)	4.68	10.18	10.53	**1.64**
Light ($le-pl$)	0.83	0.66	0.58	**0.53**
Facial hair ($fl-pf$)	0.31	0.23	-	**0.19**

Notations: I2S - Image2StyleGAN; IG - InterfaceGAN; GS - GANSpace; e - expression; p - pose; l - light; f - facial hair

Table 4. Ablation Study of StyleFlow CNF Functions

Stack	Parm	Infer	LL	Result
2	565249	0.40	−4323537	Overfit
3	846849	0.37	−4325796	Overfit
4	1128449	0.61	−4327872	Selected model
6	1691649	0.21	−4328470	Faster alternative

Notation: Stack - Number of stacked functions; Parm - Number of parameters; Infer - Inference time (sec); LL - Final Log likelihood (higher the better).

8.1 Edit Specific Subset Selection

GANSpace and StyleFlow both use edit-specific subset selection. This is an advantage over InterfaceGAN (and possibly StyleRig) and contributes to disentanglement in our results. This might be the reason why GANSpace seems to perform better than InterfaceGAN.

8.2 Conditioning Editing Direction on the Starting Image

One key difference between our method and others such as Image2StyleGAN, InterfaceGAN, and GANSpace, is that the editing direction depends on the starting latent. These other three competing methods compute a single editing direction **d** for an attribute (e.g., expression or pose) and apply this same editing direction to all starting latents **w** or **w**+ by scaling and adding the vector **d**. In contrast, our methods as well as StyleRig compute an edit direction that depends on the starting latent. Our results show that this design choice contributes to better disentanglement.

8.3 Supervised vs. Unsupervised Edit Discovery

Our method as well as StyleRig, InterfaceGAN, and Image2StyleGAN are supervised. They need to have a training corpus of latent vectors/images that are labeled with attributes. In contrast, GANSpace discovers edits in an unsupervised manner. GANSpace finds a large number of edits that could be interesting and then labels them semantically using visual inspection. Our results show that the unsupervised discovery of edit directions works reasonably well. It sometimes leads to edits that are not identity-preserving, as multiple changes are entangled together. However, GANSpace has the ability to discover cool edits for which labels might not be available. One example is the expression where the mouth forms an O-shape.

8.4 Binary Attributes vs. Continuous Attributes

Our method can create edits by specifying a continuous parameter for a given attribute. InterfaceGAN and GANSpace just give an edit direction and the user has to manually tune the scaling

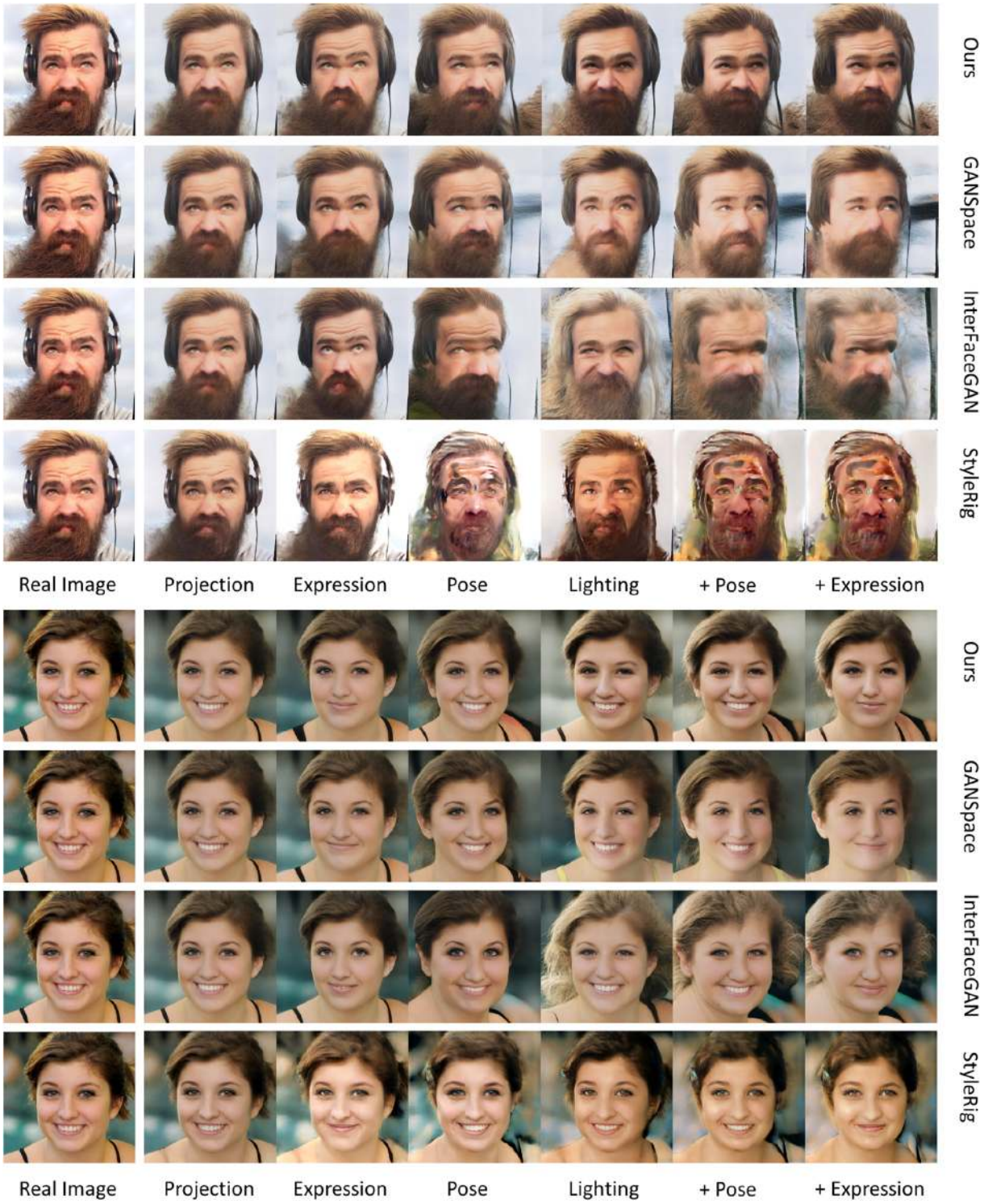

Fig. 14. Real image editing comparison with competing methods. We compare our method with GANSpace, InterFaceGAN, and StyleRig.

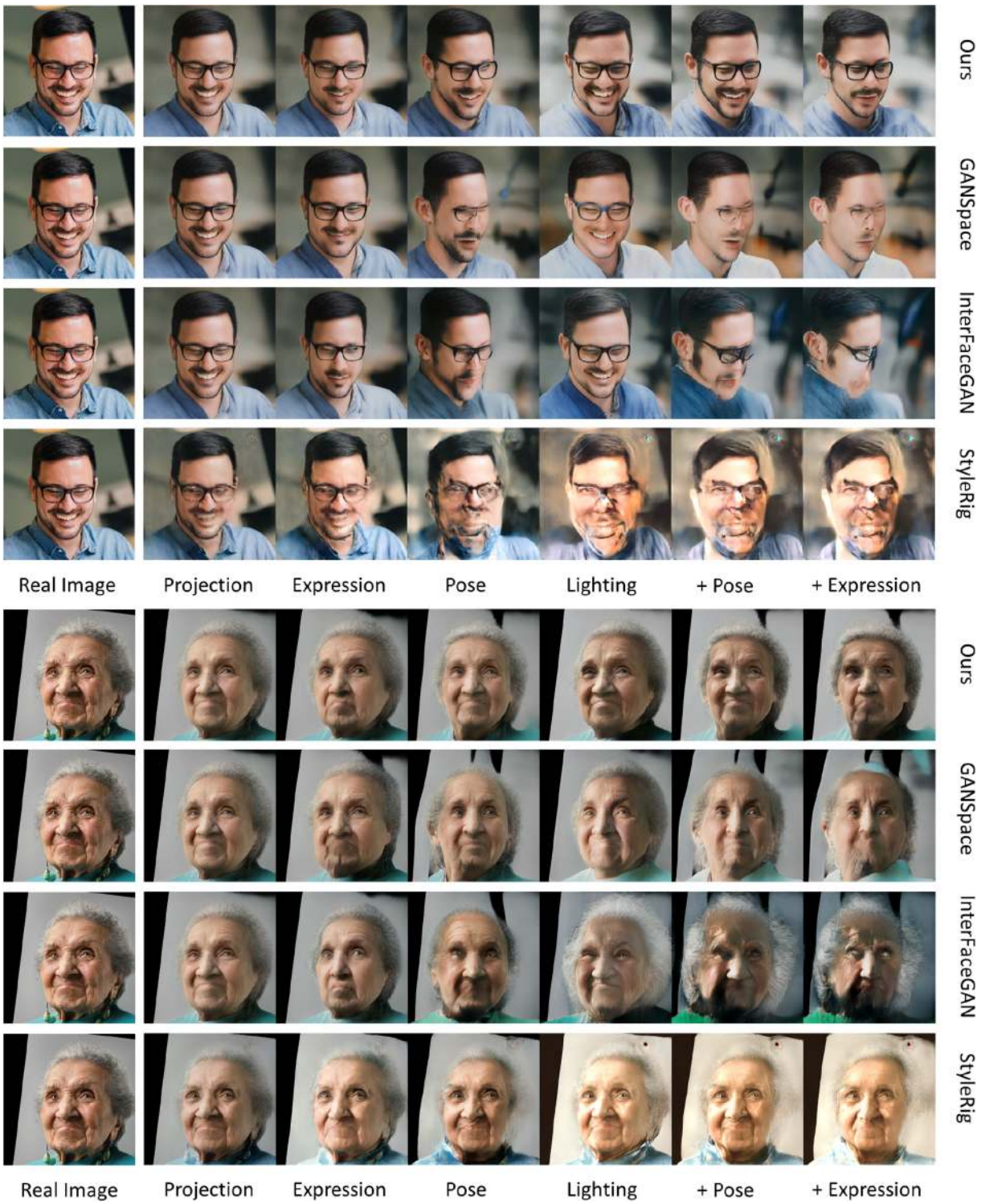

Fig. 15. Real image editing comparison with competing methods (continued). We compare our method with GANSpace, InterFaceGAN, and StyleRig.

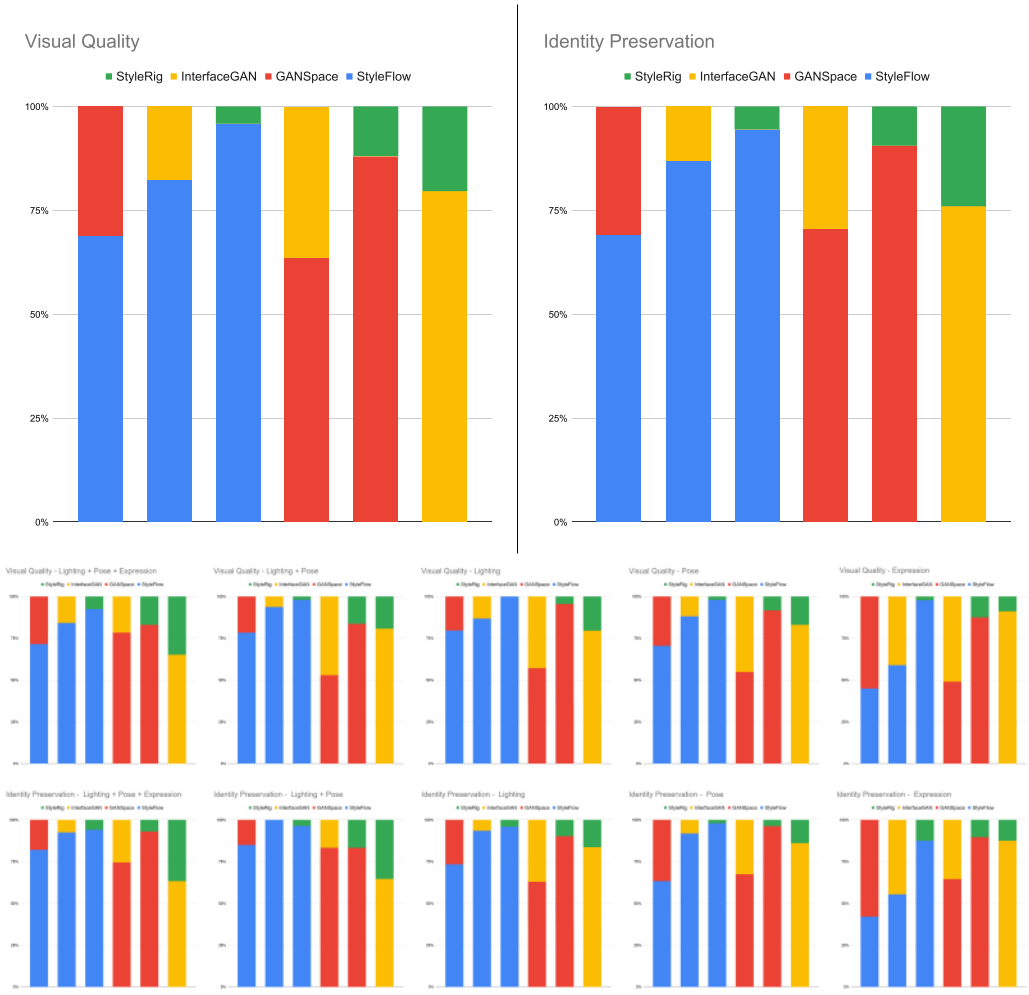

Fig. 16. Top Row: Combined user study results assessing visual quality of the edits (left) and identity preservation (right). Middle Row: Individual user study results assessing visual quality following the edits in Figure 14 and Figure 15. Last Row: Individual user study results assessing identity preservation following the edits in Figure 14 and Figure 15.

parameter to control the strength of an edit. That makes the edits not directly comparable and we have to manually tune this scaling parameter of these other methods to make it approximately match the strength of our edit.

8.5 Linear vs. Non-linear Interpolation Path

The edit paths in latent space of ours are non-linear. This is in contrast to StyleRig, InterfaceGAN, and GANSpace that use linear trajectories. Our evaluation indicates that this difference is not the only factor in explaining the edit quality of our method. It is also possible to set up our method approximating the nonlinear edit path by a linear one to achieve better results than all competing works.

8.6 Training One Attribute at a Time vs. Multiple Attributes

In our results, we show that the quality of edits improves if we train StyleFlow using all attributes at the same time. This helps disentanglement and provides higher-quality edits than training a network for each attribute separately. This is in contrast to StyleRig. In the StyleRig paper, the results are shown by training for each edit separately. In additional materials, the authors demonstrate that it is also possible to train for all edits jointly with some loss in quality. We believe that our behavior is more intuitive.

8.7 Individual vs. Sequential Edits

Previous work often focuses on applying a single edit to a starting image. In our experience, the quality of an editing framework becomes more obvious when applying sequential edits. Small errors in disentanglement accumulate and the initial face much easier loses its identity when subsequent edits are applied. While we also show results for individual edits, we primarily focus on sequential edits in the article to provide a more challenging evaluation setup.

8.8 What Type of Edits are Possible?

The quality of edits depends on the availability of good attribute labels and a good training dataset for StyleGAN. In general, the face

Table 5. Identity Preservation Achieved by Different Methods as Evaluated by a SOTA Face Classifier [Geitgey 2020] on Real Image Dataset

Edit	Metric	*IG*	*GS*	*SR*	Ours
Projected Image	$C_s \Uparrow$	0.980	0.980	**0.988**	0.980
	$E_d \Downarrow$	0.283	0.283	**0.210**	0.283
Expression	$C_s \Uparrow$	0.963	**0.969**	0.968	0.964
	$E_d \Downarrow$	0.387	0.356	**0.353**	0.380
Pose	$C_s \Uparrow$	0.899	0.961	0.954	**0.967**
	$E_d \Downarrow$	0.468	0.390	0.414	**0.364**
Light	$C_s \Uparrow$	0.952	0.902	0.955	**0.962**
	$E_d \Downarrow$	0.440	0.458	0.414	**0.394**
Light + Pose	$C_s \Uparrow$	0.874	0.893	0.941	**0.951**
	$E_d \Downarrow$	0.569	0.500	0.473	**0.444**
Light + Pose + Expression	$C_s \Uparrow$	0.765	0.884	0.925	**0.941**
	$E_d \Downarrow$	0.641	0.533	0.540	**0.484**

Notations: IG - InterfaceGAN; GS - GANSpace; SR - StyleRig; C_s - Cosine Similarity; E_d - Euclidean Distance.

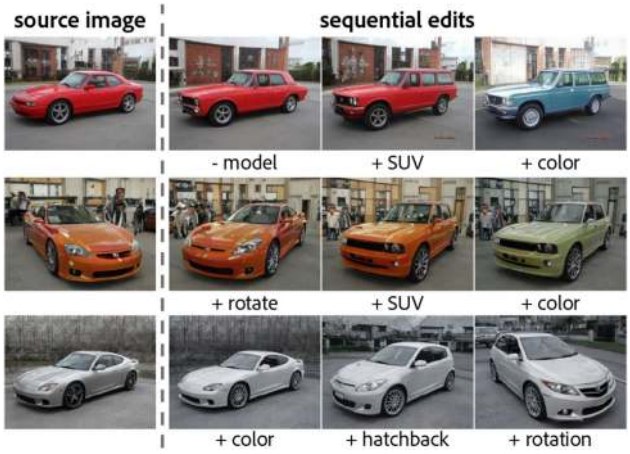

Fig. 17. Sequential attribute-conditioned edits using StyleFlow on the car latent space of StyleGAN.

dataset has the highest quality attributes and the highest quality latent space. Also, StyleRig can only work on faces. We therefore focus mainly on faces in our evaluation. However, there is nothing specific to faces, and our work can be applied to other suitable datasets. For faces, InterfaceGAN and StyleFlow can produce edits for the same set of attributes. However, StyleRig does not include several common attributes such as hair length, gender, eyeglasses, age. We are therefore also restricted in our evaluation to a subset of the attributes when comparing to StyleRig.

8.9 Editing Real vs. Synthetic Images

In principle, there is no difference between editing real images and editing synthetic images. When editing real images, it becomes important to have a good projection method into latent space. We use a reimplementation of Image2StyleGAN for StyleGAN2. This method seems to be sufficient to get good edits, but multiple researchers are working on improved embedding algorithms. Analyzing the compound effect of embedding and editing is beyond the scope of this article. We just would like to note that it is essential to embed into $\mathbf{w}+$ latent space and not into $\mathbf{w}$ latent space as for example proposed by the StyleGAN2 paper. All methods can work with real as well as synthetic images. However, StyleRig seems to have the most problems working with real images, and all results shown in their paper are on synthetic images only.

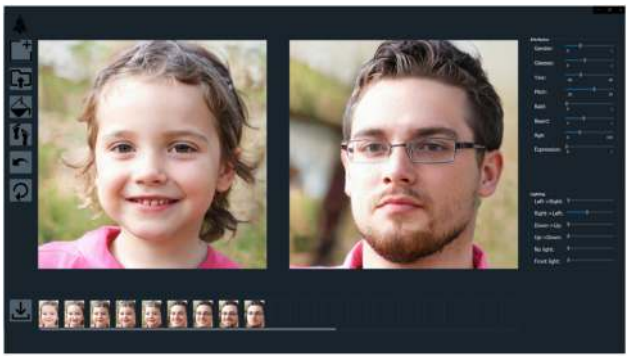

Fig. 18. StyleFlow user interface. Please refer to the supplementary video.

9 FAILURE CASES

There are multiple scenarios where StyleFlow may fail.

(1) The original FFHQ dataset does not have much variation in terms of extreme poses. Hence, some cases of extreme pose editing using StyleFlow may change the identity of the face (refer to Supplementary Material). For example, in the collected dataset yaw values vary from −52.3 to 58.1 degrees.
(2) Minor entanglements can be observed with beard removal and the female age progression using StyleFlow. There are cases where the age progression in females can end up in an old male in some extreme cases and the beard removal as an extreme edit can lead to female facial features (refer to Supplementary Material). We attribute these failures to the biases in the dataset due to irregular sampling. For example, the dataset has only 2.43% of people with beard attribute intensity greater than 0.5 or only 4.75% of women are above 45 years of age. Hence, the number of positive examples can be far less. To overcome this, we recommend training on larger balanced datasets with richer annotations. We leave this to future work.

10 CONCLUSION

We presented StyleFlow, a simple yet robust solution to the conditional exploration of the StyleGAN latent space. We investigated two important subproblems of attribute-conditioned sampling and attribute-controlled editing on StyleGAN using conditional continuous normalizing flows. As a result, we are able to sample high-quality images from the latent space given a set of attributes. Also, we demonstrate fine-grained disentangled edits along various attributes, e.g., camera pose, illumination variation, expression, skin tone, gender, and age for faces. The real face editing of our framework is of much higher quality compared to concurrent works. The qualitative and quantitative results show the superiority of the StyleFlow framework over other competing methods.

We identified three major limitations of our work. First, our work relies on the availability of attributes. These attributes might be difficult to obtain for new datasets and could require a manual labeling effort. Second, great results are only achievable with StyleGAN trained on high-quality datasets, mainly FFHQ. It would be good to have different types of datasets of similar quality, e.g., buildings or indoor scenes, to better evaluate our method. The lack of availability of very-high-quality data is still a major limitation for evaluating GAN research. Third, editing real images sometimes produces artifacts compared to editing synthesized images. While the quality of real image edits in our framework is still better than competing work, a better understanding of this problem and better projection algorithms require further research. We suggest this line of investigation as the most rewarding avenue of future work. In addition, it would be interesting to develop extensions to other attributes. In this context, it would be interesting to analyze what attributes are even captured by a GAN model. Maybe a combination of GANSpace to discover attributes and our method to encode conditional edits could be developed in the future.

REFERENCES

Rameen Abdal, Yipeng Qin, and Peter Wonka. 2019. Image2StyleGAN: How to embed images into the StyleGAN latent space? In *Proceedings of the IEEE International Conference on Computer Vision*. 4432–4441.

Rameen Abdal, Yipeng Qin, and Peter Wonka. 2020. Image2StyleGAN++: How to edit the embedded images? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'20)*.

Kfir Aberman, Mingyi Shi, Jing Liao, Dani Lischinski, Baoquan Chen, and Daniel Cohen-Or. 2019. Deep video-based performance cloning. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 219–233.

Andrew Brock, Jeff Donahue, and Karen Simonyan. 2018. Large Scale GAN Training for High Fidelity Natural Image Synthesis. arxiv:cs.LG/1809.11096 (2018).

Kaidi Cao, Jing Liao, and Lu Yuan. 2019. CariGANs. *ACM Trans. Graph.* 37, 6 (Jan 2019), 1–14. DOI: https://doi.org/10.1145/3272127.3275046

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. 2018. Neural ordinary differential equations. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 6571–6583.

Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. 2018. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. DOI: https://doi.org/10.1109/cvpr.2018.00916

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2019. StarGAN v2: Diverse Image Synthesis for Multiple Domains. arxiv:cs.CV/1912.01865 (2019).

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'09)*.

Ohad Fried, Ayush Tewari, Michael Zollhöfer, Adam Finkelstein, Eli Shechtman, Dan B. Goldman, Kyle Genova, Zeyu Jin, Christian Theobalt, and Maneesh Agrawala. 2019. Text-based editing of talking-head video. *ACM Trans. Graph.* 38, 4 (July 2019). DOI: https://doi.org/10.1145/3306346.3323028

Adam Geitgey. 2020. GitHub—Face Recognition. Retrieved from: https://github.com/ageitgey/face_recognition.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*. 2672–2680.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. 2018. FFJORD: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367* (2018).

Éric Guérin, Julie Digne, Eric Galin, Adrien Peytavie, Christian Wolf, Bedrich Benes, and Benoît Martinez. 2017. Interactive example-based terrain authoring with conditional generative adversarial networks. *ACM Trans. Graph.* 36, 6 (2017), 228.

Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian et al. 2019. The relightables: Volumetric performance capture of humans with realistic relighting. *ACM Trans. Graph.* 38, 6 (2019), 1–19.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. 2020. GANSpace: Discovering interpretable GAN controls. *arXiv preprint arXiv:2004.02546* (2020).

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. 2018. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.* 37, 6 (2018), 1–15.

IMPA-FACE3D. 2012. IMPA-FACE3D. Retrieved from: http://app.visgraf.impa.br/database/faces.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17)*. DOI: https://doi.org/10.1109/cvpr.2017.632

Wentao Jiang, Si Liu, Chen Gao, Jie Cao, Ran He, Jiashi Feng, and Shuicheng Yan. 2019. PSGAN: Pose and Expression Robust Spatial-Aware GAN for Customizable Makeup Transfer. arxiv:cs.CV/1909.06956 (2019).

Youngjoo Jo and Jongyoul Park. 2019. SC-FEGAN: Face editing generative adversarial network with user's sketch and color. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV'19)*.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2017. Progressive Growing of GANs for Improved Quality, Stability, and Variation. arxiv:cs.NE/1710.10196 (2017).

Tero Karras, Samuli Laine, and Timo Aila. 2019b. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'19)*. DOI: https://doi.org/10.1109/cvpr.2019.00453

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2019a. Analyzing and Improving the Image Quality of StyleGAN. arxiv:cs.CV/1912.04958 (2019).

Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Nießner, Patrick Pérez, Christian Richardt, Michael Zollhöfer, and Christian Theobalt. 2018. Deep video portraits. *ACM Trans. Graph.* 37, 4 (2018), 1–14.

Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. 2017. Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*. JMLR.org, 1857–1865.

Diederik P. Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. arxiv:stat.ML/1312.6114 (2013).

Ivan Kobyzev, Simon Prince, and Marcus Brubaker. 2020. Normalizing flows: An introduction and review of current methods. *IEEE Trans. Pattern Anal. Mach. Intell.* (2020). Retrieved from: https://arxiv.org/abs/1908.09257.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 2013. 3D object representations for fine-grained categorization. In *Proceedings of the 4th International IEEE Workshop on 3D Representation and Recognition (3DRR'13)*.

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. 2019. MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. arxiv:cs.CV/1907.11922 (2019).

Ricardo Martin-Brualla, Rohit Pandey, Shuoran Yang, Pavel Pidlypenskyi, Jonathan Taylor, Julien Valentin, Sameh Khamis, Philip Davidson, Anastasia Tkach, Peter Lincoln et al. 2018. LookinGood: Enhancing performance capture with real-time neural re-rendering. *arXiv preprint arXiv:1811.05029* (2018).

Microsoft. 2020. Azure Face. Retrieved from: https://azure.microsoft.com/en-in/services/cognitive-services/face/.

Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *ArXiv* abs/1411.1784 (2014).

Weili Nie, Tero Karras, Animesh Garg, Shoubhik Debnath, Anjul Patney, Ankit B. Patel, and Anima Anandkumar. 2020. Semi-Supervised StyleGAN for Disentanglement Learning. arxiv:cs.CV/2003.03461 (2020).

Yotam Nitzan, Amit Bermano, Yangyan Li, and Daniel Cohen-Or. 2020. Disentangling in Latent Space by Harnessing a Pretrained Generator. arxiv:cs.CV/2005.07728 (2020).

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Lev Semenovich Pontryagin. 2018. *Mathematical Theory of Optimal Processes*. Routledge.

Tiziano Portenier, Qiyang Hu, Attila Szabó, Siavash Arjomand Bigdeli, Paolo Favaro, and Matthias Zwicker. 2018. FaceShop: Deep sketch-based face image editing. *ACM Trans. Graph.* 37, 4 (July 2018). DOI: https://doi.org/10.1145/3197517.3201393

Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR'16)*.

Danilo Jimenez Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770* (2015).

Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. 2020. Encoding in style: A StyleGAN encoder for image-to-image translation. *arXiv preprint arXiv:2008.00951* (2020).

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. 2019. Interpreting the Latent Space of GANs for Semantic Face Editing. arxiv:cs.CV/1907.10786 (2019).

Aliaksandra Shysheya, Egor Zakharov, Kara-Ali Aliev, Renat Bashirov, Egor Burkov, Karim Iskakov, Aleksei Ivakhnenko, Yury Malkov, Igor Pasechnik, Dmitry Ulyanov et al. 2019. Textured neural avatars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2387–2397.

Aliaksandr Siarohin, Stéphane Lathuilière,, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. 2020. First Order Motion Model for Image Animation. arxiv:cs.CV/2003.00196 (2020).

Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. 2019. DeepVoxels: Learning persistent 3D feature embeddings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2437–2446.

Spectrico. 2020. Spectrico. Retrieved from: http://spectrico.com/.

Zhentao Tan, Menglei Chai, Dongdong Chen, Jing Liao, Qi Chu, Lu Yuan, Sergey Tulyakov, and Nenghai Yu. 2020. MichiGAN: Multi-input-conditioned hair image generation for portrait editing. *ACM Trans. Graph.* 38, 4 (July 2020).

Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. 2020a. StyleRig: Rigging StyleGAN for 3D control over portrait images. *arXiv preprint arXiv:2004.00121* (2020).

A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. Nießner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B Goldman, and M. Zollhöfer. 2020b. State of the art on neural rendering. *Comput. Graph. Forum (EG STAR 2020)* (2020). Retrieved from: https://arxiv.org/abs/2004.03805.

Justus Thies, Michael Zollhöfer, and Matthias Nießner. 2019. Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graph.* 38, 4 (2019), 1–12.

Justus Thies, Michael Zollhöfer, Christian Theobalt, Marc Stamminger, and Matthias Nießner. 2020. Image-guided neural object rendering. In *Proceedings of the International Conference on Learning Representations*. Retrieved from: https://openreview.net/forum.

Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. 2016. Conditional Image Generation with PixelCNN Decoders. arxiv:cs.CV/1606.05328 (2016).

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. High-resolution image synthesis and semantic manipulation with conditional GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Taihong Xiao, Jiapeng Hong, and Jinwen Ma. 2018. ELEGANT: Exchanging latent encodings with GAN for transferring multiple face attributes. In *Proceedings of the European Conference on Computer Vision (ECCV'18)*. 172–187.

Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. 2018. Deep image-based relighting from optimal sparse samples. *ACM Trans. Graph.* 37, 4 (2018), 1–13.

Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. 2019. PointFlow: 3D Point Cloud Generation with Continuous Normalizing Flows. Retrieved from: https://arxiv.org/abs/1906.12320.

Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. 2017. DualGAN: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE International Conference on Computer Vision*. 2849–2857.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. 2015. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365* (2015).

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. 2018. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589* (2018).

Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. 2019. Few-shot adversarial learning of realistic neural talking head models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV'19)*. DOI: https://doi.org/10.1109/iccv.2019.00955

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Hao Zhou, Sunil Hadap, Kalyan Sunkavalli, and David W. Jacobs. 2019. Deep single portrait image relighting. In *Proceedings of the International Conference on Computer Vision (ICCV'19)*.

Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. 2020a. In-Domain GAN Inversion for Real Image Editing. arxiv:cs.CV/2004.00049 (2020).

Jiapeng Zhu, Deli Zhao, Bo Zhang, and Bolei Zhou. 2020b. Disentangled Inference for GANs with Latently Invertible Autoencoder. arxiv:cs.LG/1906.08090 (2020).

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. 2016. Generative Visual Manipulation on the Natural Image Manifold. arxiv:cs.CV/1609.03552 (2016).

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017a. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 2223–2232.

Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A. Efros, Oliver Wang, and Eli Shechtman. 2017b. Toward multimodal image-to-image translation. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*.

Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. 2019. SEAN: Image synthesis with semantic region-adaptive normalization. *arXiv preprint arXiv:1911.12861* (2019).

Received September 2020; revised December 2020; accepted January 2021