RSGAN: Face Swapping and Editing using Face and Hair Representation in Latent Spaces

Ryota Natsume(✉), Tatsuya Yatagawa, and Shigeo Morishima

Graduate School of Advanced Science and Engineering,
Waseda University, Tokyo, Japan
ryota.natsume.26@gmail.com, tatsy@acm.org, shigeo@waseda.jp

Abstract. In this paper, we present an integrated system for automatically generating and editing face images through face swapping, attribute-based editing, and random face parts synthesis. The proposed system is based on a deep neural network that variationally learns the face and hair regions with large-scale face image datasets. Different from conventional variational methods, the proposed network represents the latent spaces individually for faces and hairs. We refer to the proposed network as *region-separative generative adversarial network* (RSGAN). The proposed network independently handles face and hair appearances in the latent spaces, and then, face swapping is achieved by replacing the latent-space representations of the faces, and reconstruct the entire face image with them. This approach in the latent space robustly performs face swapping even for images which the previous methods result in failure due to inappropriate fitting or the 3D morphable models. In addition, the proposed system can further edit face-swapped images with the same network by manipulating visual attributes or by composing them with randomly generated face or hair parts.

1 Introduction

Human face is an important symbol to recognize individuals from ancient time to the present. Drawings of human faces have been used traditionally to record the human identities of people in authorities. Nowadays, many people enjoy sharing their daily photographs, which usually includes human faces, in social networking websites. In these situations, there has been a potential demand for making the drawings or photographs to be more attractive. As a result of this demand, a large number of studies for face image analysis [1–4] and manipulation [5–11] have been introduced in the research communities of computer graphics and vision.

Face swapping is one of the most important techniques of face image editing that has a wide range of practical applications such as photomontage [5], virtual hairstyle fitting [9], privacy protection [6, 12, 13], and data augmentation for machine learning [14–16]. The traditional face swapping methods firstly detect face regions in source and target images. The face region of the source image is embedded into the target image by digital image stitching. To clarify the

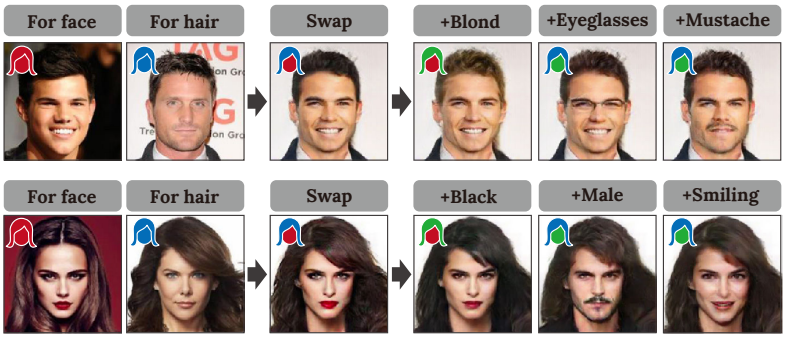

Fig. 1. Results of face swapping and additional visual attribute editing with the proposed system. The face regions of images in the first column are embedded to the images in the second column. The face-swapped results are illustrated in the third column, and their appearances are further manipulated by adding visual attributes such as "blond hair" and "eyeglasses". RSGAN can obtain these results by passing the two input images and visual attributes through the network only once.

motivation of our study, we briefly review the previous face-swapping methods in the following paragraphs.

One of the most popular approaches of face swapping is to use the 3D morphable models (3DMM) [5, 17]. In this class of methods, the face geometries and their corresponding texture maps are first obtained by fitting 3DMM [1, 2]. The texture maps of the source and target images are then swapped with the estimated UV coordinates. Finally, the replaced face textures are re-rendered using the estimated lighting condition estimated with the target image. These approaches with the 3DMM can replace the faces even for those with different orientations or in the different lighting conditions. However, these methods are prone to fail the estimations of face geometries or lighting conditions in practice. The incorrect estimations are usually problematic because people can sensitively notice even slight mismatches of these geometries and lighting conditions.

In specific applications of face swapping, such as privacy protection and virtual hairstyle fitting, either of the source image or target image can be selected arbitrarily. For instance, the privacy of the target image can be protected even though the new face region is extracted from a random image. This has suggested an idea of selecting one of the source and target image from large-scale image databases [6, 9]. The approaches in this class can choose one of two input images such that a selected image is similar to its counterpart. These approaches can consequently avoid replacing faces in difficult situations with different face orientations or different lighting conditions. However, these methods cannot be used for more general purposes of face swapping between arbitrary input face images.

A vast body of recent deep learning research has facilitated face swapping with large-scale image databases. Bao et al. have introduced a face swapping

demo in their paper of conditional image generation with their proposed neural network named CVAE-GAN [18]. Their method uses hundreds of images for each person in the training dataset, and the face identities are learned as the image conditions. A similar technique is used in a desktop software tool "FakeApp" [19] that has recently attracted much attention due to its easy-to-use pipeline for face swapping with deep neural networks (DNN). This tool requires hundreds of images of the two target people for swapping the faces. However, preparing such a large number of portrait images for non-celebrities is rather inadvisable. In contrast to these techniques, Korshunova et al. [13] have applied the neural style transfer [20] to face swapping by fine-tuning the pre-trained network with several tens of images of a single person in a source image. Unfortunately, it is still impractical for most of people to collect many images and to fine-tune the network for generating a single face-swapped image.

In this paper, we address the above problems using a generative neural network that we refer to as "region-separative generative adversarial network (RSGAN)." While a rich body of studies for such deep generative models has already been introduced, applying it to face swapping is still challenging. In ordinary generative models, the images or data which a network synthesizes are obtained as training data. However, it is difficult or even impossible to prepare a dataset which includes face images both before and after face swapping because the faces of real people can hardly be swapped without special surgical operations. We tackle this problem by designing the network to variationally learn different latent spaces for each of face and hair regions. A generator network used in the proposed method is trained to synthesize a natural face image from two random vectors that correspond to latent-space representations of face and hair regions. In consequence, the generator network can synthesize a face-swapped image from two latent-space representations calculated from real image samples. The architecture of RSGAN is illustrated in Fig. 2. This architecture consists of two variational autoencoders (VAE) and one generative adversarial network (GAN). The two VAE parts encode face and hair appearances into latent-space representations, and the GAN part generates a natural face image from the latent-space representations of faces and hairs. The detailed description of the network and its training method are introduced in Sec. 3. In addition to face swapping, this variational learning enables other editing applications, such as visual attribute editing and random face parts synthesis. To evaluate face swapping results of the proposed method, we leveraged two metrics of identity preservation and swap consistency. The identity preservation is evaluated using OpenFace [21], which is an open-source face feature extractor. The consistency of face swapping is evaluated by measuring the absolute difference and multi-scale structural similarity (MS-SSIM) [22] between a input image and a resulting image obtained by swapping faces twice between two input images. The results of applications of RSGAN and their evaluations are shown in Sec. 4.

Contributions: As a face swapping and editing system, the proposed method has following advantages over previous methods:

1. it provides an integrated system for face swapping and additional face appearance editing;
2. its applications are achieved by training a single DNN, and it does not require any additional runtime computation such as fine-tuning;
3. it robustly performs high-quality face swapping even for faces with different face orientations or in different lighting conditions;

2 Related Work

2.1 Face swapping

Face swapping has been studied in a number of research for different purposes, such as photomontage [5], virtual hairstyle fitting [9], privacy protection [6,12,13] and data augmentation for large-scale machine learning [16]. Several studies [7,12] have replaced only parts of the face, such as eyes, nose, and mouth between images rather than swapping the whole face between images. As described in the previous section, one of the traditional approaches for face swapping is based on 3DMM [5,17]. Fitting 3DMM to a target face obtains face geometry, texture map and lighting condition approximately [1, 2]. Using the 3DMM, face swapping is achieved by replacing texture maps and re-rendering the face appearance using the estimated lighting condition. The main drawback of these 3DMM-based methods is that they require manual alignment of the 3DMM to obtain accurate fitting. To alleviate this problem, Bitouk et al. [6] proposed automatic face swapping with a large-scale face image database. Their method first searches a face image with a similar layout to the input image, and then replace the face regions with boundary-aware image composition. A more sophisticated approach was recently proposed by Kemelmacher-Shlizerman [9]. She carefully designed a handmade feature vector to face image appearances and achieved high-quality face swapping. However, these methods by searching similar images cannot freely select the input images, and are not applicable to arbitrary face image pairs. Recently, Bao et al. have introduced a face swapping demo in their paper of CVAE-GAN [18], which is a DNN for conditional image generation. In their method, the CVAE-GAN is trained to generate face images of specific people in a training dataset by handling face identities as conditions for generated images. The CVAE-GAN achieves face swapping by changing the conditions of face identities of target images. Korshunova et al. applied neural style transfer, which is another technique of deep learning, to face swapping [13]. Their approach is similar to the original neural style transfer [20] in the sense that a face identity is handled similarly to an artistic style. The face identity of a target face is substituted by that of a source face. The common drawback of these DNN-based models is that users must collect at least dozens of images to obtain a face-swapped image. While collecting such a number of images is possible, it is nevertheless impractical for most of people to collect the images just for their personal photo editing.

2.2 Face image editing

To enhance the visual attractiveness of face images, various techniques, such as facial expression transfer [7, 23], attractiveness enhancement [24], face image relighting [25, 26], have been proposed in the last decades. In traditional face image editing, underlying 3D face geometries and face parts arrangements are estimated using face analysis tools, such as active appearance models [27] and 3D morphable models [1,2]. These underlying information are manipulated in editing algorithms to improve attractiveness of output images. On the other hand, recent approaches based on DNNs do not explicitly analyze such information. Typically, an input image and a user's edit intention are fed to an end-to-end DNN, and then, the edit result is directly output from the network. For example, several DNN models [18, 28–30] based on autoencoders are used to manipulate visual attributes of faces, in which visual attributes, such as facial expressions and hair colors, are modified to change face image appearances. In contrast, Brock et al. [31] proposed an image editing system with the paint-based interface in which a DNN synthesizes a natural image output following the input image and paint strokes specified by the users. Several studies for DNN-based image completion [32, 33] have presented demos of manipulating face appearances by filling the parts of an input image with the DNN. However, estimating results of these approaches is rather difficult because they only fill the regions painted by the users such that completed results plausibly exists in training data.

3 Region-Separative GAN

The main challenge of face swapping with DNNs is the difficulty of preparing face images before and after face swapping because the face of a real person cannot be replaced by that of another person without a special surgical operation. Another possible way to collect such face images is to digitally synthesize them. However, it is an chicken-and-egg problem because the synthesis of such face-swapped images is our primary purpose. To overcome this challenge, we leverage a variational method to represent appearances of facesand hairs. In face swapping, a face region and a hair region are handled separately in the image space. The face swapping problem is generalized as a problem of composing any pair of face and hair images. The purpose of the proposed RSGAN is to achieve this image composition using latent-space representations of face and hair appearances. In the proposed method, this purpose is achieved by a DNN shown in Fig. 2. As shown in this figure, the architecture of RSGAN consists of two VAEs, which we refer to as *separator network*, and one GAN, which we refer to as *composer network*. In this network, appearances of the face and hair regions are first encoded into different latent-space representations with the separator networks. Then, the composer network generates a face image with the obtained latent-space representations so that the original appearances in the input image is reconstructed. However, training with only latent-space representations from real image samples incurs over-fitting. We found that a RSGAN trained in this

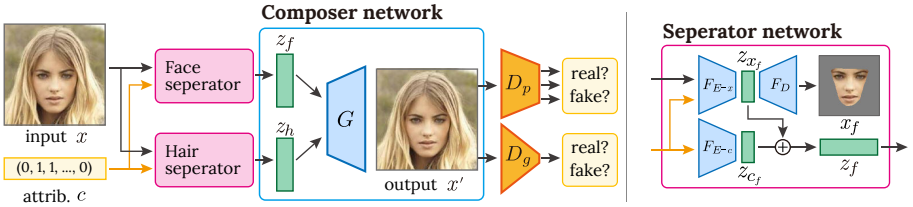

Fig. 2. The network architecture of the proposed RSGAN that comprises three partial networks, i.e., two separator networks and a composer network. The separator networks extract latent-space representations z_f and z_h respectively for face and hair regions of an input image x. The composer network reconstructs the input face image from the two latent-space representations. The reconstructed image x' and input image x are evaluated by two discriminator networks. The global discriminator D_g distinguishes whether the images are real or fake, and the patch discriminator D_p distinguishes whether local patches of the images are real or fake.

way ignores the face representation is the latent space, and synthesizes an image similar to the target image while face swapping. Thus, we also feed random latent-space representations to the composer network such that they are trained to synthesize natural face images rather than over-fitting the training data.

Let x be a training image, and c be its corresponding visual attribute vector. Latent-space representations z_{x_f} and z_{x_h} of face and hair appearances of x are obtained by a face encoder $F_{E\text{-}x_f}$ and a hair encoder $F_{E\text{-}x_h}$. Similarly, the visual attribute c is embedded into latent spaces of the attributes. Latent-space representations z_{c_f} and z_{c_h} of the face and hair attribute vectors are obtained by encoders $F_{E\text{-}c_f}$ and $F_{E\text{-}c_h}$. As standard VAEs, these latent-space representations are sampled from multivariate normal distributions whose averages and variances are inferred by the encoder networks:

$$z_\ell = \mathcal{N}\left(\mu_\ell, \sigma_\ell^2\right), \quad \left(\mu_\ell, \sigma_\ell^2\right) = F_{E\text{-}\ell}(x, c), \quad \ell \in \{x_f, x_h, c_f, c_h\},$$

where μ_ℓ and σ_ℓ^2 are the average and variance of z_ℓ obtained with the encoders. Decoder networks $F_{D\text{-}f}$ and $F_{D\text{-}h}$ for face and hair regions reconstruct the appearances x'_f and x'_h respectively from the corresponding latent-space representation. The composer network G generates the reconstructed appearance x' with the latent-space representations from the encoders. These reconstruction processes are formulated as:

$$x'_f = F_{D\text{-}f}(z_{x_f}, z_{c_f}), \quad x'_h = F_{D\text{-}h}(z_{x_h}, z_{c_h}), \quad x' = G(z_{x_f}, z_{c_f}, z_{x_h}, z_{c_h}).$$

In addition, random variables sampled from a multivariate standard normal distribution $\mathcal{N}(0, 1)$ are used together in the training. Let $\hat{z}_{x_f}$, $\hat{z}_{x_h}$, $\hat{z}_{c_f}$, and $\hat{z}_{c_h}$ be the random variables which correspond to z_{x_f}, z_{x_h}, z_{c_f}, and z_{c_h}, respectively. We also compute a random face image $\hat{x}'$ with these samples:

$$\hat{x}' = G(\hat{z}_{x_f}, \hat{z}_{c_f}, \hat{z}_{x_h}, \hat{z}_{c_h}).$$

The input image x and two generated images x' and $\hat{x}'$ are evaluated by two discriminator networks D_g and D_p. The global discriminator D_g distinguishes whether those images are real or fake as in standard GANs [34]. On the other hand, the patch discriminator D_p, which is originally used in an image-to-image network [35], distinguishes whether local patches from those images are real or fake. In addition, we train a classifier network C to estimate the visual attribute c^* from the input image x. The classifier network is typically required to edit an image for which visual attributes are not prepared. In addition, the classifier network obtains a visual attribute vector whose entries are in between 0 and 1, whereas visual attribute vectors prepared in many public datasets take discrete values of 0 or 1. Such intermediate values are advantageous, for example, when we represent dark brown hair with two visual attribute items "black hair" and "brown hair". Accordingly, we use the estimated attributes c^* rather than c even when visual attributes are prepared for x.

3.1 Training

In the proposed architecture of RSGAN, three autoencoding processes are performed, each of those reproduces x'_f, x'_h and x' from an input image x. Following standard VAEs, we define three reconstruction loss functions:

$$\begin{aligned}
\mathcal{L}_{rec\text{-}f} &= \mathbb{E}_{x,x_f\sim P_{data}}\left[\|x_f - x'_f\|_1\right],\\
\mathcal{L}_{rec\text{-}h} &= \mathbb{E}_{x,x_h\sim P_{data}}\left[\,\|(1-\beta M_{BG})\odot(x_h - x'_h)\|_1\,\right],\\
\mathcal{L}_{rec} &= \mathbb{E}_{x\sim P_{data}}\left[\,\|(1-\beta M_{BG})\odot(x - x')\|_1\,\right],
\end{aligned}$$

where M_{BG} is a background mask which take 0 for foreground pixels and 1 for background pixels, and an operator $\odot$ denotes per-pixel multiplication. The background mask M_{BG} is used to train the network to synthesize more detailed appearances in foreground regions. In our implementation, we used a parameter $\beta = 0.5$ to halve the least square errors in the background. The Kullback Leibler loss function is also defined as standard VAEs for each of the four encoders:

$$\mathcal{L}_{KL\text{-}\ell} = \frac{1}{2}\left(\mu_\ell^T\mu_\ell + \sum\left(\sigma_\ell - \log(\sigma_\ell) - 1\right)\right), \qquad \ell \in \{x_f, x_h, c_f, c_h\}.$$

The set of separator and composer networks, and the two discriminator networks are trained adversarially as standard GANs. Adversarial losses are defined as:

$$\begin{aligned}
\mathcal{L}_{adv\text{-}\ell} = &-\mathbb{E}_{x\sim P_{data}}\left[\log D_\ell(x)\right]\\
&-\mathbb{E}_{z\sim P_z}\left[\log(1-D_\ell(x'))\right]\\
&-\mathbb{E}_{z\sim P_z}\left[\log(1-D_\ell(\hat{x}'))\right], \quad \ell \in \{g, p\}.
\end{aligned}$$

In addition, the classifier network C is trained to estimate correct visual attributes c^* which is close to c. We defined a cross entropy loss function L_{BCE}, and used it to define a loss function of classifier network $\mathcal{L}_C$:

$$\mathcal{L}_C = L_{BCE}(c, c^*) := -\sum_i \left(c_i \log c_i^* + (1-c_i)\log(1-c_i^*)\right),$$

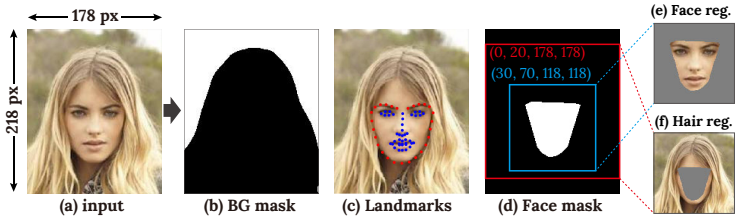

Fig. 3. The process of generating face and hair region images from portraits in CelebA [3]. The background mask in (b) is computed with PSPNet [37], which is a state-of-the-art DNN-based semantic segmentation. Clipping rectangles in (d) for face and hair regions are computed using blue facial landmarks in (c). To improve the reconstruction quality of face identities, face regions are magnified with larger scale than hair regions.

where c_i denotes the i-th entry of the visual attribute vector c. To preserve the visual attributes in generated images x' and $\hat{x}'$, we add the following loss functions to train the composer network:

$$\mathcal{L}_{GC} = L_{BCE}(c, c') + L_{BCE}(c, \hat{c}'),$$

where c' and $\hat{c}'$ are estimated visual attributes of x' and $\hat{x}'$, respectively.

The total loss function for training RSGAN is defined by a weighted sum of the above loss functions:

$$\begin{aligned}\mathcal{L} = &\lambda_{rec}(\mathcal{L}_{rec\text{-}f} + \mathcal{L}_{rec\text{-}h} + \mathcal{L}_{rec}) \\ &+ \lambda_{KL}(\mathcal{L}_{KL\text{-}x_f} + \mathcal{L}_{KL\text{-}x_h} + \mathcal{L}_{KL\text{-}c_f} + \mathcal{L}_{KL\text{-}c_h}) \\ &+ \lambda_{adv\text{-}g}\mathcal{L}_{adv\text{-}g} + \lambda_{adv\text{-}p}\mathcal{L}_{adv\text{-}p} \\ &+ \lambda_C\mathcal{L}_C + \lambda_{GC}\mathcal{L}_{GC}\end{aligned}$$

We empirically determined the weighting factors as $\lambda_{rec} = 4000$, $\lambda_{KL} = 1$, $\lambda_{adv\text{-}g} = 20$, $\lambda_{adv\text{-}p} = 30, \lambda_C = 1$, and $\lambda_{GC} = 50$. In our experiment, the loss functions were minimized using ADAM optimizer [36] with an initial learning rate of 0.0002, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. The size of a mini-batch was 50. The detailed training algorithm is provided in the supplementary materials.

3.2 Dataset

The training of RSGAN requires to sample face image x, face region image x_f, hair region image x_h, and background mask M_{BG} from real samples. For this purpose, we computationally generate face and hair region images with a large-scale face image dataset, i.e., CelebA [3]. Figure 3 illustrates the process of dataset generation. The size of original images in CelebA is 178×218 (Fig. 3(a)). We first estimate the foreground mask using PSPNet [37], which is a state-of-the-art semantic segmentation method, with the "person" label. The background

mask is obtained by inverting masked and non-masked pixels in the foreground mask (Fig. 3(b)). Second, we extract 68 facial landmarks (Fig. 3(c)) with a common machine learning library, i.e., Dlib [38]. The face region is defined with the 41 landmarks that correspond to eyes, nose, and mouth, which are indicated with blue circles in Fig. 3(c). We calculate a convex hull of these landmarks and stretch the hull by 1.3 times and 1.4 times along horizontal and vertical directions, respectively. The resulting hull is used as a face mask as depicted in Fig. 3(d). The face and hair regions are extracted with the mask and crop these regions to be square (Fig. 3(e) and (f)). The face region has its top-left corner at $(30, 70)$ and its size is 118×118. The hair region has its top-left corner at $(0, 20)$ and its size is 178×178. Finally, we resize these cropped images to the same size. In our experiment, we resize them to 128×128. Since the face region is more important to identify a person in the image, we used a higher resolution for the face region. While processing images in the dataset, we could properly extract the facial landmarks for 195,361 images out of 202,599 images included in CelebA. Among these 195,361 images, we used 180,000 images for training and the other 15,361 images for testing.

3.3 Face swapping with RSGAN

A face-swapped image is computed from two images x_1 and x_2 with RSGAN. For each of these images, visual attributes c_1^* and c_2^* are first estimated by the classifier. Then, latent-space representations of these variables z_{1,x_f}, z_{1,c_f}, z_{2,x_h}, and z_{2,c_h} are computed by the encoders. Finally, the face-swapped image is generated by the composer network as $x' = G(z_{1,x_f}, z_{1,c_f}, z_{2,x_h}, z_{2,c_h})$. This operation of just feeding two input images to RSGAN usually performs face swapping appropriately. However, hair and background regions in an input image are sometimes not recovered properly with RSGAN. To alleviate this problem, we optionally perform gradient-domain image stitching for a face-swapped image. In this operation, the face region of a face-swapped image is extracted with a face mask, which is obtained in the same manner as in the dataset generation. Then, the face region of the face-swapped image is composed of the target image by a gradient-domain image composition [39]. In order to distinguish these two approaches, we denote them as "RSGAN" and "RSGAN-GD", respectively. Unless otherwise specified, the results shown in this paper are computed only using RSGAN without the gradient-domain stitching.

4 Results and Discussion

This section introduces the results of applications using the pre-trained RSGAN. For these results, we implemented a program using TensorFlow [40] in Python, and executed it on a computer with an Intel Xeon 3.6 GHz E5-1650 v4 CPU, NVIDIA GTX TITAN X GPU, and 64 GB RAM. We used 180,000 training images, and trained the proposed RSGAN network over 120,000 global steps. The training spent about 50 hours using a single GPU. All the results in this

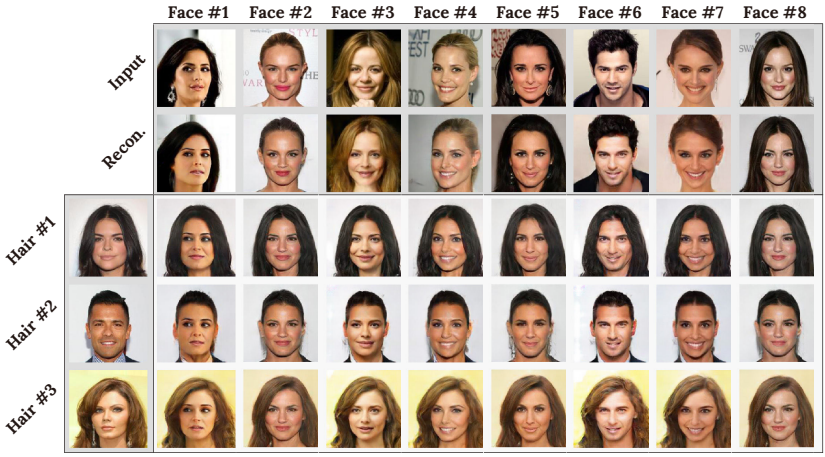

Fig. 4. Face swapping results for different face and hair appearances. Two top rows in this figure represent the original inputs and their reconstructed appearances obtained by RSGAN. In these results, face regions of the images in each row is replaced by a face from the image in each column.

paper are generated using test images which are not included in the training images.

Face swapping: Face-swapping results of the proposed system are illustrated in Fig. 4. In this figure, the first row illustrates source images, the second row illustrates reproduced appearances for the source images, and the bottom three rows illustrate face-swapped results for different target images in the leftmost column. In each result, we observe that face identities, expressions, shapes of facial parts, and shading are naturally presented in face-swapped results. Among these input image pairs, there are a large difference in the facial expression between Face #7 and Hair #1, a difference in the face orientation between Face #4 and Hair #2, and a difference in the lighting condition in Face #3 and Hair #1.

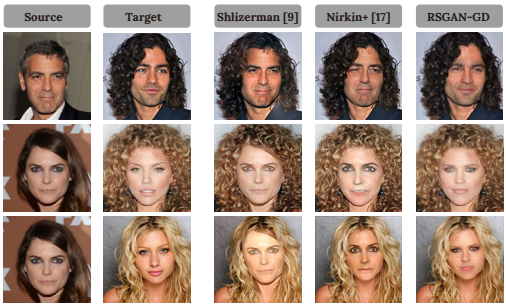

Fig. 5. Comparisons to the state-of-the-art face swapping methods [9, 17].

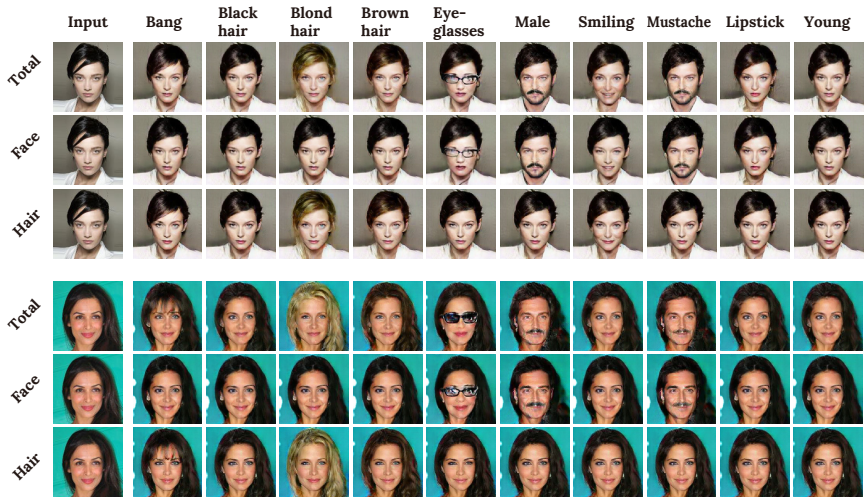

Fig. 6. Results of visual attribute editing using RSGAN. In this figure, the results in the rows marked with "Total" are obtained adding a new visual attribute both for face and hair regions. The visual attributes used for these results are indicated in the top. On the other hand, the results in the rows marked with "Face" are obtained by adding a new visual attribute only for the face region, and the original visual attributes are used for the hair region. The results in the rows marked with "Hair" are generated in the same way.

Even for such input pairs, the proposed method achieves natural face swapping. In addition, we compared our face-swapping results with the state-of-the-art methods [9, 17] in Fig. 5. The results are compared for the input images used in [9] that are searched from a large-scale database such that their layouts are similar to the source images. While these input images are more favorable for [9], the results of our RSGAN-GD are compatible to those of [9]. Compared to the other state-of-the-art method [17], the sizes of facial parts in our results look more natural in the sense that the proportions of the facial parts to the entire face sizes are more similar to those in the source images. As reported in the paper [17], these performance losses are due to their sensitiveness to the quality of landmark detection and 3DMM fitting, even though their proposed semantic segmentation is powerful.

Visual attribute editing: To perform face swapping together with visual attribute vectors, the proposed RSGAN embeds visual attribute vectors into the latent spaces of face and hair visual attributes. As a result, the proposed editing system enables to manipulate the attributes in only either of face or hair region. The results of visual attribute editing are illustrated in Fig. 6. This figure includes two image groups, each of which has three rows. In the first row, visual attributes indicated on the top is added to both face and hair regions.

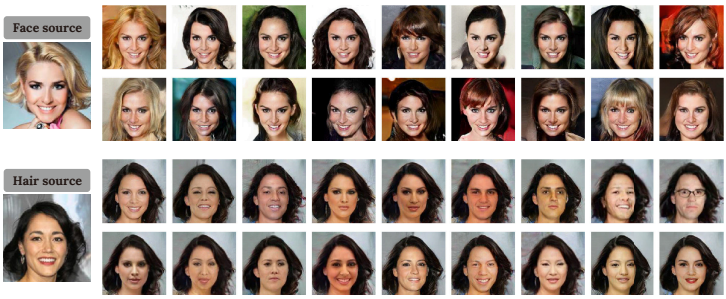

Fig. 7. Results of random face and hair parts generation and composition. We can sample independent latent-space representations for face and hair appearances, and combine them with the proposed RSGAN. In the top group, random hair appearances are combined with the face region of an input image in the left. In the bottom group, random face appearances are combined with the hair region of an input image.

In the second and third rows, the visual attributes are added to either of the face or hair region. As shown in this figure, adding visual attributes for only one of face and hair region do not affect the other region. For example, the hair colors have not been changed when the attribute "Blond hair" is added to the face regions. In addition, the attributes such as "Male" and "Aged", which can affect both regions, change the appearance of only one region when they are added to either of two regions. For example, the attribute "Male" is added to the face regions, only face appearances become masculine while hair appearances are not changed. In addition, the visual attribute editing can be applied to the face-swapped images with RSGAN. The results for this application is shown in Fig. 1. Note that RSGAN can achieve both face swapping and visual attribute editing by feeding two input images and modified visual attribute vectors to the network at the same time.

Random face parts synthesis: With the proposed RSGAN, we can generate a new face image which has an appearance of face or hair in a real image sample, and an appearance of the counterpart region defined by a random latent-space. Such random image synthesis is used in privacy protection by changing face regions randomly, and in data augmentation for face recognition by changing hair regions randomly. The results for the random image synthesis are shown in Fig. 7. This figure consists of two group of images in the top and bottom. In each group, an input image is shown in the left. Its face or hair region is combined with random hair or face region in the right images. The top group illustrates the images with random hairs, and the bottom group illustrates those with random faces. Even though the random face and hair regions cover a significant range of appearances, the appearances of face and hair in the real inputs are preserved appropriately in the results.

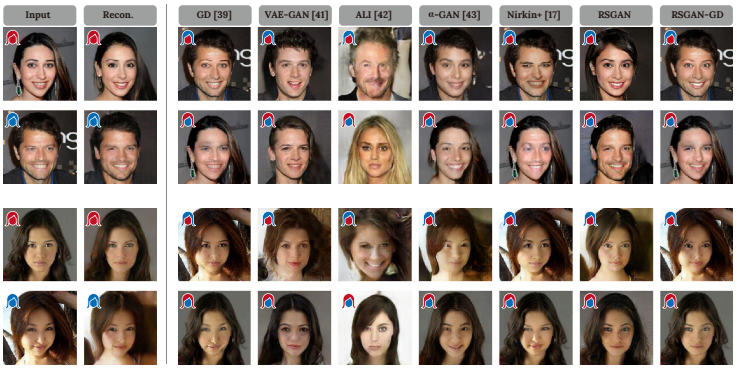

Fig. 8. Face swapping results of the proposed RSGAN and the other methods compared in Table 1. In two image groups in the top and bottom, face regions of the two input images in the leftmost column are replaced.

Table 1. Performance evaluation in identity preservation and swap consistency.

		OpenFace	Abs. Errors		MS-SSIM	
		Swap	Recon.	Swap ×2	Recon.	Swap ×2
VAE-GAN [41]	Avg.	1.598	0.082	0.112	0.694	0.563
	Std.	0.528	0.018	0.024	0.089	0.099
ALI [42]	Avg.	1.687	0.230	0.270	0.338	0.254
	Std.	0.489	0.065	0.068	0.133	0.108
α-GAN [43]	Avg.	1.321	**0.058**	0.099	**0.823**	0.638
	Std.	0.465	0.013	0.026	0.057	0.102
Nirkin et al. [17]	Avg.	**0.829**	—	**0.027**	—	**0.961**
	Std.	0.395	—	0.010	—	0.022
RSGAN	Avg.	*1.127*	*0.069*	*0.093*	*0.760*	*0.673*
	Std.	0.415	0.016	0.020	0.074	0.087

4.1 Experiments

We evaluated the face swapping results of the proposed and other previous methods using two metrics, i.e., identity preservation and swap consistency. In this experiment, we compared these two values with previous self-reproducing generative networks VAE-GAN [41], ALI [42], α-GAN [43] and the state-of-the-art face swapping method by Nirkin et al. [17]. With the generative networks, we computed face-swapped results in three steps. First, we compute a face mask in the same manner as in our dataset synthesis. Second, the face region of the source image in the mask is copy-and-pasted to the target image such that the two eye locations are aligned. Finally, the entire image appearance after copy-and-pasting is repaired by feeding it to self-reproducing networks. The examples of the face-swapped images made by these algorithms are illustrated in Fig. 8. We computed the results of different algorithms for 1,000 random image pairs selected from 15,361 test images. The averages and standard deviations of the

two metrics are provided in Table 1. In this table, the best score in each column is indicated with bold characters, and the second-best score is indicated with italic characters.

The identity preservation in face swapping is evaluated by the squared Euclidean distance between feature vectors of the input and face-swapped images. The feature vectors are computed with OpenFace [21], which is an open-source face feature extractor. The measured distances in the third column indicate that RSGAN outperform the other generative neural networks but it performs worse than Nirkin et al.'s method. However, the method of Nirkin et al. could perform face swapping only 81.7% of 1,000 test image pairs used in this experiment because it often fails to fit the 3DMM to at least one of the two input images. In contrast, the proposed RSGAN and the other method based on generative neural networks perform face swapping for all the test images. Therefore, we consider face swapping by RSGAN is practically useful even though the identity preservation is slightly worse than the state-of-the-art method of Nirkin et al.

The swap consistency is evaluated with an absolute difference and MS-SSIM [22] between an input image and a resulting image obtained after swapping the face region twice between two input images. For the previous generative neural networks and RSGAN, we computed these values also for images reconstructed by the networks. As shown in Table 1, evaluation results with absolute errors and MS-SSIM indicate that the method of Nirkin et al. outperforms the generative neural networks including RSGAN. We consider this is because Nirkin et al.'s method generates only a face region while face swapping, whereas the generative neural networks synthesize both the face and hair regions. Therefore, the scores of absolute differences and MS-SSIM becomes relatively lower for the method of Nirkin et al. in which the differences in pixel values only occur in the face regions. In addition, Nirkin et al.'s method is rather unstable for using in practice as mentioned in the previous paragraph. Consequently, the proposed method with RSGAN is worth using in practice because it has achieved the best swap consistency compared to the other generative neural networks as can be seen in the "Swap $\times 2$" columns.

4.2 Discussion

Variational vs non-variational: For the purpose of visual feature extraction, many variants of autoencoders have been used [28, 44–46]. Among these approaches, non-variational approaches are often preferably used when a real image appearance needs to be reproduced in their applications. For example, recent studies [45, 46], which have introduced a similar idea to our study, used non-variational approaches for image parts extraction [45] and manipulation of people's ages in portraits [46]. We have also experimented non-variational one of the proposed RSGAN in a prototype implementation, and we found that its self-reproducibility is slightly better than the variational one that is introduced in this paper. However, considering the wide applicability of the variational one of RSGAN such as random face parts sampling, we determined that the variational one is practically more useful than the non-variational one.

Region memorization with RNN: In image parts synthesis, some of the previous studies have applied the recurrent neural networks to memorize which parts are already synthesized or not [44, 45, 47]. Following these studies, we have experimentally inserted the long-short term memory (LSTM) [48] such that the outputs from the two image encoder networks are fed to it. However, in our experiment, we found that this application of the LSTM makes the training difficult and its convergence slower. The visual qualities of the results in face-swapping and the other applications are not significantly better than RSGAN without LSTM. We illustrated the RSGAN architecture with LSTM, and the results of this network in the supplementary materials.

Limitation: The main drawback of the proposed system is its limited image resolution. In our implementation, the image size of in a training dataset is 128×128. Therefore, the image editing can be performed only in this resolution. To improve the image resolution, we need to train the network with higher-resolution images as in CelebA-HQ [49]. In recent studies [33, 49], training with such a high-resolution image dataset is robustly performed by progressively increasing the resolutions of input images. This approach can be straightforwardly applied to the proposed RSGAN as well. Therefore, the limited image resolution of the proposed system will be evidently resolved.

5 Conclusion

This paper proposed an integrated editing system for face images using a novel generative neural network that we refer to as RSGAN. The proposed system achieves high-quality face swapping, which is the main scope of this study, even for faces with different orientations and in different lighting conditions. Since the proposed system can encode the appearances of faces and hairs into underlying latent-space representations, the image appearances can be modified by manipulating the representations in the latent spaces. As a deep learning technique, the success of the RSGAN architecture and our training method implies that deep generative models can obtain even a class of images that are not prepared in a training dataset. We believe that our experimental results provide a key for generating images which are hardly prepared in a training dataset.

Acknowledgments

This study was granted in part by the Strategic Basic Research Program ACCEL of the Japan Science and Technology Agency (JPMJAC1602). Tatsuya Yatagawa was supported by a Research Fellowship for Young Researchers of Japan's Society for the Promotion of Science (16J02280). Shigeo Morishima was supported by a Grant-in-Aid from Waseda Institute of Advanced Science and Engineering. The authors would also like to acknowledge NVIDIA Corporation for providing their GPUs in the academic GPU Grant Program.

References

1. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3d faces. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques. (1999) 187–194
2. Cao, C., Weng, Y., Zhou, S., Tong, Y., Zhou, K.: Facewarehouse: A 3d facial expression database for visual computing. IEEE Tranactions on Visualization and Computer Graphics **20**(3) (2014) 413–425
3. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: IEEE International Conference on Computer Vision (ICCV). (2015) 3730–3738
4. Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. IEEE Signal Processing Letters **23** (2016) 1499–1503
5. Blanz, V., Scherbaum, K., Vetter, T., Seidel, H.P.: Exchanging faces in images. Computer Graphics Forum **23**(3) (2004) 669–676
6. Bitouk, D., Kumar, N., Dhillon, S., Belhumeur, P., Nayar, S.K.: Face swapping: automatically replacing faces in photographs. ACM Trans. Graph. (TOG) **27**(3) (2008) 39:1–39:8
7. Yang, F., Wang, J., Shechtman, E., Bourdev, L., Metaxas, D.: Expression flow for 3d-aware face component transfer. ACM Trans. Graph. **30**(4) (2011) 60:1–60:10
8. Chai, M., Wang, L., Weng, Y., Yu, Y., Guo, B., Zhou, K.: Single-view hair modeling for portrait manipulation. ACM Trans. Graph. **31**(4) (2012) 116:1–116:8
9. Kemelmacher-Shlizerman, I.: Transfiguring portraits. ACM Trans. Graph. **35**(4) (2016) 94:1–94:8
10. Shu, Z., Hadap, S., Shechtman, E., Sunkavalli, K., Paris, S., Samaras, D.: Portrait lighting transfer using a mass transport approach. ACM Trans. Graph. **37**(1) (2017) 2:1–2:15
11. Fišer, J., Jamriška, O., Simons, D., Shechtman, E., Lu, J., Asente, P., Lukáč, M., Sýkora, D.: Example-based synthesis of stylized facial animations. ACM Trans. Graph. **36**(4) (2017) 155:1–155:11
12. Mosaddegh, S., Simon, L., Jurie, F.: Photorealistic face de-identification by aggregating donors face components. In: Asian Conference on Computer Vision. (2014) 159–174
13. Korshunova, I., Shi, W., Dambre, J., Theis, L.: Fast face-swap using convolutional neural networks. arXiv preprint arXiv:1611.09577 (2016)
14. Hassner, T.: Viewing real-world faces in 3d. In: IEEE International Conference on Computer Vision (ICCV). (2013) 3607–3614
15. McLaughlin, N., Martinez-del Rincon, J., Miller, P.: Data-augmentation for reducing dataset bias in person re-identification. In: IEEE International Conference on Advanced Video and Signal Based Surveillance. (2015) 1–6
16. Masi, I., an Trãn, A.T., Hassner, T., Leksut, J.T., Medioni, G.: Do we really need to collect millions of faces for effective face recognition? In: European Conference on Computer Vision (ECCV). (2016)
17. Nirkin, Y., Masi, I., Tran, A.T., Hassner, T., Medioni, G.: On face segmentation, face swapping, and face perception. arXiv preprint arXiv:1704.06729 (2017)
18. Bao, J., Chen, D., Wen, F., Li, H., Hua, G.: CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training. In: IEEE International Conference on Computer Vision (ICCV). (2017) 2745–2754
19. FakeApp: https://www.fakeapp.org/ (2018)

20. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2016)
21. Amos, B., Ludwiczuk, B., Satyanarayanan, M.: OpenFace: A general-purpose face recognition library with mobile applications. Technical report, CMU School of Computer Science (2016)
22. Wang, Z., Simoncelli, E.P., Bovik, A.C.: Multiscale structural similarity for image quality assessment. In: The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers. (2003) 1398–1402
23. Liu, Z., Shan, Y., Zhang, Z.: Expressive expression mapping with ratio images. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques. (2001) 271–276
24. Leyvand, T., Cohen-Or, D., Dror, G., Lischinski, D.: Data-driven enhancement of facial attractiveness. ACM Trans. Graph. **27**(3) (2008) 38:1–38:9
25. Chai, M., Luo, L., Sunkavalli, K., Carr, N., Hadap, S., Zhou, K.: High-quality hair modeling from a single portrait photo. ACM Trans. Graph. **34**(6) (2015) 204:1–204:10
26. Shu, Z., Yumer, E., Hadap, S., Sunkavalli, K., Shechtman, E., Samaras, D.: Neural face editing with intrinsic image disentangling. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)
27. Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active appearance models. IEEE Tranactions on Pattern Analysis and Machine Intelligence **23**(6) (2001) 681–685
28. Kingma, D.P., Mohamed, S., Rezende, D.J., Welling, M.: Semi-supervised learning with deep generative models. In: Advances in Neural Information Processing Systems. (2014) 3581–3589
29. Odena, A., Olah, C., Shlens, J.: Conditional Image Synthesis With Auxiliary Classifier GANs. arXiv preprint arXiv:1610.09585 (October 2016)
30. 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. arXiv preprint arXiv:1711.09020 (2017)
31. Brock, A., Lim, T., Ritchie, J.M., Weston, N.: Neural photo editing with introspective adversarial networks. arXiv preprint arXiv:1609.07093 (2016)
32. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Trans. Graph. **36**(4) (2017) 107:1–107:14
33. Chen, Z., Nie, S., Wu, T., Healey, C.G.: High Resolution Face Completion with Multiple Controllable Attributes via Fully End-to-End Progressive Generative Adversarial Networks. arXiv preprint arXiv:1801.07632 (2018)
34. 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 (NIPS). Number 27 (2014) 2672–2680
35. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2017)
36. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980 (2014)
37. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid Scene Parsing Network. arXiv preprint arXiv:1612.01105 (2016)
38. King, D.E.: Dlib-ml: A machine learning toolkit. Journal of Machine Learning Research **10** (2009) 1755–1758
39. Levin, A., Zomet, A., Peleg, S., Weiss, Y.: Seamless image stitching in the gradient domain. In: European Conference on Computer Vision (ECCV). (2004) 377–389

40. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: A system for large-scale machine learning. In: Operating Systems Design and Implementation. Volume 16. (2016) 265–283
41. Larsen, A.B.L., Kaae Sønderby, S., Larochelle, H., Winther, O.: Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300 (2015)
42. Dumoulin, V., Belghazi, I., Poole, B., Lamb, A., Arjovsky, M., Mastropietro, O., Courville, A.: Adversarially learned inference. arXiv preprint arXiv:1606.00704 (2016)
43. Rosca, M., Lakshminarayanan, B., Warde-Farley, D., Mohamed, S.: Variational Approaches for Auto-Encoding Generative Adversarial Networks. arXiv preprint arXiv:1706.04987 (2017)
44. Gregor, K., Danihelka, I., Graves, A., Jimenez Rezende, D., Wierstra, D.: DRAW: A Recurrent Neural Network For Image Generation. arXiv preprint arXiv:1502.04623 (2015)
45. Yang, J., Kannan, A., Batra, D., Parikh, D.: LR-GAN: Layered Recursive Generative Adversarial Networks for Image Generation. arXiv preprint arXiv:1703.01560 (2017)
46. Li, P., Hu, Y., Li, Q., He, R., Sun, Z.: Global and Local Consistent Age Generative Adversarial Networks. arXiv preprint arXiv:1801.08390 (2018)
47. Kwak, H., Zhang, B.T.: Generating images part by part with composite generative adversarial networks. arXiv preprint arXiv:1607.05387 (2016)
48. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8) (1997) 1735–1780
49. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive Growing of GANs for Improved Quality, Stability, and Variation. arXiv preprint arXiv:1710.10196 (2017)

Supplementary Materials: RSGAN: Face Swapping and Editing via Region Separation in Latent Spaces

Ryota Natsume[1(✉)], Tatsuya Yatagawa[2], and Shigeo Morishima[3]

Graduate School of Advanced Science and Engineering,
Waseda University, Tokyo, Japan
[1]ryota.natsume.26@gmail.com, [2]tatsy@acm.org, [3]shigeo@waseda.jp

A Table of contents

1. **RSGAN's training algorithm** (Algorithm A1)
2. **Additional results:**
   - Face swapping (Fig. A1)
   - Visual attribute editing (Fig. A2)
   - Random face parts sampling (Fig. A3)
   - Face parts interpolation (Fig. A4)
3. **RSGAN with LSTM**
   - Illustration of the network architecture with LSTM (Fig. A5)
   - Face swapping results with LSTM (Fig. A6)

Algorithm A1 Training pipeline of the proposed RSGAN.

Require: $\lambda_{rec} = 4000$, $\lambda_{KL} = 1$, $\lambda_{adv\text{-}g} = 20$, $\lambda_{adv\text{-}p} = 30$, $\lambda_C = 1.0$, and $\lambda_{GC} = 50$.

while Convergence not reached **do**

Step 1. Compute loss functions:

Sample a batch $(x, x_f, x_h, c) \sim P_{data}$ from the real data.

$z_{x_f} \leftarrow F_{E\text{-}x_f}(x, c),\ z_{x_h} \leftarrow F_{E\text{-}x_h}(x, c)$.

Compute the KL losses, $\mathcal{L}_{KL\text{-}x_f}$ and $\mathcal{L}_{KL\text{-}x_h}$ with Eq. 4.

$z_{c_f} \leftarrow F_{E\text{-}c_f}(c),\ z_{c_h} \leftarrow F_{E\text{-}c_h}(c)$.

Compute the KL losses, $\mathcal{L}_{KL\text{-}c_f}$ and $\mathcal{L}_{KL\text{-}c_h}$ with Eq. 4.

$x'_f \leftarrow F_{D\text{-}f}(z_{x_f}, z_{c_f}),\ x'_h \leftarrow F_{D\text{-}h}(z_{x_h}, z_{c_h})$.

Compute the parts reconstruction losses, $\mathcal{L}_{rec\text{-}f}$ and $\mathcal{L}_{rec\text{-}h}$ with Eq. 1 and Eq. 2, respectively.

$x' \leftarrow G(z_{x_f}, z_{c_f}, z_{x_h}, z_{c_h})$.

Compute the reconstruction loss, $\mathcal{L}_{rec}$ with Eq. 3.

Sample a batch of random vector $\hat{z}_{x_f}$, $\hat{z}_{x_h}$, $\hat{z}_{c_f}$, and $\hat{z}_{c_h}$ from the multivariate standard normal distribution P_z.

$\hat{x}' \leftarrow G(\hat{z}_{x_{x_f}}, \hat{z}_{c_f}, \hat{z}_{x_h}, \hat{z}_{c_h})$.

Compute the adversarial losses, $\mathcal{L}_{adv\text{-}g}$ and $\mathcal{L}_{adv\text{-}p}$ with Eq. 5.

Estimate visual attributes c^*, c' and $\hat{c}'$ using the classifier network.

Compute the classification loss, $\mathcal{L}_C$ with Eq. 6.

Compute the classification losses for the composer network, $\mathcal{L}_{GC}$ with Eq. 7.

Step 2. Update network parameters Θ:

$\mathcal{L}_G \xleftarrow{+} \mathcal{L}_{adv\text{-}g} + \mathcal{L}_{adv\text{-}p} + \lambda_{GC}\mathcal{L}_{GC}$

$\Theta_{F_{E\text{-}x_f}} \xleftarrow{+} -\nabla_{\Theta_{F_{E\text{-}x_f}}}(\lambda_{rec}\mathcal{L}_{rec\text{-}f} + \lambda_{KL}\mathcal{L}_{KL\text{-}x_f} + \mathcal{L}_G)$

$\Theta_{F_{E\text{-}x_h}} \xleftarrow{+} -\nabla_{\Theta_{F_{E\text{-}x_h}}}(\lambda_{rec}\mathcal{L}_{rec\text{-}h} + \lambda_{KL}\mathcal{L}_{KL\text{-}x_h} + \mathcal{L}_G)$

$\Theta_{F_{E\text{-}c_f}} \xleftarrow{+} -\nabla_{\theta_{F_{E\text{-}c_f}}}(\lambda_{rec}\mathcal{L}_{rec\text{-}f} + \lambda_{KL}\mathcal{L}_{KL\text{-}c_f} + \mathcal{L}_G)$

$\Theta_{F_{E\text{-}c_h}} \xleftarrow{+} -\nabla_{\Theta_{F_{E\text{-}c_h}}}(\lambda_{rec}\mathcal{L}_{rec\text{-}h} + \lambda_{KL}\mathcal{L}_{KL\text{-}c_h} + \mathcal{L}_G)$

$\Theta_{F_{D\text{-}f}} \xleftarrow{+} -\nabla_{\Theta_{F_{D\text{-}f}}}(\lambda_{rec}\mathcal{L}_{rec\text{-}f})$

$\Theta_{F_{D\text{-}h}} \xleftarrow{+} -\nabla_{\Theta_{F_{D\text{-}h}}}(\lambda_{rec}\mathcal{L}_{rec\text{-}h})$

$\Theta_G \xleftarrow{+} -\nabla_{\Theta_G}(\lambda_{rec}\mathcal{L}_{rec} + \mathcal{L}_{adv})$

$\Theta_{D_g} \xleftarrow{+} -\nabla_{\Theta_{D_g}}(\lambda_{adv\text{-}g}\mathcal{L}_{adv\text{-}g})$

$\Theta_{D_p} \xleftarrow{+} -\nabla_{\Theta_{D_p}}(\lambda_{adv\text{-}p}\mathcal{L}_{adv\text{-}p})$

$\Theta_C \xleftarrow{+} -\nabla_{\Theta_C}(\lambda_C\mathcal{L}_C)$

end while

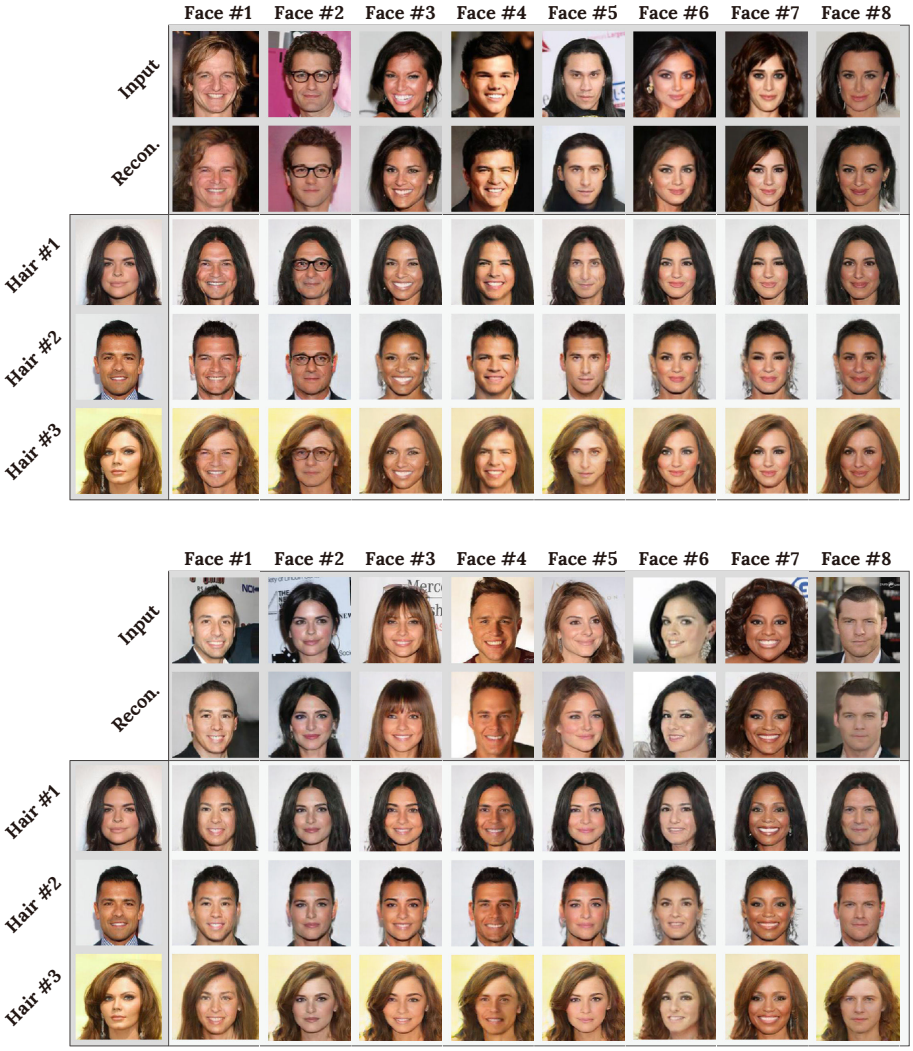

Fig. A1. Additional results for face swapping.

Fig. A2. Additional results for visual attribute editing.

Fig. A3. Additional results for random face parts sampling.

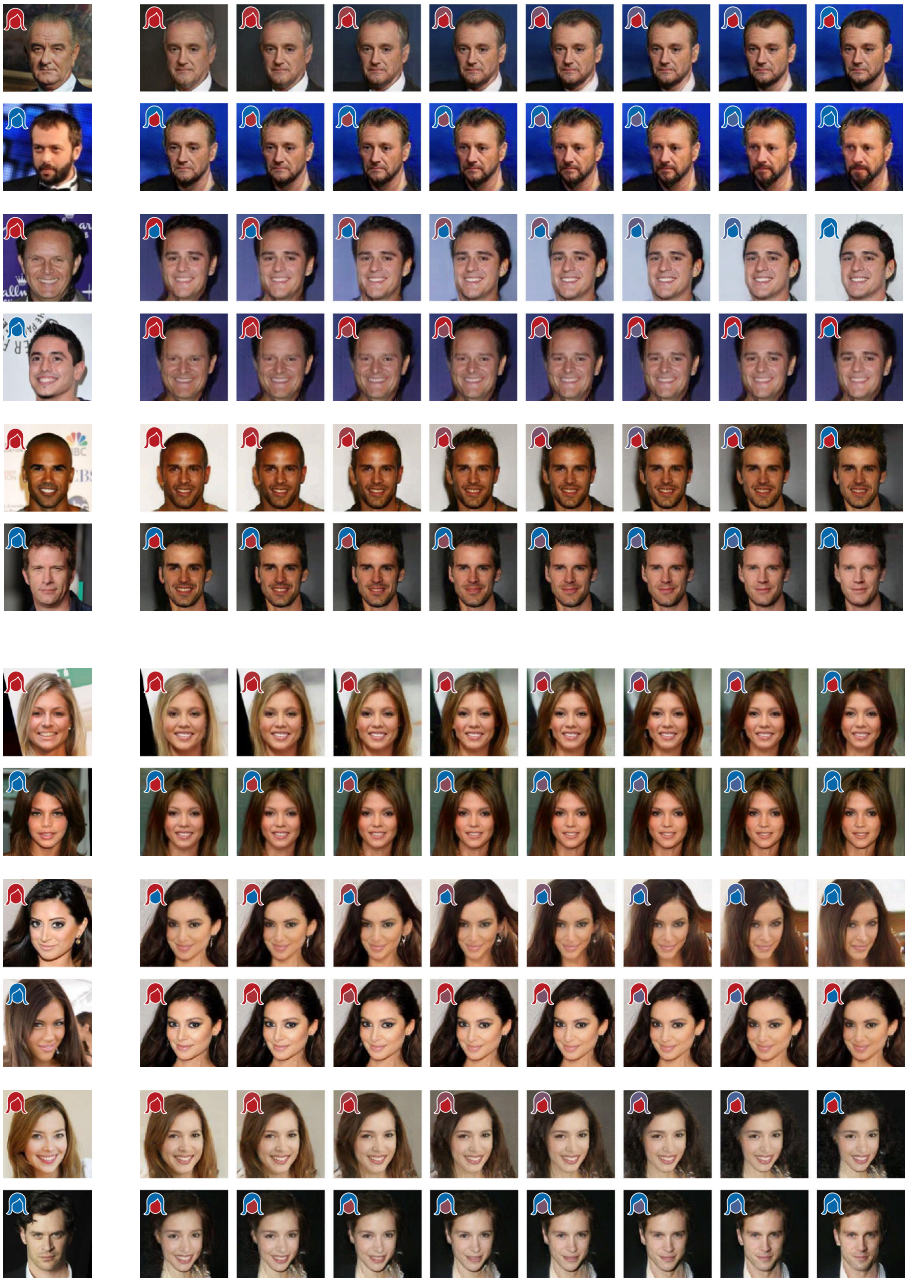

Fig. A4. Additional results for face parts interpolation.

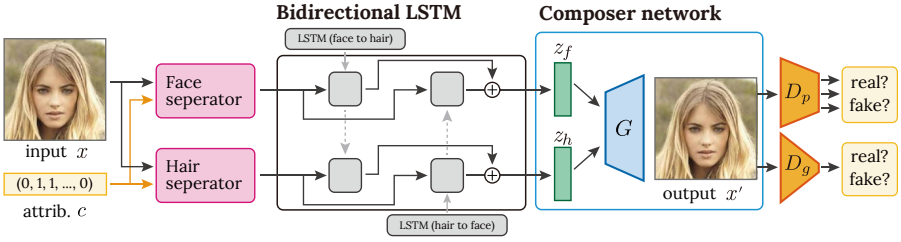

Fig. A5. The network architecture of RSGAN with LSTM. In this architecture bidirectional LSTM is inserted after the two separator.

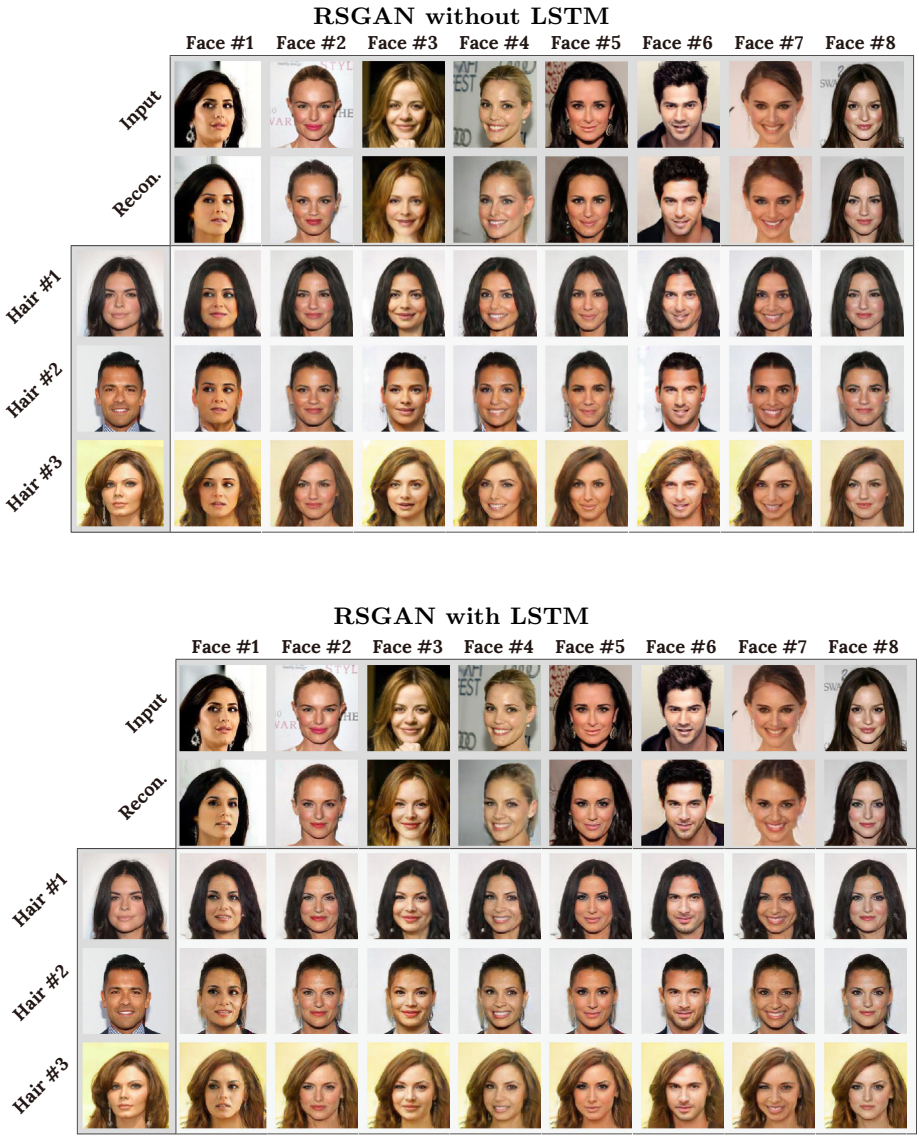

Fig. A6. Comparison of results with and without LSTM. The above image group illustrates the results of RSGAN without LSTM that is the same one as in Fig. 4. The bottom group illustrates the results with LSTM.