A STYLEMAP-BASED GENERATOR FOR REAL-TIME IMAGE PROJECTION AND LOCAL EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

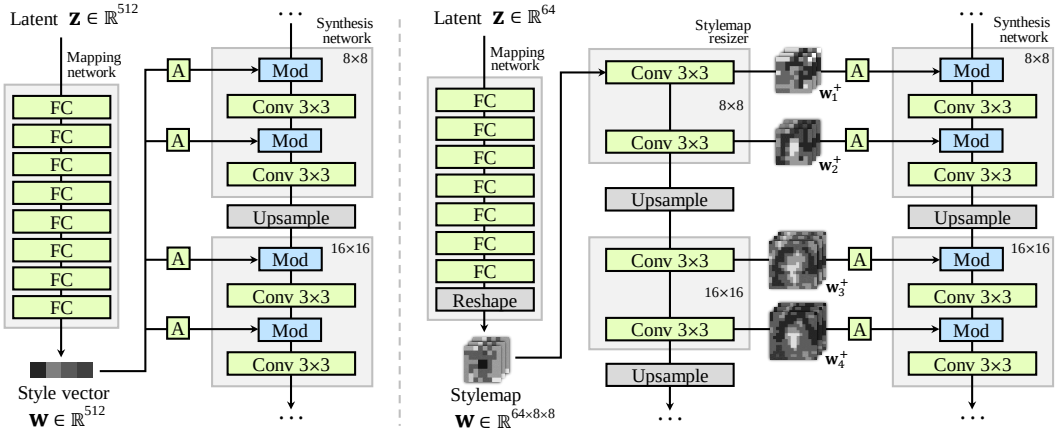
Generative adversarial networks (GANs) have been successful in synthesizing and manipulating synthetic but realistic images from latent vectors. However, it is still challenging for GANs to manipulate real images, especially in real-time. State-of-the-art GAN-based methods for editing real images suffer from time-consuming operations in projecting real images to latent vectors. Alternatively, an encoder can be trained to embed real images to the latent space instantly, but it loses details drastically. We propose StyleMapGAN, which adopts a novel representation of latent space, called stylemap, incorporating spatial dimension into embedding. Because each spatial location in the stylemap contributes to its corresponding region of the generated images, the real-time projection through the encoder becomes accurate as well as editing real images becomes spatially controllable. Experimental results demonstrate that our method significantly outperforms state-of-the-art models in various image manipulation tasks such as local editing and image interpolation. Especially, detailed comparisons show that our local editing method successfully reflects not only the color and texture but also the shape of a reference image while preserving untargeted regions.

1 INTRODUCTION

Generative adversarial networks (GANs) (Goodfellow et al., 2014) have evolved dramatically in recent years, enabling high-fidelity image synthesis with models which are learned directly from data (Brock et al., 2019; Karras et al., 2019; 2020). Recent studies have shown that GANs naturally learn to encode rich semantics within the latent space, thus changing the latent code leads to manipulating the corresponding attributes of the output images (Jahanian et al., 2020; Shen et al., 2020; Härkönen et al., 2020; Goetschalckx et al., 2019; Shen & Zhou, 2020; Alharbi & Wonka, 2020). However, it is still challenging to apply these manipulations to real images, since the GAN itself lacks an inverse mapping from an image back to its corresponding latent code.

One promising approach for manipulating real images is image-to-image translation (Isola et al., 2017; Zhu et al., 2017; Choi et al., 2018), where the model learns to directly synthesize an output image given a user's input. However, these methods require pre-defined tasks and heavy supervision (*e.g.*, input-output pairs, class labels) for training, and also limit the user controllability at inference time. Another approach is to utilize pretrained GAN models, by directly optimizing the latent code for an individual image (Abdal et al., 2019; Zhu et al., 2016; Ma et al., 2018; Noguchi & Harada, 2019). However, even on high-end GPUs, it requires minutes of computation for each target image, and it does not guarantee that the optimized code would be placed in the original latent space of GAN.

A more practical approach is to train an extra encoder which learns to project an image into its corresponding latent code (Zhu et al., 2020a; Perarnau et al., 2016; Luo et al., 2017). Although this approach enables real-time projection in a single feed-forward manner, it suffers from the low fidelity of the projected image (*i.e.*, losing details of the target image). We attribute this limitation to the absence of spatial dimensions in the latent space. Without the spatial dimensions, an encoder compresses the local semantics of an image into a vector in an entangled manner, making it difficult to reconstruct the image (*e.g.*, vector-based or low-resolution bottleneck layer is not capable of producing high-frequency details (Lample et al., 2017; Chang et al., 2018)).
As a solution to such problems, we propose StyleMapGAN which exploits *stylemap*, a novel representation of the latent space. Our key idea is simple. Instead of learning a vector-based latent repre-



(a) Style-based generator (b) StyleMap-based generator

Figure 1: While a traditional mapping network produces style vectors to control feature maps, we create a stylemap with spatial dimensions, which not only makes the projection of a real image much more effective at inference, but also enables local editing. The style map $\mathbf{w}$ is resized to $\mathbf{w}^+$ through convolutional layers to match the spatial resolution of each feature in the synthesis network. Here "A" stands for a learned affine transform, and "Mod" indicates modulation consisting of element-wise multiplication and addition.

sentation, we utilize a tensor with explicit spatial dimensions. Our proposed representation benefits from its spatial dimensions, enabling GANs to easily encode the local semantics of images into the latent space. This property allows an encoder to effectively project an image into the latent space, thus providing high-fidelity and real-time projection. In addition, our method offers a new capability to edit specific regions of an image by manipulating the matching positions of the stylemap.

We demonstrate, on multiple datasets, that our stylemap indeed substantially enhances the projection quality compared to the traditional vector-based latent representation (Section 3.2). Furthermore, we show the advantage of our method over state-of-the art methods on image projection, interpolation, and local editing (Section 3.3 & Section 3.4). Finally, we show that our method can transplant regions even when the regions are not aligned between one image and another (Section 3.5). We will make our code and pretrained models publicly available for research community.

2 StyleMapGAN

Our goal is to accurately project images to a latent space with an encoder in real-time and to locally manipulate images on the latent space. We propose StyleMapGAN which adopts *stylemap*, a novel representation of the intermediate latent space with spatial dimensions. It allows accurate reconstruction with the encoder by alleviating the spatial discrepancy between images and the latent space which has been causing the encoder to lose details. Furthermore, local changes in the stylemap lead to local editing of images thanks to the explicit spatial correspondence between the stylemap and images. Section 2.1 explains how we design the mapping network and the synthesis network to incorporate the stylemap. Section 2.2 describes our procedure for the image-to-latent projection and the local editing.

2.1 Stylemap-based generator

Figure 1 compares the traditional style-based generator (Karras et al., 2019) and our stylemap-based generator. We propose to incorporate a stylemap instead of a style vector and to replace AdaIN operations with spatially adaptive operations. The stylemap has spatial dimensions as opposed to the style vector, thus can represent different styles across spatial locations. Accordingly, we revise SPADE (Park et al., 2019b) which modulates feature maps with spatially varying values.

Since the feature maps in the synthesis network grow larger as getting closer to the output image, we introduce a stylemap resizer, which consists of convolutions and upsampling, to match the resolutions of stylemaps with the feature maps. The stylemap resizer not only resizes the stylemap,

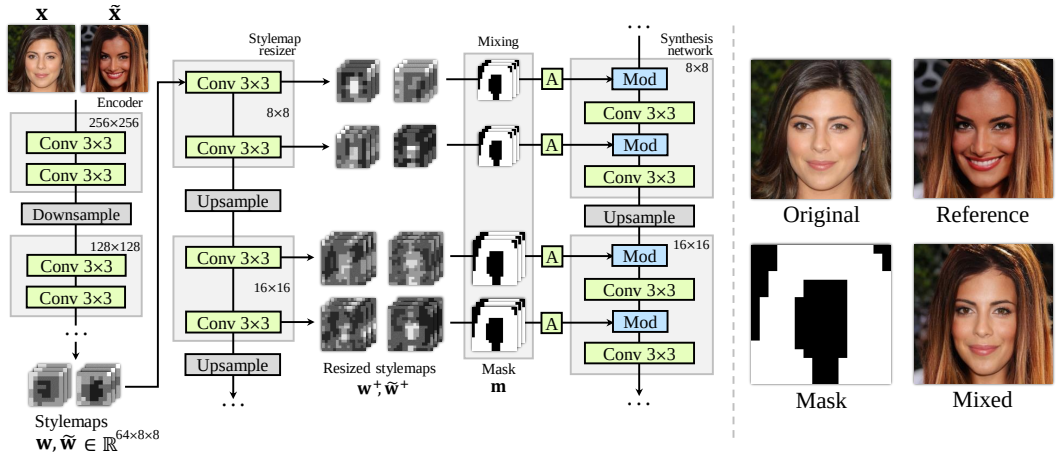


Figure 2: Our local editing starts with a learned encoder for fast image-to-stylemap projection. We estimate the stylemaps $\mathbf{w}$ and $\widetilde{\mathbf{w}}$ of the original $\mathbf{x}$ and the reference $\widetilde{\mathbf{x}}$, and transform them to multiple resolutions through the learned stylemap resizer. For each resolution, we calculate the convex combination of the two stylemaps using the user-defined binary mask $\mathbf{m}$. Finally, the learned generator produces the output using the spatially-mixed stylemaps. The right one shows an example generated using our method.

but also transforms them with learned convolutions to convey more detailed and structured styles. Figure 1 shows examples of changes of resized stylemaps across layers.

Then, the affine transform A produces parameters for the modulation regarding the resized stylemaps. The modulation operation of the i-th layer in the synthesis network is as follows:

$$h_{i+1} = \left(\gamma_i \odot \frac{h_i - \mu_i}{\sigma_i}\right) \oplus \beta_i \tag{1}$$

where $\mu_i, \sigma_i \in \mathbb{R}$ are the mean and standard deviation of activations $h_i \in \mathbb{R}^{C_i \times H_i \times W_i}$ of the layer, respectively. $\gamma_i, \beta_i \in \mathbb{R}^{C_i \times H_i \times W_i}$ are modulation parameters. $\odot$ and $\oplus$ are element-wise multiplication and addition with broadcasting, respectively.

We use layer normalization (Ba et al., 2016) instead of instance normalization and find it helpful to resolve droplet artifacts. In addition, we remove per-pixel noise which is an extra source of variation because it makes the projection complicated. Instead, β plays the similar role. Note that weight modulation (Karras et al., 2020) cannot be applied to spatially varying modulation because weights are shared across all locations in a convolution. Other details about the networks such as a design choice of mapping network are given in Appendix A and B.

2.2 Real image projection and local editing

Since the stylemap eases the spatial discrepancy between images and the latent space, we train an encoder to project real images into its corresponding stylemaps, which accurately reconstructs the images through the generator. The encoder is jointly trained with the generator and the discriminator. More training details are described in Appendix A. Now we have access to the accurate projection of images to the style space which is essential to latent-based editing. Furthermore, local changes in the stylemap leads to natural local editing of images on the learned semantic manifold. Especially, we design a procedure for local transplantation which now becomes feasible.

The goal of local editing is to transplant some part of a reference image to an original image with respect to a mask which indicates the region to be modified. We project the original image and the reference image through the encoder to obtain stylemaps $\mathbf{w}$ and $\widetilde{\mathbf{w}}$, respectively. In general, the mask is finer than 8×8, we blend the stylemaps on $\mathbf{w}^+$ space to achieve detailed manipulation. The edited i-th resized stylemap $\ddot{\mathbf{w}}^+$ is an alpha blending of $\mathbf{w}^+$ and $\widetilde{\mathbf{w}}^+$:

$$\ddot{\mathbf{w}}_i^+ = \mathbf{m}_i \odot \widetilde{\mathbf{w}}_i^+ + (1 - \mathbf{m}_i) \odot \mathbf{w}_i{}^+ \tag{2}$$

where i-th resized mask $\mathbf{m}_i$ is shrunk by max pooling. If the mask's shape aligns with the 8×8 stylemap, we can do the same alpha blending on the $\mathbf{w}$ space instead of the $\mathbf{w}^+$ space. Note that the mask can be in any shape allowing usage of semantic segmentation methods or user scribbles. On the contrary to SPADE (Park et al., 2019b) or SEAN (Zhu et al., 2020b), even coarse masks as coarse as 8×8 produces plausible images so that the burden for user to provide detailed masks is lifted. This operation can be further revised for unidentical masks of the two images (Section 3.5).

3 EXPERIMENTS

Our proposed method efficiently projects images into the style space in real-time and effectively manipulate specific regions of real images. We first describe our experimental setup (Section 3.1) and show how the proposed spatial dimensions of stylemap affect the image projection and generation quality (Section 3.2). We then compare our method with the state-of-the art methods on real image projection (Section 3.3) and local editing (Section 3.4). We finally show a more flexible editing scenario and usefulness of our proposed method (Section 3.5). Implementation details are described in Appendix A.

3.1 EXPERIMENTAL SETUP

Datasets and protocols. For evaluation, we train our model on CelebA-HQ (Karras et al., 2018) and AFHQ (Choi et al., 2020), both at resolution of 256×256. We use 500 images for validation, another 500 images for testing, and the rest for training. Because the optimization methods take an extremely long time, we limited the test set to 500 images. When we compute Fréchet inception distance (FID), the numbers of generated samples are matched to the training set. Reconstruction errors are measured with all test images. For FID_{lerp}, we choose random numbers between 0 and 1 for 500 random pairs of images from the test set to synthesize 500 interpolated images and compute FID between those and the test set. For local editing comparison, 250 pairs of test images in CelebA-HQ are composed with ten semantic masks (*e.g.*, background, hair) (Lee et al., 2020) to produce 2500 images. For local editing on AFHQ, masks are randomly chosen between horizontal and vertical half-and-half masks to produce 250 images.

Baselines. We compare our method against recent methods. For image projection, StyleGAN2 (Karras et al., 2020) and Image2StyleGAN (Abdal et al., 2019) infer the per-layer style vectors (analogous to our $\mathbf{w}^+$) via iterative optimization. In-DomainGAN (Zhu et al., 2020a) relies on optimization preceded by initialization using a domain-guided encoder. SEAN (Zhu et al., 2020b) also includes an encoder but it requires semantic segmentation masks for training. Structured noise (Alharbi & Wonka, 2020) adds input tensor with spatial dimensions to the synthesis network of StyleGAN but it does not enhance the rest of the network where the style vector still plays an important role. Editing in style (Collins et al., 2020) tries to find local semantics in the style vector.

3.2 ANALYSIS OF OUR METHOD

To manipulate an image using a generative model, we first need to accurately project the image into its latent space. In Table 1, we vary the spatial resolution of stylemap and compare the performance of reconstruction and generation. As the spatial resolution increases, the reconstruction accuracy improves significantly. It demonstrates that our stylemap with spatial dimensions is highly effective for image projection. FID varies differently across datasets, possibly due to different contextual relationship between locations for generation. Note that our method with spatial resolution accurately preserves small details, *e.g.*, the eyes are not blurred.

We next evaluate the effect of the stylemap's resolution in editing scenarios, mixing specific parts of one image and another. Figure 3 shows that the 8×8 stylemap synthesizes the most accurate and seamlessly blended images. We see that when the spatial resolution is higher than 8×8, the edited parts are easily detected. We suppose that too large stylemap harms contextual relationship across locations which is essential for realistic images. Considering the editing quality, we choose the 8×8 resolution as our best model and use it consistently for all subsequent experiments.

Method	Style resolution	CelebA-HQ MSE	CelebA-HQ LPIPS	CelebA-HQ FID	AFHQ MSE	AFHQ LPIPS	AFHQ FID
StyleGAN2	1×1	0.089	0.428	4.97	0.139	0.539	8.59
StyleMapGAN	4×4	0.062	0.351	**4.03**	0.070	0.394	14.82
StyleMapGAN	8×8	0.024	0.242	4.92	0.037	0.304	11.10
StyleMapGAN	16×16	0.010	0.146	4.71	0.016	0.183	**6.71**
StyleMapGAN	32×32	**0.004**	**0.076**	7.18	**0.006**	**0.090**	7.87

Table 1: Comparison on reconstruction and generation quality across different resolutions of the stylemap. Mean squared error (MSE) is in [-1, 1] scale and learned perceptual image patch similarity (LPIPS) measures reconstruction accuracy with the encoder. Fréchet Inception Distance (FID) measures the quality of randomly generated samples from the standard Gaussian distribution. The higher resolution helps accurate reconstruction, validating the effectiveness of stylemap. We observe that 8×8 stylemap already provides accurate enough reconstruction and accuracy gain and afterward improvements get visually negligible. Although FID varies differently across datasets, possibly due to the different contextual relationship between locations for generation, the stylemap does not seriously harm quality of the images.

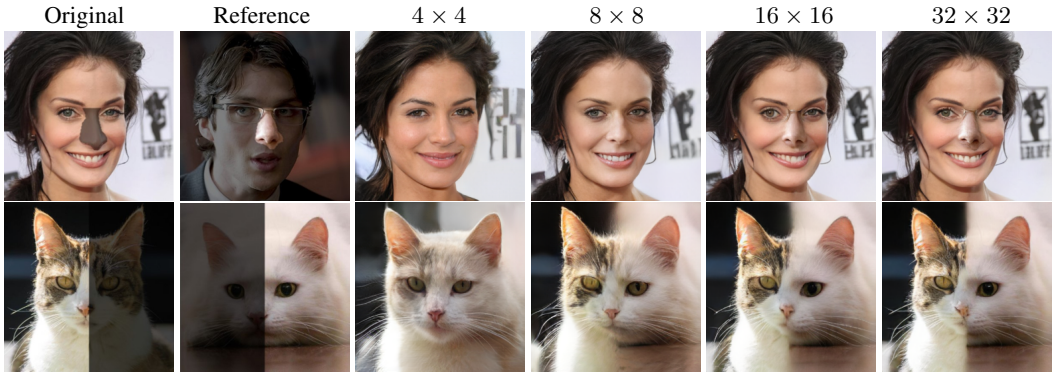
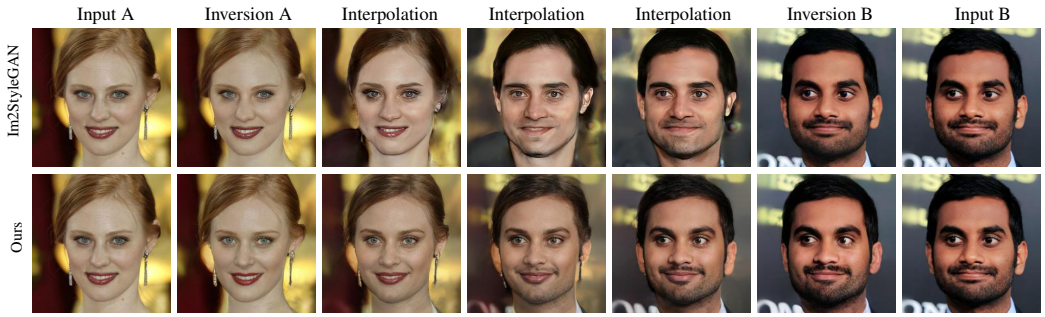


Figure 3: Local editing comparison across different resolutions of the stylemap. Regions to be discarded are faded on the original and the reference images. 4×4 suffers from poor reconstruction. Resolutions greater than or equal to 16×16 result in too heterogeneous images. 8×8 resolution shows the acceptable reconstruction and natural integration.

3.3 Real image projection

In Table 2, we compare our approach with the state-of-the art methods for real image projection. For both datasets, StyleMapGAN achieves better reconstruction quality (MSE & LPIPS) than all competitors. Also, it achieves the best FID, which implicitly shows that our manipulation on the style space leads to the most realistic images. Importantly, our method runs 100× faster than the optimization-based baselines since a single feedforward pass provides accurate projection thanks to



Method	Runtime (s)	CelebA-HQ MSE	CelebA-HQ LPIPS	CelebA-HQ FID_{lerp}	AFHQ MSE	AFHQ LPIPS	AFHQ FID_{lerp}
StyleGAN2	80.4	0.079	0.247	55.38	0.091	0.288	30.65
Image2StyleGAN	192.5	**0.009**	**0.203**	55.22	**0.018**	**0.282**	62.99
Structured Noise	64.4	0.097	0.256	54.71	0.144	0.332	54.61
In-DomainGAN	6.8	0.052	0.340	49.87	0.077	0.414	35.07
SEAN	0.146	0.064	0.334	44.08	N/A	N/A	N/A
StyleMapGAN	**0.080**	**0.024**	**0.242**	**36.71**	**0.037**	**0.304**	**26.97**

Table 2: Comparison with the baselines for real image projection. Runtime covers the end-to-end interval of projection and generation in seconds. Protocols for MSE and LPIPS are the same with Table 1. FID_{lerp} measures quality of the images interpolated on the style space as a proxy for potential quality of the manipulated images. Our method allows real-time manipulation of real images while achieving the best reconstruction accuracy *and* the best quality of the interpolated images. Although Image2StyleGAN produces the smallest reconstruction error, it suffers from minutes of runtime and poor interpolation quality which are not suitable for a practical editing. Its flaws can be found in the figure: deviating identity, odd changes on neck and background, and sudden changes on eyes. SEAN is not applicable to AFHQ due to no existence of segmentation masks for training. The horizontal line between methods separates optimization-based methods and encoder-based methods.

Method	Runtime	CelebA-HQ AP	CelebA-HQ MSE_{src}	CelebA-HQ MSE_{ref}	AFHQ AP	AFHQ MSE_{src}	AFHQ MSE_{ref}
Structured Noise	64.4	99.16	0.105	0.395	99.88	0.137	0.444
Editing in Style	55.6	98.34	0.094	0.321	99.52	0.130	0.417
In-DomainGAN	6.8	98.72	0.164	**0.015**	99.59	0.172	**0.028**
SEAN	0.155	90.41	0.067	0.141	N/A	N/A	N/A
StyleMapGAN (Ours)	**0.099**	**83.60**	**0.039**	**0.105**	**98.66**	**0.050**	**0.050**

Table 3: Comparison with the baselines for local image editing. Average precision (AP) is measured with the binary classifier trained on real and fake images (Wang et al., 2020). The lower AP indicates that manipulated images are more indistinguishable from real images. MSE_{src} and MSE_{ref} measure error from the source image outside the mask and from the reference image inside the mask, respectively. Compared with the baselines, our method seamlessly composes the two images giving better reconstructions.

the stylemap, which is measured in a single GPU. SEAN also runs with a single feedforward pass, but it requires ground-truth segmentation masks for training which is a severe drawback for practical uses. Image2StyleGAN fails to meet requirements for editing in that it produces spurious artifacts in latent interpolation (FID_{lerp} and figures) and suffers from minutes of runtime.

3.4 Local Editing

We evaluate local editing performance regarding three aspects: detectability, faithfulness to the reference image in the mask and preservation of the source image outside the mask. Figure 4 and Figure

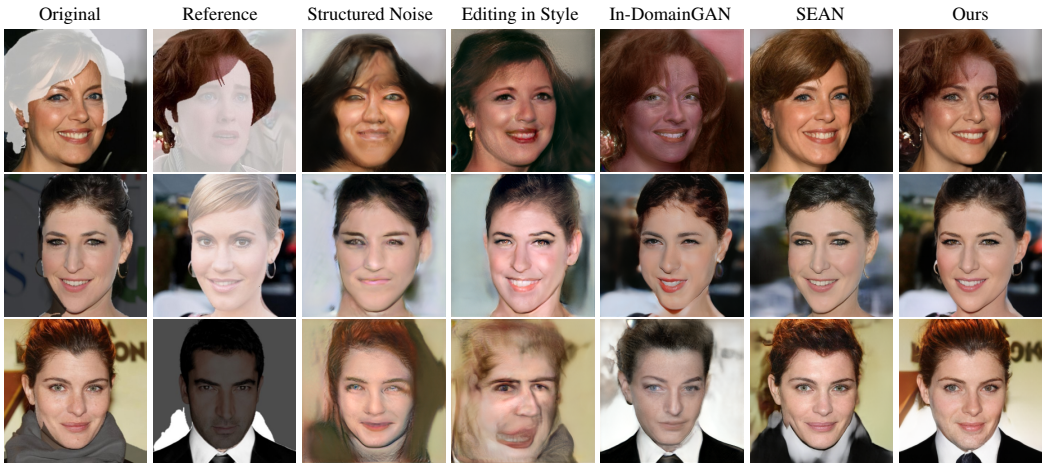


Figure 4: Local editing comparison on CelebA-HQ. The first two baselines even fail to reconstruct untouched region. In-Domain GAN inversion poorly blends the two images, leaking colors to faces, hair, or background, respectively. SEAN locally transfers coarse structure and color but significantly loses details. Ours seamlessly transplants the target region from the reference to the original.

Figure 5: Local editing comparison on AFHQ. Each row blends the two images with vertical, horizontal and custom masks, respectively. Our method seamlessly composes two species with well-preserved details resulting in non-existing creatures, while others tend to lean towards one species.

5 visually demonstrate that our method seamlessly composes the two images while others struggle. Since there is no metrics for evaluating the last two aspects, we propose two quantitative metrics: MSE_{src} and MSE_{ref}. Table 3 shows that the results from our method are the hardest for the classifier to detect and both source and reference images are best reflected. Note that MSEs are not the sole measures but AP should be considered together.

3.5 Unaligned transplantation

Here, we demonstrate more flexible use case, unaligned transplantation, showing that our local editing does not require the masks on the original and the reference images to be aligned. We project the images to the stylemaps and and replace the designated region of the original stylemap with the crop

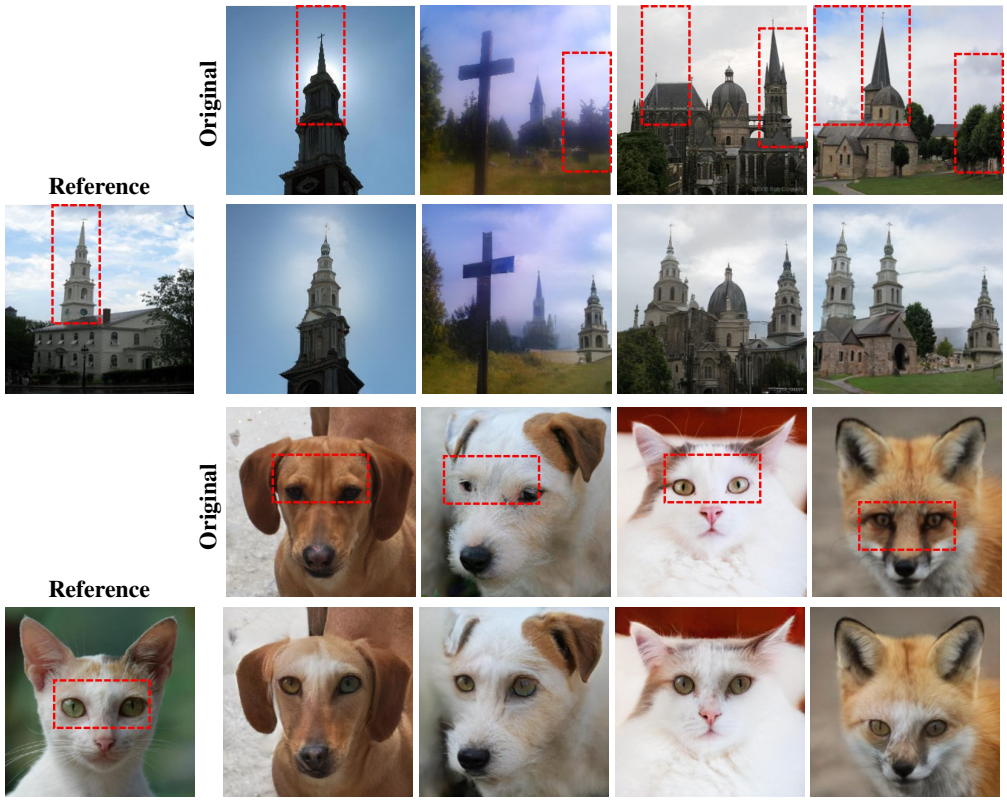


Figure 6: Examples of unaligned transplantation. StyleMapGAN allows composing arbitrary number of any regions. Note that the size of the tower and eyes are automatically adjusted regarding the surroundings. The masks are specified on 8×8 grid and the stylemaps are blended on $\mathbf{w}$ space.

of the reference stylemap even though they are on the different locations. Users can specify what to replace. Figure 6 shows examples.

4 Discussion and Conclusion

Invertibility of GANs has been essential for editing real images with unconditional GAN models at a practical time and it has not been properly answered yet. To achieve this goal, we propose StyleMapGAN, which introduces explicit spatial dimensions to the latent space, called a stylemap. We show, through extensive evaluation, that our method based on the stylemap has a number of advantages over prior approaches, in that it can accurately project real images in real-time, into the latent space, and synthesize high-quality output images by both interpolation and local editing. The proposed latent representation is simple, general, and can be easily integrated into existing GAN models (*e.g.*, StyleGAN) with wide range of network designs and data modality. We believe that improving fidelity by applying our latent representation to other methods such as conditional GANs (*e.g.*, BigGAN) or variational autoencoders (Kingma & Welling, 2013) would be an exciting future work.

References

Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *CVPR*, 2019.

Yazeed Alharbi and Peter Wonka. Disentangled image generation through structured noise injection. In *CVPR*, 2020.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. In *arXiv preprint*, 2016.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

Huiwen Chang, Jingwan Lu, Fisher Yu, and Adam Finkelstein. Pairedcyclegan: Asymmetric style transfer for applying and removing makeup. In *CVPR*, 2018.

Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *ICCV*, 2017.

Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *CVPR*, 2020.

Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *CVPR*, 2020.

Antonia Creswell and Anil Anthony Bharath. Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 2018.

Lore Goetschalckx, Alex Andonian, Aude Oliva, and Phillip Isola. Ganalyze: Toward visual definitions of cognitive image properties. In *ICCV*, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *NeurIPS*, 2014.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint*, 2020.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 2006.

Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial nets. In *CVPR*, 2017.

Ali Jahanian, Lucy Chai, and Phillip Isola. On the"steerability" of generative adversarial networks. In *ICLR*, 2020.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

Hyunsu Kim, Ho Young Jhoo, Eunhyeok Park, and Sungjoo Yoo. Tag2pix: Line art colorization using text tag with secat and changing loss. In *ICCV*, 2019.

Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwang Hee Lee. U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. In *ICLR*, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint*, 2013.

Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato. Fader networks: Manipulating images by sliding attributes. In *NeurIPS*, 2017.

Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *ECCV*, 2016.

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *CVPR*, 2020.

Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Kumar Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *ECCV*, 2018.

Zachary C Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint*, 2017.

Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *ECCV*, 2018.

Junyu Luo, Yong Xu, Chenwei Tang, and Jiancheng Lv. Learning inverse mapping by autoencoder based generative adversarial nets. In *ICNIP*, 2017.

Fangchang Ma, Ulas Ayaz, and Sertac Karaman. Invertibility of convolutional generative networks from partial measurements. In *NeurIPS*, 2018.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In *ICML*, 2018.

Atsuhiro Noguchi and Tatsuya Harada. Image generation from small datasets via batch statistics adaptation. In *ICCV*, 2019.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019a.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019b.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

Guim Perarnau, Joost Van De Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint*, 2016.

Tiziano Portenier, Qiyang Hu, Attila Szabo, Siavash Arjomand Bigdeli, Paolo Favaro, and Matthias Zwicker. Faceshop: Deep sketch-based face image editing. *arXiv preprint*, 2018.

Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. *arXiv preprint*, 2020.

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020.

Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. Cnn-generated images are surprisingly easy to spot... for now. In *CVPR*, 2020.

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018.

Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *CVPR*, 2017.

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.

Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *ECCV*, 2020a.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2016.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *ICCV*, 2017.

Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Sean: Image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020b.

A IMPLEMENTATION DETAILS

Architecture. We follow StyleGAN2 (Karras et al., 2020) regarding the discriminator architecture and the feature map counts in the convolution layers of the synthesis network. Our mapping network is an MLP with eight fully connected layers followed by a reshape layer. The channel sizes are 64 except the last being 4,096. Our encoder adopts the discriminator architecture until the 8×8 layer and without minibatch discrimination.

Training. We jointly train the generator, the encoder and the discriminator. It is simpler and leads to more stable training and higher performance than separately training the adversarial networks and the encoder. For the rest, we mostly follow the settings of StyleGAN2, *e.g.*, the discriminator architecture, Adam optimizer with the same hyperparameters, exponential moving average of the generator and the encoder, leaky ReLU, equalized learning rate for all layers, random horizontal flip for augmentation, and reducing the learning rate by two orders of magnitude for the mapping network. Our code is based on unofficial PyTorch implementation of StyleGAN2[1]. All StyleMapGAN variants on comparison are trained for two weeks on 5M images with two Tesla V100 GPUs using minibatch size of 16. We note that most cases keep slowly improving until 10M images. Our code will be publicly available online for reproduction[2].

Losses. Here we use G, D and E as short forms of the generator, the discriminator and the Encoder. The adversarial loss for G and D are non-saturating loss (Goodfellow et al., 2014). R_1 regularization term (Mescheder et al., 2018) is computed every 16 steps for D. G and E are trained with image reconstruction loss (MSE) and perceptual loss (Johnson et al., 2016). Domain-guiding loss (Zhu et al., 2020a) is applied to all networks. Table 4 summarizes the losses.

Loss	Generator	Discriminator	Encoder
Adversarial loss	✓	✓	
R_1 regularization		✓	
Latent reconstruction			✓
Image reconstruction	✓		✓
Perceptual loss	✓		✓
Domain-guided loss	✓	✓	✓

Table 4: Losses for training each network.

B MAPPING NETWORK DESIGN FOR THE STYLEMAP

There are several choices when designing a mapping network. We can easily think of convolutional layers due to the spatial dimensions of the stylemap. Alternatively, we can remove the mapping network so that our method does not generate images from the standard Gaussian distribution, and

[1] https://github.com/rosinality/stylegan2-pytorch

[2] http://publicurl.com

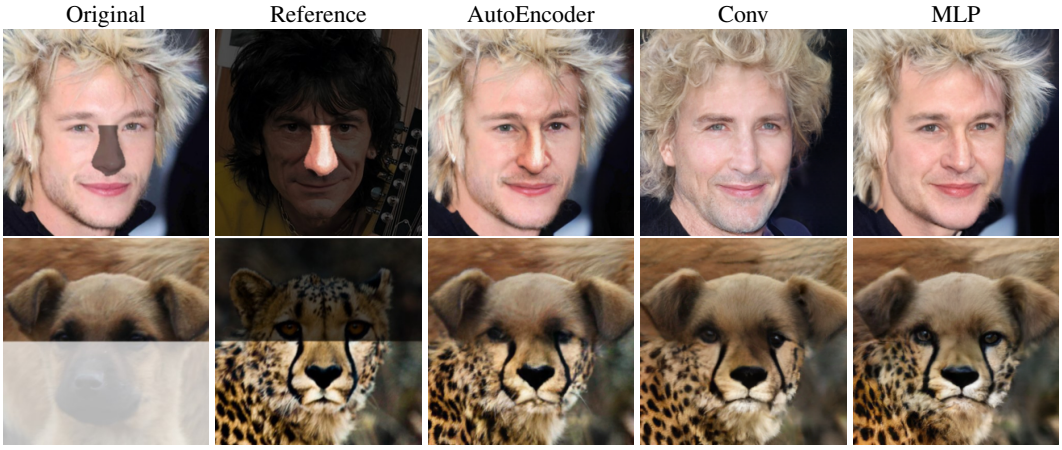


Figure 7: Local editing comparison across different mapping network structures in the generator. The autoencoder method without a mapping network is most unnatural in a modified image. The mapping network with convolutional layers has more natural results than the autoencoder. Nevertheless, due to its bad reconstruction quality, it suffers from preserving the characteristics of original images. Our MLP mapping network is natural in local editing and preserves the original image well. Also, even if the eye part is not properly inserted like the animal image, it naturally creates it.

uses only real images for training like autoencoder (Hinton & Salakhutdinov, 2006). As shown in Figure 3, autoencoder fails to produce realistic images using the projected stylemap. It seems to copy and paste between two images on RGB space. We give continuous input to the generator from the standard Gaussian distribution using a mapping network, letting the network generate seamless images in image space. However, the autoencoder only gives discrete input, which is projected from the encoder. On the other hand, the mapping network with convolutional layers often struggles in reconstruction so that the edited results images are quite different from the original images. We assume that there is such a limit because the convolutional layer's mapping is bounded to the local area. In MLP, each weight and input are fully-connected so that it can make a more plausible latent space.

C Related work

C.1 Latent-based image editing

There are active studies (Abdal et al., 2019; Collins et al., 2020; Zhu et al., 2020a) on image editing using latent vector arithmetic where well-trained GANs (Karras et al., 2019; 2020) are adopted for real-world applications. These studies aim to find a latent vector to reconstruct an original image. In general, there are two approaches to embed images into latent vectors, learning and optimization-based ones. The learning-based approach (Zhu et al., 2020a; Perarnau et al., 2016; Zhu et al., 2016) trains an encoder that maps a given image to a latent vector. This method has a potential of projecting an image in real time. However, the existing methods suffer from low quality of the reconstructed images, which indicates the difficulty of embedding real images. The optimization-based approach (Creswell & Bharath, 2018; Lipton & Tripathi, 2017; Ma et al., 2018; Abdal et al., 2019), given an input image, aims at optimizing the latent vector to minimize the pixel-wise reconstruction loss. Though it is not feasible to project images in real time due to its iterative nature, it exhibits high quality of the reconstructed images while enabling edits include global changes in semantic attribute, e.g. smiling, beard, etc. Compared with these approaches, our StyleMapGAN can project images in real time while offering high quality of reconstruction images.

C.2 Local editing

Several methods (Collins et al., 2020; Alharbi & Wonka, 2020; Zhu et al., 2020b) tackle locally editing specific parts (e.g., nose, background) as opposed to the most GAN-based image editing

methods modifying global appearance. Editing in style (Collins et al., 2020) tries to identify components in the style vector which are responsible for specific parts and achieves local editing. It requires preceding component analysis and the correspondence between the components and regions is lose. Structured noise injection (Alharbi & Wonka, 2020) replaces the learned constant from StyleGAN with an input tensor which has spatial dimensions and is a combination of local and global codes. Though it learns some sense of spatial disentanglement, its applicability is limited due to the separate source of variation, the style vector. These two methods are limited to editing fake images while editing real images with them requires projecting the images to the latent space. SEAN (Zhu et al., 2020b) facilitates editing real images by encoding images into the per-region style codes and manipulating them. However, per-region style codes do not capture details and it requires semantic segmentation masks for training. On the other hand, our StyleMapGAN captures and controls fine details of images with a stylemap which has explicit spatial correspondence with images. Our method does not require segmentation masks for training.

C.3 CONDITIONAL IMAGE SYNTHESIS

Conditional image synthesis models, such as image-to-image translation (Isola et al., 2017; Zhu et al., 2017; Kim et al., 2020), learn to synthesize an output image given an original image. Thanks to this framework, many applications have been successfully built, including colorization (Kim et al., 2019; Larsson et al., 2016; Zhang et al., 2016), image inpainting (Liu et al., 2018; Pathak et al., 2016; Yang et al., 2017), semantic image synthesis and editing (Wang et al., 2018; Chen & Koltun, 2017; Park et al., 2019a; Portenier et al., 2018). Recent models extend it to multi-domain and multi-modal (Huang et al., 2018; Lee et al., 2018; Choi et al., 2020). Image-to-image translation and local edit have been separately studied since they target different objectives, *i.e.*, regarding global and local levels of detail in image generation. However, our method can be applied to the both tasks by semantic manipulation of stylemap for image-to-image translation and local manipulation of stylemap. For example, our StyleMapGAN can make only the eyes laugh or the mouth laugh via local editing as well as change the domain of generated image via global semantic manipulation.