Pacific Graphics 2020
E. Eisemann, A. Jacobson, and F.-L Zhang
(Guest Editors)

*Volume 39 (2020), Number 7*

# Simultaneous Multi-Attribute Image-to-Image Translation Using Parallel Latent Transform Networks

Sen-Zhe Xu[1] and Yu-Kun Lai[2]

[1]BNRist, Department of Computer Science and Technology, Tsinghua University, China
[2]School of Computer Science and Informatics, Cardiff University, UK

**Abstract**

*Image-to-image translation has been widely studied. Since real-world images can often be described by multiple attributes, it is useful to manipulate them at the same time. However, most methods focus on transforming between two domains, and when they chain multiple single attribute transform networks together, the results are affected by the order of chaining, and the performance drops with the out-of-domain issue for intermediate results. Existing multi-domain transfer methods mostly manipulate multiple attributes by adding a list of attribute labels to the network feature, but they also suffer from interference of different attributes, and perform worse when multiple attributes are manipulated. We propose a novel approach to multi-attribute image-to-image translation using several parallel latent transform networks, where multiple attributes are manipulated in parallel and simultaneously, which eliminates both issues. To avoid the interference of different attributes, we introduce a novel soft independence constraint for the changes caused by different attributes. Extensive experiments show that our method outperforms state-of-the-art methods.*

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

## 1. Introduction

Image-to-image translation is a widely studied problem in computer vision. It transfers images from a source domain to a target domain, while keeping the content (other than those related to domain differences) unchanged. With the development of generative adversarial networks (GANs) [GPAM*14, MO14, RMC15, MLX*17, ACB17] in recent years, image-to-image translation has achieved impressive results in many applications. Some of them help remove image defects and improve visual quality, such as denoising [CCCY18, KBJ19, WW19], colorization [CZZY17, NNE18], harmonization [TSL*17, WZZH19], super-resolution [LTH*17, WYW*18, LGLS19, LAD*19], and inpainting [YCYL*17, YLY*18], while others are dedicated to image attribute transformation, including style transfer [IZZE17, ZPIE17], facial attribute transfer [LZZ16, CCK*18], etc.

Although existing methods achieve promising results in image attribute transfer, most of them can only manipulate one attribute at a time. For example, existing works can well achieve "smiling" and "male" attributes for a facial image respectively, but most methods cannot simultaneously manipulate "smiling" and "male" attributes on one image, as illustrated in Fig. 1. A naive solution is to chain multiple translation tasks one after another, and use the output of the previous task as the input of the next task. But there are at least two problems with this approach:

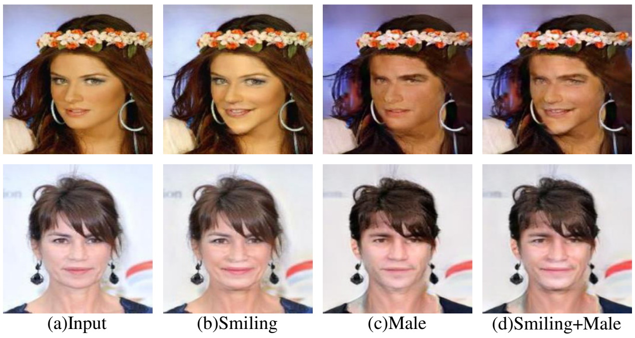

**Figure 1:** *Illustration of the simultaneous multi-attribute translation problem. (a) is the input, (b) and (c) are our single attribute translation results for "Smiling" and "Male" respectively, and (d) is our simultaneous translation of both attributes.*

1. **Order affects results.** For example, the result of making a facial image "aged" and then "smiling" is usually different from the result of making it "smiling" and then "aged".
2. **Out-of-domain issue.** The output of the first task does not necessarily fall strictly within the input domain of the next task.

Therefore it will cause the image quality to degrade after going through the consecutive networks.

The fundamental cause of these two problems is, when the network edits a certain attribute, the modification is distributed throughout the whole latent space. Therefore, when manipulating multiple attributes in succession, these latent changes at different stages will interfere and affect each other.

In recent years, methods like StarGAN [CCK*18], STGAN [LDX*19] have been put forward to learn transforms of single or multiple attributes. They map between multiple domains with one model by adding a list of attribute (0, 1) labels to the network feature. But these methods mix all attributes together, ignoring the inherent differences of attributes. Since the difficulties of manipulating different attributes vary, this strategy makes the learning of different attribute transforms uneven. When applying multiple attributes at the same time, these methods often lead to interference between attributes, which degrades the quality of the resulting images.

Disentangled representations have also been used to solve this problem. The idea is to restrict the representation of a certain attribute to be in a fixed area of the latent vector, such as some specific channels, thereby the coupling of different attributes in the latent vector is eliminated, and exact editing of different attributes can be achieved at the latent vector level.

However, disentangled learning also has its disadvantages. Firstly, disentangling is at the expense of learning efficiency. Since every attribute is different in complexity, the latent space requirement of the representation for each attribute should also be different. Disentangled learning restricts the representation of any attribute to be in some artificially specified latent size, which will cause a waste or deficiency of the corresponding latent space in representation ability, thereby increase the training burden. Secondly, when applying a disentangled representation to attribute transfer, a key idea is to exchange some channels of the latent vector, so such methods require a reference image that carries desired channels for swapping. Different choices of reference images would lead to different results, so they cannot automatically transfer the image's domain using a single input image.

We take a different approach to tackle this problem. To address the dependencies on ordering, we achieve multi-attribute image-to-image translation *simultaneously* using parallel latent transform networks, one for each attribute. We further introduce a soft independence constraint loss term to ensure that different attributes do not interfere with each other in the latent space, while not affecting the attribute transform learning. Unlike disentangled learning, our method does not need to restrict the representation of an attribute to be in a fixed area of the latent vector.

Specifically, we firstly propose a novel unsupervised image-to-image translation framework, which consists of an Encoder, a Decoder, and multiple parallel Latent Transform Networks (LTNs) in the middle. The framework does not require dual training like CycleGAN, and this facilitates our multi-attribute translation learning. We then constrain all the attribute changes to only occur in the middle LTNs, with each LTN corresponding to one attribute conversion. Finally the increment of the latent vector of each LTN is constrained to be decoupled from each other, so that multiple attributes can be modified at the same time independently by increasing or decreasing the latent increments.

## 2. Related Work

### 2.1. Image-to-image Translation with GANs

Recently image-to-image translation has shown rapid development with the adoption of convolutional neural networks (CNNs). Pix2pix [IZZE17] has shown impressive results on paired image translation learning by applying a conditional GAN (cGAN) [MO14] to learn a conditional generative model. Pix2pixHD [WLZ*18] makes one step further to generate High-Definition (HD) images by using a coarse-to-fine generator and multi-scale discriminators. As the paired data is often scarce, many unpaired image-to-image translation frameworks [ZPIE17, YZTG17, KCK*17, LT16, LBK17] have been proposed. CycleGAN [ZPIE17] uses cycle consistency loss to constrain the learned mapping to be cycle-consistent, which helps to preserve image content. The concurrent work DualGAN [YZTG17] and DiscoGAN [KCK*17] use the same principle as CycleGAN. CoGAN [LT16] enforces the decoding layer of high-level features to share weights, which helps the framework to learn the joint distribution of images from marginal distributions. The follow-up work UNIT (Unsupervised Image-to-Image Translation Network) [LBK17] views this constraint as the shared-latent space assumption, and extends CoGAN for unsupervised image-to-image translation problems using a VAE (Variational Auto-Encoder)-GAN [KW13] framework. Some other works aim to improve translation quality from other aspects. Attention-GAN [MRT*18] introduces unsupervised attention mechanism to the generator and discriminator, and makes quality improvement for individual objects without altering the background. InstaGAN [MCS18] improves the shapes of multiple target instances by introducing a context-preserving loss.

However, these methods are all focused on mapping two opposing domains, which means they can only transform one attribute at a time. In recent years several methods are proposed to learn multiple attribute transforms by adding a list of attribute (0, 1) labels to the network feature. StarGAN [CCK*18] concatenates attribute (0,1) labels as extra channels to the input, to learn the mappings among multiple domains. Similar to StarGAN, STGAN [LDX*19] replaces such attribute labels with attribute status differences, and introduces skip connections in its networks. AttGAN [HZK*19] and FaderNet [LZU*17] both make the encoded latent code not contain attribute information, and use attribute labels to guide the decoded images to have the desired attributes. However, these methods mix all attributes together, ignoring the inherent differences of attributes. Since the difficulty of manipulating different attributes is different, the transform learning of different attributes tends to be uneven. And they also have no internal mechanism to avoid interference between mappings of different attributes. When manipulating multiple attributes, their performance is not as good as manipulating a single attribute.

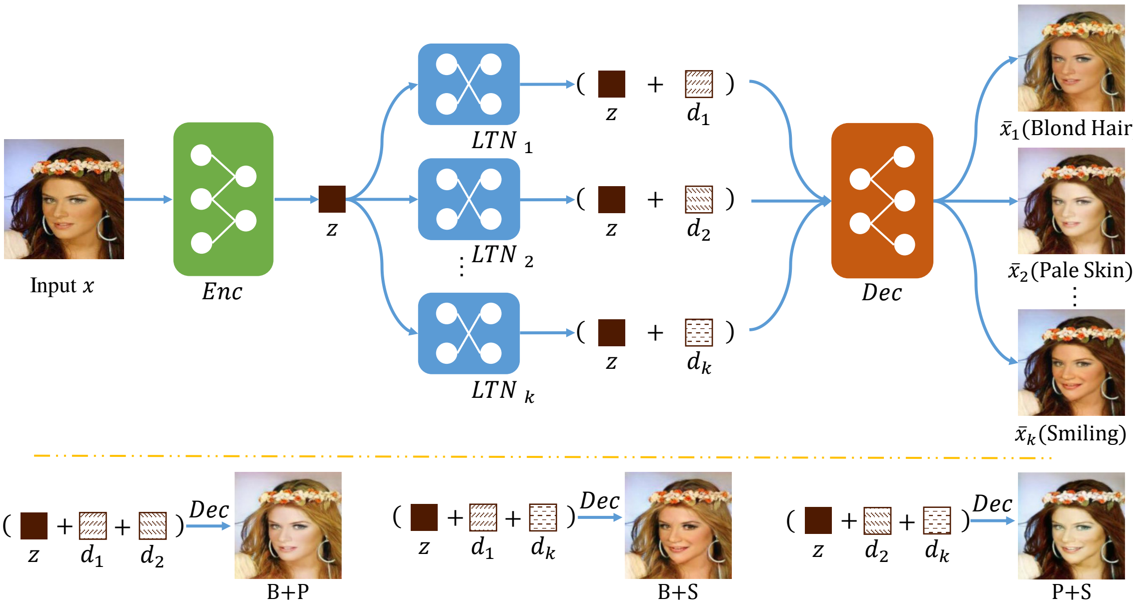

**Figure 2:** *The parallel framework of our simultaneous multi-attribute transform method. Enc and Dec are the encoder and decoder. They do not change the attributes of an image or latent vector. Latent transform networks (LTNs) are embedded in the middle of Enc and Dec in parallel, in charge of transforming specific attributes. Increments produced by LTNs do not interfere with each other, so they can be accumulated to manipulate multiple attributes at the same time.*

### 2.2. Disentangled Representation

Learning to disentangle the latent representation by specified factors of variation is a challenging problem. We discuss the current methods in two categories.

The first category is dedicated to distilling a single factor of variation from the representation, and is mainly used for generating images while keeping certain invariance. InfoGAN [CDH*16] learns to disentangle representations by providing the generator with an incompressible noise and a latent code, and maximizing the mutual information between the latent code and data variation. It can discover meaningful hidden representations in an unsupervised way, but the user cannot specify a fixed attribute to disentangle. Liu *et al*. [LWS*18] disentangle identity of facial images by breaking an autoencoder into the "identity distilling" and "identity dispelling" branches, and it can be used for face editing while keeping the identity unchanged. Lee *et al*. [LTH*18] divide the latent space to a content space and a domain-specific attribute space, which helps to keep the content unchanged when changing the image style. Kazemi *et al*. [KIN19] also learn to disentangle the representations of style from content of the data. Since these methods only focus on a certain aspect of the representation, they generally cannot be used to transfer multiple specified attributes simultaneously.

The other category of methods tries to disentangle multiple factors of specified attributes into the latent vector, and is mainly used for attribute transfer. DNA-GAN [XHM17] tries to disentangle different attributes in a supervised way, and each piece of the latent vector represents an attribute. ELEGANT [XHM18] is a similar work which also encodes all specified attributes in the latent space in a disentangled manner. Hu *et al*. [HSP*18] propose an unsupervised method to learn disentangled representations without exploiting any manual labeling or data domain knowledge. It is achieved by using a Mixing/Unmixing Autoencoder. Feng *et al*. [FWK*18] propose a semi-supervised disentangling method using both labeled and unlabeled data, which is achieved by using a dual swap mechanism. However, all of these methods pre-allocate different attributes to different parts of the latent vector, and in practice assign the same latent size for different attributes, ignoring the inherent differences of attributes. They also achieve attribute transfer by swapping/mixing corresponding latent code parts from an exemplar, so cannot achieve image transform with a single image as input.

## 3. Our Approach

We propose a novel unpaired image-to-image translation method. Our method only encodes the input image once to obtain its latent vector, and calculates its latent increment for each attribute translation. The latent increments are calculated by parallel internal latent transform sub-networks. The latent increments are constrained to be in their own spaces and do not affect each other. By adding different attribute transform increments to the original latent vector, multiple attributes of an image can be changed simultaneously with decoding only once.

### 3.1. Principle

Denote the input image as $x \in X$ where $X$ is a collection of 2D images. $A = \{a_1, a_2, \dots, a_k\}$ are $k$ attributes of $X$, which are meaningful inherent features to describe the images in $X$. For example, if $X$ is a collection of facial images, $A$ may include attributes such as

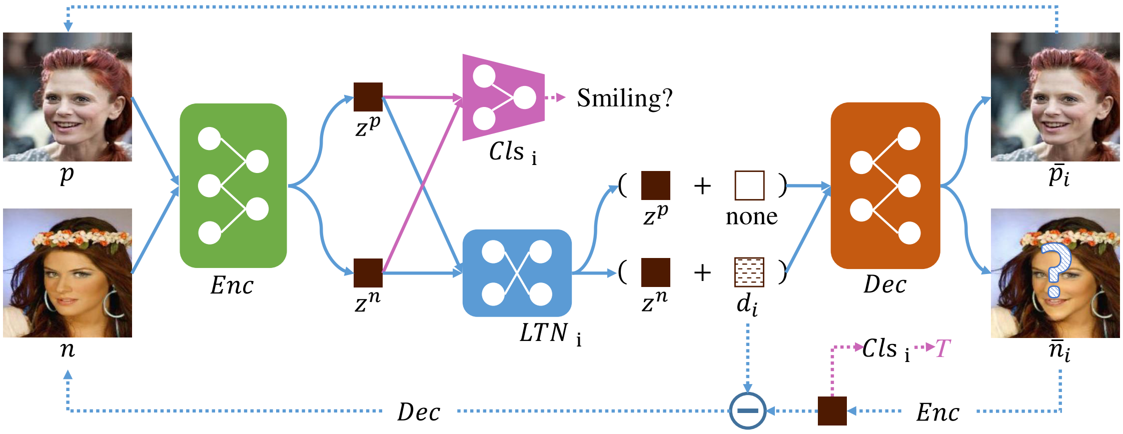

**Figure 3:** *The training flow of a single attribute. The figure takes attribute "*Smiling*" as an example. p and n are samples from the positive and negative domains for training. Their latent vectors $z^p$ and $z^n$ are used for the learning of the auxiliary classifier $Cls_i$. $LTN_i$ learns to transfer the latent vector to gain the target attribute, and its output is represented as the input vector plus an increment. Output $\overline{p}_i$ for p is supposed to keep appearance, while output $\overline{n}_i$ is constrained such that its re-encoded vector must get the positive classification, and be able to reconstruct the original image n.*

*Smiling, Male, Pale skin, Blond hair*, etc. The value of attribute $a_i$ of image $x$ is denoted as $v_i(x)$. We stipulate $v_i(x) \in \{0,1\}$, which represents whether $x$ has the attribute $a_i$. Therefore, the full attributes information of $x$ can be represented as a vector $V(x) = (v_1(x), v_2(x), \ldots, v_k(x))$. Suppose $x$ does not have several attributes in $A$, our goal is to generate a synthesized image $\overline{x}$ that gains multiple specified attributes at the same time.

As shown in Fig. 2, our framework consists of an Encoder $Enc$, a Decoder $Dec$, and $k$ parallel latent transform networks $LTN = \{LTN_i \mid 1 \leq i \leq k\}$, where $LTN_i \in LTN$ is responsible for the i-th attribute $a_i$. In our framework, we stipulate the transform process of all the attributes to only occur in $LTN$.

$Enc$ encodes the input image $x$ into a latent vector $z = Enc(x)$. The vector $z$ is generic for transforming all attributes. $Dec$ decodes any latent vector into an output image. Since there is mutual correspondence between the latent vector and the image, information of all the attributes in the image will be encoded in its latent vector. We denote the attributes contained in $z$ as $V(z)$. As neither $Enc$ nor $Dec$ modifies the attribute information of $x$, naturally $V(z) = V(x)$. In addition, decoding $z$ should also get the original input image $x$, i.e., $Dec(z) = x$.

Intuitively, the latent vector $z$ is a high-dimensional vector, and the attribute information is hidden in it, which can be hard to obtain by direct observation. On the other hand, the attribute value of the image $x$ can be easily judged by human eyes, e.g. a person can easily distinguish whether a facial image is smiling or not. Inspired by this, we train a set of classifier $Cls = \{Cls_i \mid 1 \leq i \leq k\}$ to classify the value of every attribute from a latent vector. Since $V(z) = V(x)$, $Cls$ exploits $V(x)$ as ground-truth to learn the classification from $z$.

The $LTN$ consists of multiple parallel networks with the same structure, each corresponding to an attribute to transform. When a latent vector without attribute $a_i$ is inputted, $LTN_i$ will convert it to get the attribute $a_i$. The latent vector $z$ obtained by $Enc$ passes through $LTN$ in parallel, and we denote the output latent vector of $LTN_i$ as $\overline{z}_i = LTN_i(z)$. $\overline{z}_i$ and $z$ have the same size, and it ensures that $v_i(\overline{z}_i) \equiv 1$, where $v_i(\cdot)$ is the $i$-th attribute value. $Dec$ decodes $\overline{z}_i$ to obtain a synthesized image $\overline{x}_i = Dec(\overline{z}_i)$, and similarly $v_i(\overline{x}_i) \equiv 1$. The content other than the domain difference of $\overline{x}_i$ should keep the same as $x$. If the input image $x$ itself does not have the attribute $a_i$, i.e., $v_i(z) = v_i(x) = 0$, then we are concerned about the increment of $LTN_i$ to $z$: $d_i = \overline{z}_i - z$. $d_i$ represents the modification that needs to be applied to $z$ in order to gain the attribute $a_i$. Similarly, for the attribute $a_j$, it is also possible to obtain such an increment $d_j$ by $LTN_j$.

Focusing on the increment rather than absolute values has clear advantages. We only need to ensure that for any $i, j \in \{1,2,\ldots,k\}$, $i \neq j$, if $d_j$ does not affect the expression of $d_i$, then we can make the image obtain two attributes by decoding $z + d_i + d_j$. In this way, the input $x$ only needs to be encoded and decoded once, and then the user can manipulate multiple attributes by adding different increments to its latent vector. Instead of chaining multiple tasks into a sequence to manipulate multiple attributes, our simultaneous multi-attribute transfer approach avoids "the order affects results" problem and the out-of-domain issue.

In the training phase, we take care about one attribute at a time, and all the attributes are trained in a round robin manner. In the following we take the $i$-th attribute $a_i$ as an example, as shown in Fig. 3, and other attributes are trained in the same way. The attribute $a_i \in A$ will divide $X$ into two domains, namely the positive domain $P_i = \{x \mid x \in X, v_i(x) = 1\}$ with attribute $a_i$ and the negative domain $N_i = \{x \mid x \in X, v_i(x) = 0\}$ without attribute $a_i$. Images from both $p \in P_i$ and $n \in N_i$ are taken to pass the network individually for training. For concise description, we define the latent vector of $p$ and $n$ as $z^p = Enc(p)$, $z^n = Enc(n)$, the outputs for both types of inputs are $\overline{p}_i = Dec(LTN_i(z^p))$, $\overline{n}_i = Dec(LTN_i(z^n))$. In the following subsections, we introduce three types of loss terms for training, based on encoder/decoder properties, learning of attribute transform and independence of increments, respectively.

### 3.2. Encoder and Decoder Properties

We first ensure several properties of *Enc* and *Dec*.

**Invariance of encoding and decoding.** Firstly, for both the input images $p$ and $n$, the original image should be obtained by directly decoding its latent vector, regardless of whether the image has the attribute $a_i$ or not. This makes sure that without the effect of $LTN_i$ any latent vector will definitely be decoded into the original image, and guarantees that attribute transform only occurs in the middle components $LTN_i$.

$$\mathcal{L}_{InvL1}(Enc,Dec) = \mathbb{E}_{p,n}\left[\|Dec(z^p)-p\|_1 + \|Dec(z^n)-n\|_1\right], \tag{1}$$

$$\mathcal{L}_{InvVGG}(Enc,Dec) = \mathbb{E}_{p,n}\left[\|\varphi(Dec(z^p))-\varphi(p)\|_1 + \|\varphi(Dec(z^n))-\varphi(n)\|_1\right], \tag{2}$$

where $\varphi$ denotes the features extracted by a pretrained VGG19 [SZ15] network. The two losses ensure the content is unchanged in terms of both $\ell_1$ and VGG-based perceptual losses. Following existing work, using the $\ell_1$ norm in loss terms related to image appearance tends to generate images with better details.

**Increment stability.** Modifying the latent vector will affect the decoding result. Ideally, we expect such sensitivity to be restricted to increments generated by $LTN$, which adds attributes to the input, but insensitive to other interference increments not generated by $LTN$. To achieve this, we add Gaussian noise $\varepsilon$ to the latent vector, and design the following losses (both in $\ell_1$ and VGG spaces) that penalize changes of the decoding results:

$$\mathcal{L}_{StbL1}(Enc,Dec) = \mathbb{E}_{p,n}\left[\|Dec(z^p+\varepsilon)-p\|_1 + \|Dec(z^n+\varepsilon)-n\|_1\right], \tag{3}$$

$$\begin{aligned}\mathcal{L}_{StbVGG}(Enc,Dec) = \mathbb{E}_{p,n}\big[&\|\varphi(Dec(z^p+\varepsilon))-\varphi(p)\|_1 \\ &+\|\varphi(Dec(z^n+\varepsilon))-\varphi(n)\|_1\big],\end{aligned} \tag{4}$$

**Latent consistency.** For $n$, the output $\overline{n}_i$ is a fake image with attribute $a_i$, which aims to be in the same distribution as $P_i$. Thus it can be re-encoded by *Enc*, and the resulting latent vector needs to be consistent with the vector before decoding, so as to ensure one-to-one correspondence between latent vectors and images:

$$\mathcal{L}_{LC}(Enc,Dec) = \mathbb{E}_n\left[\|Enc(\overline{n}_i) - LTN_i(z^n)\|_1\right], \tag{5}$$

**Reconstruction.** The re-encoded vector of $\overline{n}_i$, minus the previous increment generated by $LTN_i$, is supposed to change back to a vector without attribute $a_i$. Further, its decoding is expected to be the original image $n$:

$$\mathcal{L}_{Rec}(Enc,Dec) = \mathbb{E}_n\left[\|Dec(Enc(\overline{n}_i) - (LTN_i(z^n)-z^n)) - n\|_1\right], \tag{6}$$

This loss simply guarantees the content consistency of the output and input, eliminating the need of the cycle consistency by an inverse task like CycleGAN to ensure content consistency.

**Total loss for *Enc* and *Dec*.** The total loss for *Enc* and *Dec* is written as:

$$\begin{aligned}\mathcal{L}_G(Enc,Dec) = {}&\lambda_{L1}(\mathcal{L}_{InvL1}+\mathcal{L}_{StbL1}) \\ &+\lambda_{VGG}(\mathcal{L}_{InvVGG}+\mathcal{L}_{StbVGG}) \\ &+\lambda_{LC}\mathcal{L}_{LC}+\lambda_{Rec}\mathcal{L}_{Rec}.\end{aligned} \tag{7}$$

We use $\lambda_{L1} = 2.5$, $\lambda_{VGG} = 5.0$, $\lambda_{LC} = 10.0$, $\lambda_{Rec} = 5.0$ as the hyper-parameters to control the relative importance of the losses.

### 3.3. Learning for Attribute Transforms

The losses defined so far do not show how to make $LTN_i$ to learn the attribute transform yet, which we will now address.

**Domain classification loss.** In order to learn attribute transform, we propose a classification loss. We first use $Cls_i$ to learn attribute classification for latent vectors encoded by *Enc*:

$$\mathcal{L}_{Cls}(Cls_i) = \mathbb{E}_{p,n}\left[-\log(Cls_i(z^p)) - \log(1-Cls_i(z^n))\right], \tag{8}$$

Since the ground-truth of the classification task, i.e. attribute values of $p$ and $n$ are known, the classifier $Cls_i$ is well-defined. So we use this to direct the training for attribute transform of $LTN_i$. Intuitively, if the attribute transform by $LTN_i$ is successful, $Cls_i$ should classify the re-encoded vector of $\overline{n}_i$ into the $P_i$ class:

$$\mathcal{L}_C(LTN_i) = \mathbb{E}_n\left[-\log(Cls_i(Enc(\overline{n}_i)))\right], \tag{9}$$

**Adversarial loss.** In addition to the domain classification, we also include $k$ adversarial networks $D = \{D_1, D_2, \dots, D_k\}$ to discriminate genuine and fake images for each attribute. $D_i \in D$ corresponds to attribute $a_i$. The adversarial loss further ensures that $\overline{n}_i$ is the same as $P_i$ and increases the realism of $\overline{n}_i$. Here, we use LSGAN [MLX*17] and PatchGAN [IZZE17] as $D$ for stable training. The adversarial loss is as follows:

$$\mathcal{L}_{adv}(LTN_i,D_i) = \mathbb{E}_{p,n}\left[\|D_i(p)\|_2^2 + \|1-D_i(\overline{n}_i)\|_2^2\right], \tag{10}$$

which is solved by $\arg\min_{LTN_i}\max_{D_i}\mathcal{L}_{adv}(LTN_i,D_i)$. The latent transform network $LTN_i$ tries to minimize this objective, while the discriminator $D_i$ tries to maximize it.

**Total loss for attribute transform.** So the total loss for transform is written as:

$$\mathcal{L}_T(Cls,LTN_i,D) = \mathcal{L}_{Cls} + \mathcal{L}_C + \mathcal{L}_{adv}. \tag{11}$$

The weights of these losses are all 1.0 in our experiment, so we omit them in the formula.

### 3.4. Independence of Increments

The above losses are only competent for learning the transform of attribute $a_i$. Since the increments $d_i = LTN_i(z^n) - z^n$ and $d_j = LTN_j(z^n) - z^n$ caused by $LTN_i$ and $LTN_j$ are distributed in the same latent space, if $d_j$ has an impact on the expression of $d_i$, then we cannot manipulate both attributes by simultaneously adding increments $d_i$ and $d_j$.

To eliminate this correlation, we propose a soft independence constraint for $d_i$ and $d_j$. Intuitively, if $LTN_i$ is good enough and $d_i$ and $d_j$ are independent, the increment by $LTN_j$ to a latent vector should not be influenced by the effect of $LTN_i$. So we expect the increment $d_j$ obtained by applying $LTN_j$ to the latent vector $z^n$, and $d'_j$, obtained by applying $LTN_j$ to the resulting vector of $LTN_i(z^n)$ to be equal, which is formulated as:

$$\mathcal{L}_{ind}(LTN_i) = \frac{1}{k-1}\sum_{j\neq i}\mathbb{E}_n\left[\|(LTN_j(LTN_i(z^n)) - LTN_i(z^n)) - (LTN_j(z^n)-z^n)\|_1\right]. \tag{12}$$

For $LTN_i$, it needs to avoid interference with the remaining $(k-1)$ LTNs in the latent space. Therefore, we normalize the loss term with $\frac{1}{k-1}$ so that its magnitude does not vary with the number of attributes $k$. We also tried reducing the cosine similarities of the increments $d_i$ and $d_j$ instead to ensure independence, but found that it will break the balance of learning transforms and increment independence and is hard to train. Our soft independence constraint not only enhances the independence of the increments, but also does not affect the transform learning.

### 3.5. Full Objective

Finally, our full objective is:

$$\mathcal{L}_{Total} = \mathcal{L}_G(Enc, Dec) + \mathcal{L}_T(Cls, LTN_i, D) + \mathcal{L}_{ind}(LTN_i). \quad (13)$$

where $i$ refers to the attribute index, which is selected from 1 to $k$ in turn for each iteration of the training phase. The coefficients of these objectives are all 1.0 in our experiment.

## 4. Experiments

### 4.1. Implementation Details

Our *Enc* is composed of two convolution layers with stride 2 followed by two residual blocks [HZRS16]. *Dec* is symmetric with *Enc*, which has two residual blocks followed by two transposed convolution layers. Each $LTN_i$ consists of five residual blocks. All auxiliary classifiers $Cls_i \in Cls$ share a feature extractor with four convolution layers, and each $Cls_i$ has a feature classifier with three fully connected layers. The activation function is ReLU. We adopt the PatchGAN discriminator [IZZE17] with a receptive field of $70 \times 70$ as our $D_i$, which is conducive to image detail quality. Inputs are resized to a fixed size of $256 \times 256$ before inputting, due to the existence of fully connected layers in *Cls*. Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ is used for training. The learning rate is set to $2 \times 10^{-4}$. Our framework is implemented with Jittor [HLY*20], a recently proposed novel deep learning framework, which is proven to run faster than PyTorch. Our hardware environment is a PC with an Intel(R) Core(TM) i7-6850K CPU 3.60GHz and an Nvidia GTX1080Ti GPU.

### 4.2. Test on Facial Images

Facial images are not only the most common type of images, but also have a large number of inherent attributes. We firstly evaluate our method on facial images, and use CelebFaces Attributes (CelebA) dataset [LLWT15] for training.

CelebA consists of 202,599 aligned facial images, and each image is labeled with up to 40 attributes. We randomly select 2000 images for testing, and use the remaining images for training.

**Quantitative and qualitative comparisons.** We compare our method with several state-of-the-art multi-attribute transform methods, namely FaderNet [LZU*17], AttGAN [HZK*19], StarGAN [CCK*18] and STGAN [LDX*19]. Methods that need an exemplar as input [XHM17, XHM18, CUYH20] are not included in comparison. Since their inputs are much different from our method, it is not possible to make a meaningful comparison with them. In this experiment, we consider manipulating 5 attributes at the same time, including *Blond Hair*, *Pale Skin*, *Smile*, *Male* and *Young*. We select these attributes because they are representative since they cover the changes from overall to detail, and are most selected by relevant works. The qualitative results are shown in Fig. 4. It can be observed that our method has higher quality results for multi-attribute manipulation. More importantly, it can be seen that our results of manipulating multiple attributes are more consistent with the results of manipulating a single component attribute. We can find that to all the methods, the problem becomes more challenging when manipulating more attributes at the same time. Many other works only show at most three attributes to transform simultaneously. Here we take the challenge to manipulate all these five attributes at the same time. It can be seen that although the quality of our results are also declined, it is much better than the comparative methods.

To quantitatively measure the attribute generation accuracy, we use a well-trained attribute classifier for each attribute. To conduct this, we firstly train an additional deep attribute classifier as a referee that performs binary classification of whether the attribute is present or not, whose average classification accuracy is higher than 90%, and then apply this classifier to the results to compute their classification accuracy. Table 1 shows the result. As can be seen from Table 1, our method has a relatively high generation accuracy. Although our method is inferior to the comparative methods in some attributes, our method achieves a better balance among all the attributes.

We then show more results to evaluate the quality of the results. In order to better show the efficiency of our method in more detail, we apply two attributes to input images each time. Specifically, we select three attributes *Blond Hair*, *Pale Skin*, *Smile* (which are abbreviated as $B$, $P$ and $S$, respectively) and test the results of all their combinations. Besides the multi-attribute transform methods, we also include a classic single attribute image-to-image translation method CycleGAN [ZPIE17] to show its results with two different orders.

We use Fréchet Inception Distance (FID) [HRU*17] with images in the target domain as the quantitative measure. That is when we transform *Blond Hair* and *Pale Skin*, we use the image set with both *Blond Hair* and *Pale Skin* as the target set to calculate FID scores, so the FID scores not only reflect the image quality, but also somehow reflect the classification accuracy. Smaller FID values are better. As shown in Table 4, our method produces consistently better results than the comparison methods. The independence constraint in our method also contributes to significant improvements of results. Visual comparisons of different results are presented in Figs. 5-7. The results of the two different orders of CycleGAN are slightly different, suggesting that the results are not self-consistent when a single-attribute transform method is used to transform multiple attributes. We will evaluate this "order affects results" issue later. Other comparison methods and our method without the independence constraint tend to create blurred results and results with artifacts, affected by the interference of multiple attributes. In contrast, our results are plausible and do not have such artifacts.

**Evaluation of the increment independence.** We evaluate the increment independence in our framework, to show the effectiveness of our increment independence constraint. Each time we select two different LTNs to create their respective latent increments of the same input, and we use the inner product of the increments $d_i \cdot d_j$ where $d_i$ and $d_j$ are two increments for attributes $i$ and $j$ to measure the independence. A smaller inner product means less interference. The mean score of the test set is taken as the result for each group of attributes, as shown in Table 2.

By using $\mathcal{L}_{ind}$, the inner product is reduced to about one tenth,

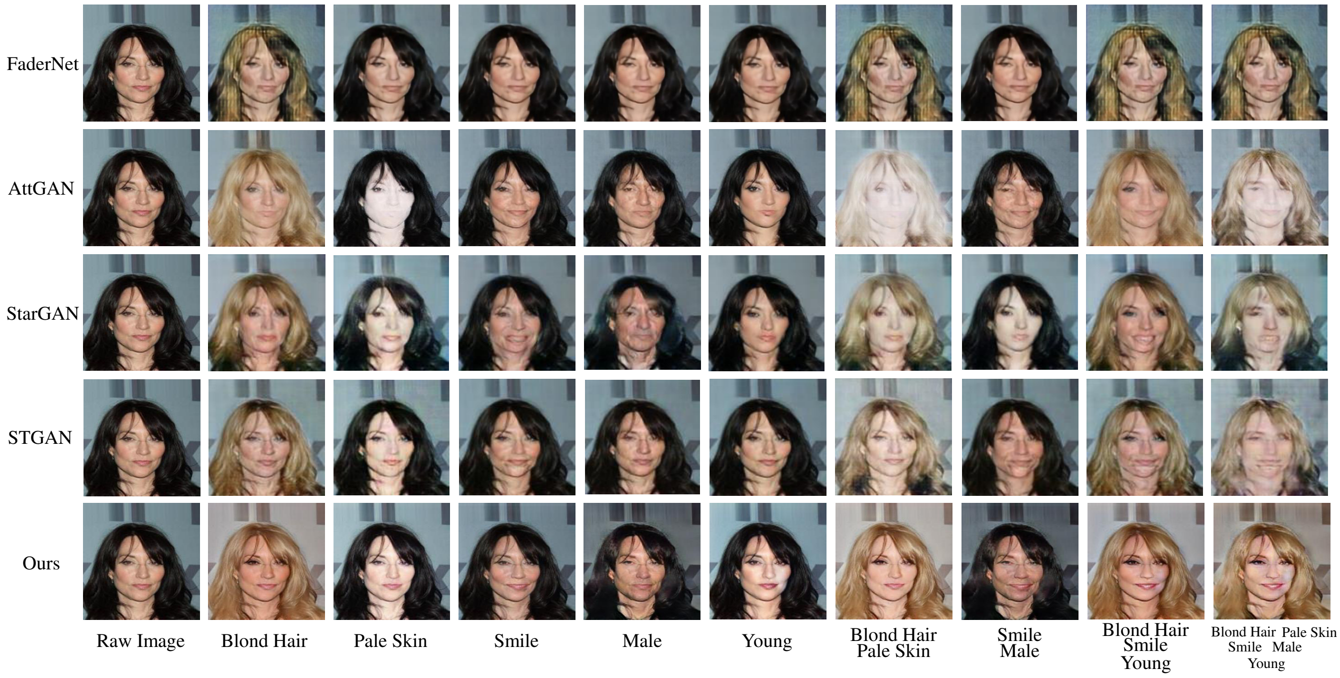

**Figure 4:** *Facial attribute editing results on the CelebA dataset.*

**Table 1:** *Attribute generation accuracy.*

| | *Blond Hair* | *Pale Skin* | *Smile* | *Male* | *Young* |
|---|---|---|---|---|---|
| FaderNet | 75.885% | 12.123% | 40.991% | 10.887% | 74.233% |
| AttGAN | 88.351% | 92.488% | 82.664% | 91.617% | 88.160% |
| StarGAN | 91.072% | 15.406% | 77.627% | 94.937% | 89.387% |
| STGAN | 96.897% | 90.359% | 97.768% | 70.005% | 93.865% |
| Ours | 95.393% | 93.601% | 92.200% | 92.650% | 94.479% |

**Table 2:** *Inner products of increments of different attributes for the same input.*

| | *B&P* | *B&S* | *P&S* |
|---|---|---|---|
| Ours *w/o* $\mathcal{L}_{ind}$ | $1.120 \times 10^5$ | $5.857 \times 10^4$ | $3.552 \times 10^5$ |
| Ours | $1.601 \times 10^4$ | $8.168 \times 10^3$ | $1.184 \times 10^4$ |

compared with not using this loss, which shows $\mathcal{L}_{ind}$ is effective to ensure the independence between increments and reduces interference. Although our method does not restrict attributes in fixed parts of latent code, we achieve a similar effect as the disentangled representation does.

Note that here the inner product is used as a metric but *not* as an objective, because according to our experiments it is too strong a constraint and will stop the network from learning transforms. Our soft independence constraint and transform learning complement each other.

**Evaluation of the effect of order on results.** When changing two (or more) attributes together, most methods such as CycleGAN will need to apply these attribute manipulations one by one. Ideally, the final results given the same input and same attribute changes should be the same. In practice, however, the order of applying these changes affects the results. Here we use four measurements to evaluate the mean difference of outputs for three pairs of attributes, and the results are shown in Table 3. Mean Squared Error (MSE) is the pixel-by-pixel average squared difference between the two images. Structural Similarity (SSIM) index [WBSS04] uses luminance, contrast and structure to measure image similarity, and its

**Table 3:** *Quantitative evaluation of the "order affects results" effect. The table shows four measurements for the mean differences between results obtained by two inference sequences using CycleGAN.*

| *inference order 1* | MSE | SSIM | PSNR | LPIPS | *inference order 2* |
|---|---|---|---|---|---|
| CycleGAN(*B+P*) | 244.65 | 0.7472 | 25.79 | 0.0254 | CycleGAN(*P+B*) |
| CycleGAN(*B+S*) | 85.72 | 0.8059 | 29.87 | 0.0165 | CycleGAN(*S+B*) |
| CycleGAN(*P+S*) | 87.85 | 0.7924 | 29.46 | 0.0197 | CycleGAN(*S+P*) |

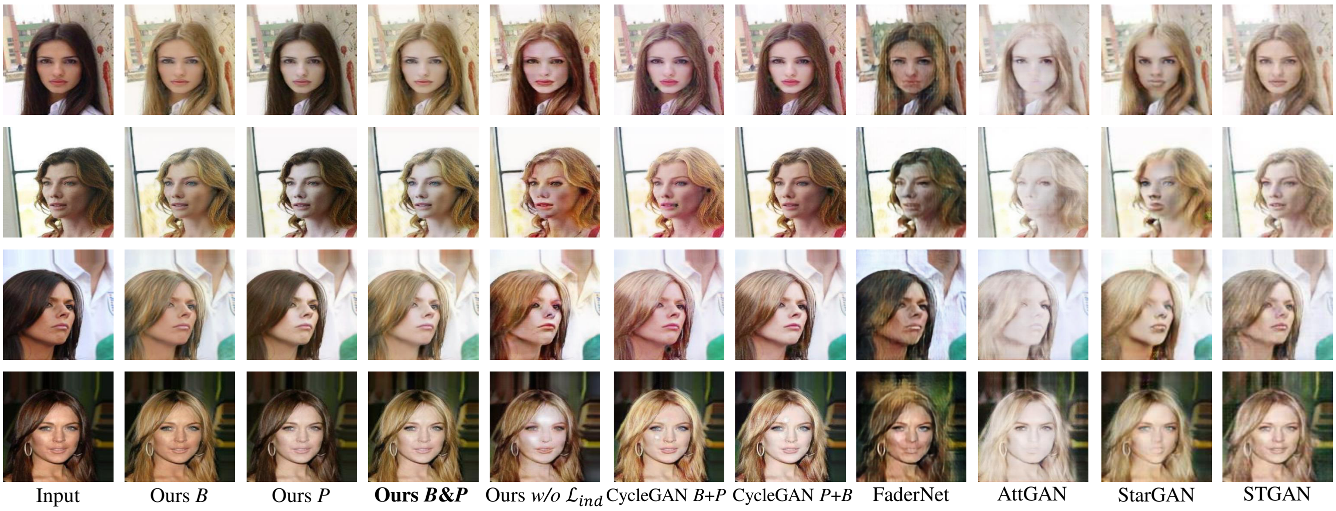

**Figure 5:** *Qualitative comparisons of* Blond hair *and* Pale skin *transform.*

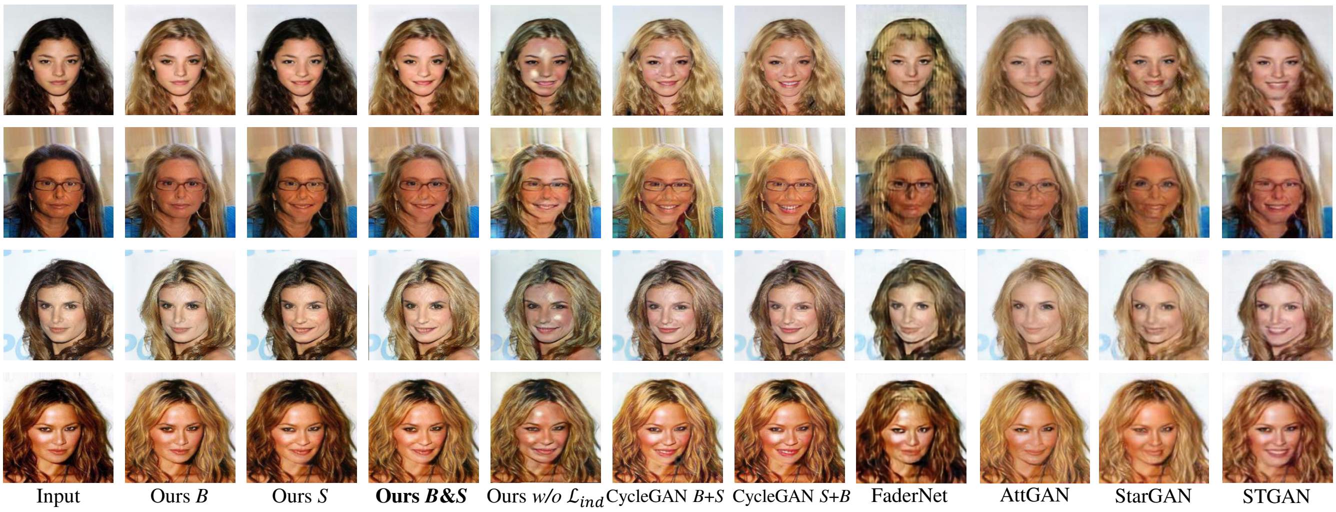

**Figure 6:** *Qualitative comparisons of* Blond hair *and* Smiling *transform.*

value extends between $[-1,1]$ and only equals 1 if the two images are identical. Peak signal-to-noise ratio (PSNR) [HZ10] is a common measurement for image quality loss, which is calculated logarithmically via MSE, and the higher the PSNR, the more similar the images are. Recently proposed Learned Perceptual Image Patch Similarity (LPIPS) [ZIE*18] measures subjective perception of image differences, and smaller LPIPS means they are more perceptually similar.

We can see from Table 3 that all four measurements agree that the order has the greatest impact on (*Blond hair, Pale skin*) transform. This is because these two attributes both affect a wide area on the image, and their modifications will cause more interference in the latent space. Our method does not have this problem since it does multi-attribute transforms simultaneously and has an internal increment independence mechanism to prevent interference.

**Multi-attribute interpolation.** Our method uses a parallel architecture and latent increment mechanism, and it can easily generate a series of interpolation results by interpolating the increments in the latent space, as shown in Fig. 8. Since the results of our method are more self-consistent when simultaneously manipulating multi-attribute, this property is helpful to make a self-consistent interpolation among two attributes, that is, the attribute transform results travel from the two attribute paths will meet to a consistent result, while each attribute is monotonically changing. Latent vector interpolation is generally common in disentangling methods, since they can interpolate between the source vector and the exemplar vector. However our method does not rely on explicit latent space disentangling, but is also fully capable for attribute interpolation.

**Ablation study of the auxiliary constraints.** To enhance the effect of our framework, we use several auxiliary constraints, including:

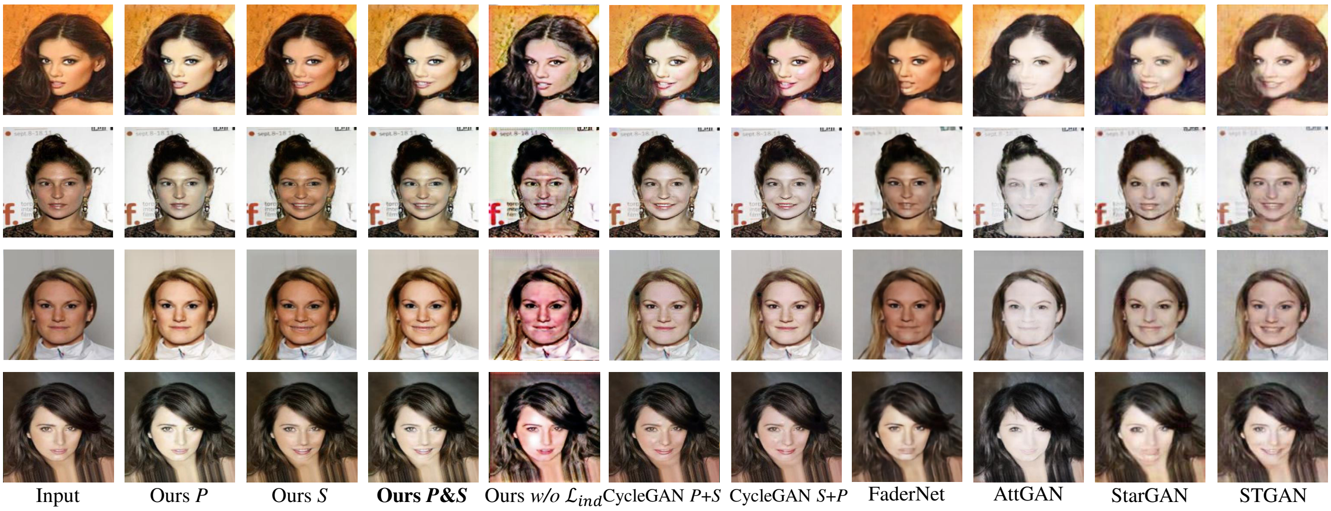

**Figure 7:** *Qualitative comparisons of* Pale skin *and* Smiling *transform.*

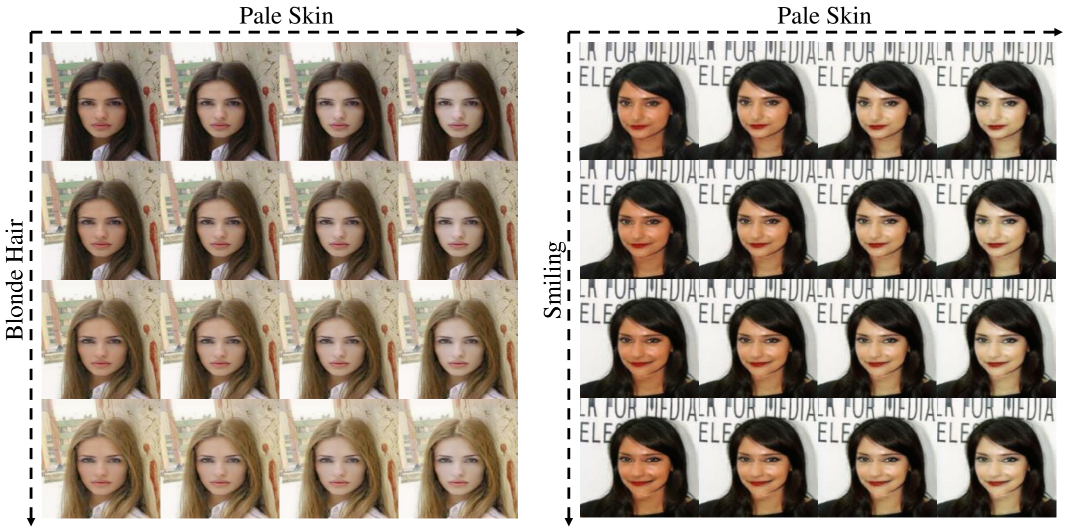

**Figure 8:** *Interpolation results of attribute transforms.*

1. increment stability loss $\mathcal{L}_{StbL1}$ and $\mathcal{L}_{StbVGG}$ to make the decoded images insensitive to the interference increments not generated by $LTN$;
2. latent consistency loss $\mathcal{L}_{LC}$ to enhance the one-to-one correspondence between latent vectors and images;
3. domain classification loss $\mathcal{L}_{C}$ to direct the attribute transform besides the adversarial loss.

In order to verify the effectiveness of these auxiliary constraints, we remove them separately and observe the effect on the results. We use the attributes of *Blonde hair, Pale skin* and their combinations to conduct our experiments.

Fig. 9 shows the qualitative results of the ablation study. When $\mathcal{L}_{StbL1}$ and $\mathcal{L}_{StbVGG}$ are removed, the decoded images become more sensitive to all disturbances on the latent vector, and lose their specific sensitivity to LTN. So there exist some slight artifacts on the image (eye area and ear area). When $\mathcal{L}_{LC}$ is removed, the one-to-one correspondence between latent vectors and images is weakened, and the re-encoded latent vector is not necessarily equal to the latent vector before decoding. This affects the effectiveness of the method, and the image quality decreases significantly and the results become distorted. When $\mathcal{L}_{C}$ is removed, the networks' ability to guide the attribute transforms is reduced. Although it is also possible to learn the attribute transforms by solely relying on the adversarial loss and reconstruction loss, the results become less natural than those with $\mathcal{L}_{C}$.

**Table 4:** *Quantitative comparison for facial image multi-attribute translation. Three sets of target attributes are tested:* B *for* Blond hair, P *for* Pale skin *and* S *for* Smiling.

| | | CycleGAN<br>*forward order* | CycleGAN<br>*reverse order* | FaderNet | AttGAN | StarGAN | STGAN | Ours<br>w/o $\mathcal{L}_{ind}$ | Ours |
|---|---|---|---|---|---|---|---|---|---|
| *B+P* | FID (with $B \cap P$ domain) | 31.872 | 29.815 | 60.453 | 72.240 | 52.366 | 29.209 | 32.266 | 28.844 |
| *B+S* | FID (with $B \cap S$ domain) | 27.939 | 27.687 | 53.160 | 30.610 | 41.047 | 25.908 | 33.548 | 23.241 |
| *P+S* | FID (with $P \cap S$ domain) | 35.138 | 34.103 | 55.748 | 62.927 | 51.696 | 29.302 | 56.894 | 31.396 |

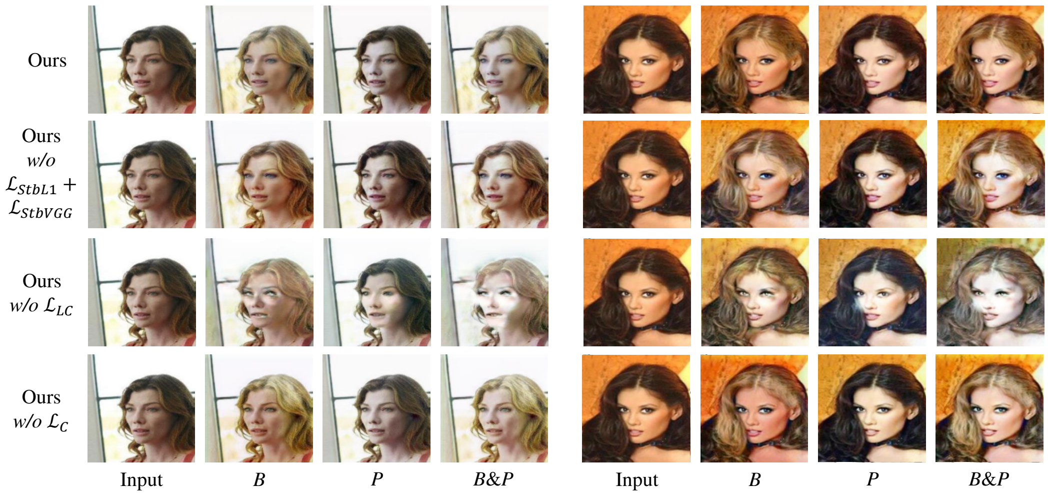
Ours
Ours w/o $\mathcal{L}_{StbL1}$ + $\mathcal{L}_{StbVGG}$
Ours w/o $\mathcal{L}_{LC}$
Ours w/o $\mathcal{L}_{C}$
Input
B
P
B&P
Input
B
P
B&P

**Figure 9:** *Qualitative results of the ablation study.*

### 4.3. Test on Artistic Style Transform

We further experiment with transferring two artistic styles (Van Gogh and Ukiyo-e, abbreviated as $V$ and $U$, respectively), and treat them as two attributes. We use the images provided by CycleGAN for training and testing. We compare our method with CycleGAN and StarGAN. The quantitatively results based on FID are reported in Table 5 and visual comparisons are shown in Fig. 10. As there are no ground truth images for the $V+U$ style, FID values are calculated with $V$ and $U$ domains separately. It can be seen that CycleGAN results tend to show stronger effects by the latter transform, and our results achieve a better balance with both styles. Although StarGAN can learn both attributes at the same time, it tends to generate results with artifacts, showing the significant interference between styles.

## 5. Conclusion

In this paper, we have proposed a novel approach to multi-attribute image-to-image translation. Our network architecture only requires to perform the encoder and decoder once for multiple attributes, and uses several parallel Latent Transform Networks to simultaneously transform multiple attributes. We further introduce an effective soft independence constraint to avoid interference between different attributes. Experimental results show that our method outperforms state-of-the-art methods both qualitatively and quantitatively.

Our method has some limitations. Since our method relies on the parallel LTNs in the middle of the framework for attribute transform, and our LTN is currently unidirectional, our current framework only has the concept of *adding* attributes. In our framework, the reverse operation of “removing” attributes is also considered as an “adding” task. If the forward and reverse LTNs for an attribute both exist in the framework, the independence constraint between them is removed. As future work, we will consider introducing positive and negative LTNs to make the network have two-way functions and improve efficiency.

## Acknowledgements

This work was supported by the Natural Science Foundation of China (Project Number 61521002), Research Grant of Beijing Higher Institution Engineering Research Center and Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology.

## References

[ACB17] ARJOVSKY M., CHINTALA S., BOTTOU L.: Wasserstein GAN. *arXiv preprint arXiv:1701.07875* (2017). 1

[CCCY18] CHEN J., CHEN J., CHAO H., YANG M.: Image blind denoising with generative adversarial network based noise modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 3155–3164. 1

**Table 5:** *Quantitative evaluation for style transfer with* Van Gogh *and* Ukiyo-e *styles.*

| | Ours | CycleGAN ($V$+$U$) | CycleGAN ($U$+$V$) | StarGAN |
|---|---|---|---|---|
| FID (with $V$ domain) | 136.16 | 151.14 | 139.19 | 201.86 |
| FID (with $U$ domain) | 144.44 | 121.62 | 145.38 | 215.74 |

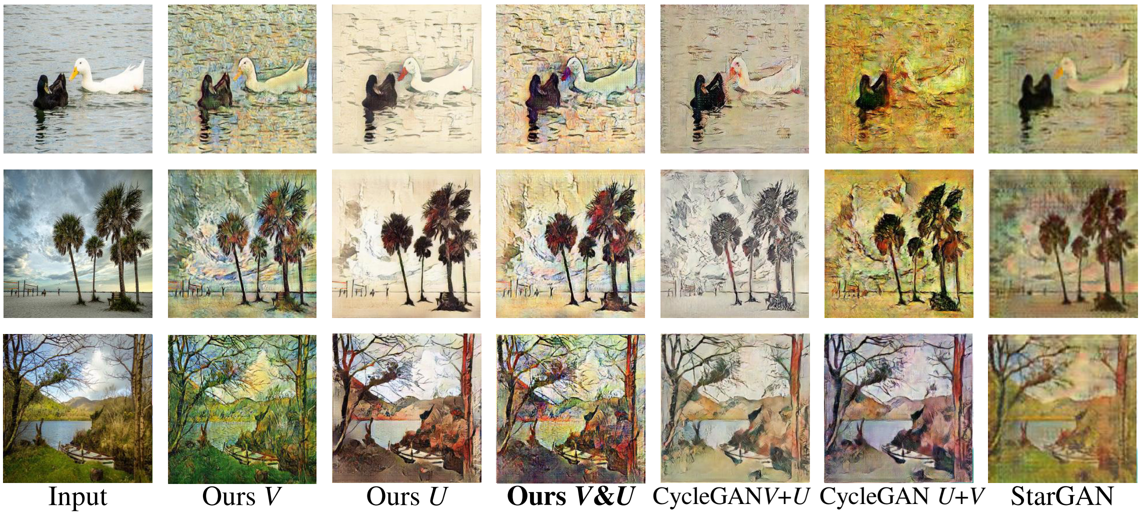

**Figure 10:** *Qualitative comparisons of* Van Gogh *and* Ukiyo-e *artistic style transform.*

[CCK*18] Choi Y., Choi M., Kim M., Ha J.-W., Kim S., Choo J.: StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 8789–8797. 1, 2, 6

[CDH*16] Chen X., Duan Y., Houthooft R., Schulman J., Sutskever I., Abbeel P.: InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems* (2016), pp. 2172–2180. 3

[CUYH20] Choi Y., Uh Y., Yoo J., Ha J.-W.: StarGAN v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 8188–8197. 6

[CZZY17] Cao Y., Zhou Z., Zhang W., Yu Y.: Unsupervised diverse colorization via generative adversarial networks. In *Joint European conference on machine learning and knowledge discovery in databases* (2017), Springer, pp. 151–166. 1

[FWK*18] Feng Z., Wang X., Ke C., Zeng A.-X., Tao D., Song M.: Dual swap disentangling. In *Advances in neural information processing systems* (2018), pp. 5894–5904. 3

[GPAM*14] Goodfellow I., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y.: Generative adversarial nets. In *Advances in neural information processing systems* (2014), pp. 2672–2680. 1

[HLY*20] Hu S.-M., Liang D., Yang G.-Y., Yang G.-W., Zhou W.-Y.: Jittor: A novel deep learning framework with unified graph execution and meta operators. *Science China-Information Sciences* (2020). URL: `https://github.com/Jittor/Jittor`. 6

[HRU*17] Heusel M., Ramsauer H., Unterthiner T., Nessler B., Hochreiter S.: GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in neural information processing systems* (2017), pp. 6626–6637. 6

[HSP*18] Hu Q., Szabó A., Portenier T., Favaro P., Zwicker M.: Disentangling factors of variation by mixing them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 3399–3407. 3

[HZ10] Hore A., Ziou D.: Image quality metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition* (2010), IEEE, pp. 2366–2369. 8

[HZK*19] He Z., Zuo W., Kan M., Shan S., Chen X.: AttGAN: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing 28*, 11 (2019), 5464–5478. 2, 6

[HZRS16] He K., Zhang X., Ren S., Sun J.: Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778. 6

[IZZE17] Isola P., Zhu J.-Y., Zhou T., Efros A. A.: Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 1125–1134. 1, 2, 5, 6

[KBJ19] Krull A., Buchholz T.-O., Jug F.: Noise2void-learning denoising from single noisy images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), pp. 2129–2137. 1

[KCK*17] Kim T., Cha M., Kim H., Lee J. K., Kim J.: Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), JMLR. org, pp. 1857–1865. 2

[KIN19] Kazemi H., Iranmanesh S. M., Nasrabadi N.: Style and content disentanglement in generative adversarial networks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2019), IEEE, pp. 848–856. 3

[KW13] Kingma D. P., Welling M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013). 2

[LAD*19] Li Y., Ao D., Dumitru C. O., Hu C., Datcu M.: Super-resolution of geosynchronous synthetic aperture radar images using dialectical GANs. *Science China Information Sciences 62*, 10 (2019), 209302. 1

[LBK17] Liu M.-Y., Breuel T., Kautz J.: Unsupervised image-to-image translation networks. In *Advances in neural information processing systems* (2017), pp. 700–708. 2

[LDX*19] Liu M., Ding Y., Xia M., Liu X., Ding E., Zuo W., Wen S.: STGAN: A unified selective transfer network for arbitrary image attribute editing. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2019), pp. 3673–3682. 2, 6

[LGLS19] Liu S., Gang R., Li C., Song R.: Adaptive deep residual network for single image super-resolution. *Computational Visual Media 5*, 4 (2019), 391–401. 1

[LLWT15] Liu Z., Luo P., Wang X., Tang X.: Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 3730–3738. 6

[LT16] Liu M.-Y., Tuzel O.: Coupled generative adversarial networks. In *Advances in neural information processing systems* (2016), pp. 469–477. 2

[LTH*17] Ledig C., Theis L., Huszár F., Caballero J., Cunningham A., Acosta A., Aitken A., Tejani A., Totz J., Wang Z., 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* (2017), pp. 4681–4690. 1

[LTH*18] Lee H.-Y., Tseng H.-Y., Huang J.-B., Singh M., Yang M.-H.: Diverse image-to-image translation via disentangled representations. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 35–51. 3

[LWS*18] Liu Y., Wei F., Shao J., Sheng L., Yan J., Wang X.: Exploring disentangled feature representation beyond face identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 2080–2089. 3

[LZU*17] Lample G., Zeghidour N., Usunier N., Bordes A., Denoyer L., Ranzato M.: Fader networks: Manipulating images by sliding attributes. In *Advances in neural information processing systems* (2017), pp. 5967–5976. 2, 6

[LZZ16] Li M., Zuo W., Zhang D.: Deep identity-aware transfer of facial attributes. *arXiv preprint arXiv:1610.05586* (2016). 1

[MCS18] Mo S., Cho M., Shin J.: InstaGAN: Instance-aware image-to-image translation. *arXiv preprint arXiv:1812.10889* (2018). 2

[MLX*17] Mao X., Li Q., Xie H., Lau R. Y., Wang Z., Paul Smolley S.: Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 2794–2802. 1, 5

[MO14] Mirza M., Osindero S.: Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014). 1, 2

[MRT*18] Mejjati Y. A., Richardt C., Tompkin J., Cosker D., Kim K. I.: Unsupervised attention-guided image-to-image translation. In *Advances in Neural Information Processing Systems* (2018), pp. 3693–3703. 2

[NNE18] Nazeri K., Ng E., Ebrahimi M.: Image colorization using generative adversarial networks. In *International conference on articulated motion and deformable objects* (2018), Springer, pp. 85–94. 1

[RMC15] Radford A., Metz L., Chintala S.: Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015). 1

[SZ15] Simonyan K., Zisserman A.: Very deep convolutional networks for large-scale image recognition. In *ICLR* (2015). 5

[TSL*17] Tsai Y.-H., Shen X., Lin Z., Sunkavalli K., Lu X., Yang M.-H.: Deep image harmonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 3789–3797. 1

[WBSS04] Wang Z., Bovik A. C., Sheikh H. R., Simoncelli E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*, 4 (2004), 600–612. 7

[WLZ*18] Wang T.-C., Liu M.-Y., Zhu J.-Y., Tao A., Kautz J., Catanzaro B.: High-resolution image synthesis and semantic manipulation with conditional GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018). 2

[WW19] Wong K.-M., Wong T.-T.: Deep residual learning for denoising monte carlo renderings. *Computational Visual Media 5*, 3 (2019), 239–255. 1

[WYW*18] Wang X., Yu K., Wu S., Gu J., Liu Y., Dong C., Qiao Y., Change Loy C.: ESRGAN: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 0–0. 1

[WZZH19] Wu H., Zheng S., Zhang J., Huang K.: GP-GAN: Towards realistic high-resolution image blending. In *Proceedings of the 27th ACM International Conference on Multimedia* (2019), pp. 2487–2495. 1

[XHM17] Xiao T., Hong J., Ma J.: DNA-GAN: Learning disentangled representations from multi-attribute images. *arXiv preprint arXiv:1711.05415* (2017). 3, 6

[XHM18] Xiao T., Hong J., Ma J.: ELEGANT: Exchanging latent encodings with gan for transferring multiple face attributes. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 168–184. 3, 6

[YCYL*17] Yeh R. A., Chen C., Yian Lim T., Schwing A. G., Hasegawa-Johnson M., Do M. N.: Semantic image inpainting with deep generative models. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 5485–5493. 1

[YLY*18] Yu J., Lin Z., Yang J., Shen X., Lu X., Huang T. S.: Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 5505–5514. 1

[YZTG17] Yi Z., Zhang H., Tan P., Gong M.: DualGAN: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2849–2857. 2

[ZIE*18] 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* (2018), pp. 586–595. 8

[ZPIE17] 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* (2017), pp. 2223–2232. 1, 2, 6