SEAN: Image Synthesis with Semantic Region-Adaptive Normalization

Peihao Zhu[1] Rameen Abdal[1] Yipeng Qin[2] Peter Wonka[1]

[1]KAUST [2]Cardiff University

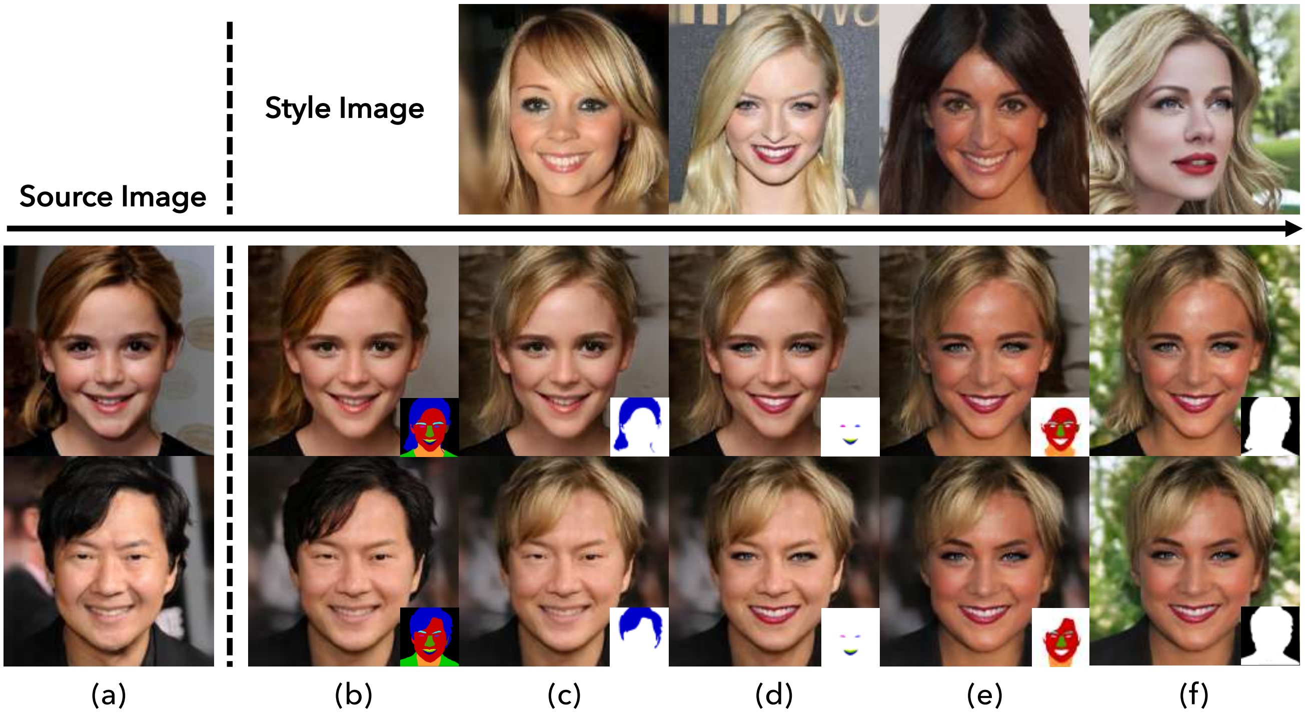


Figure 1: Face image editing controlled via style images and segmentation masks. a) source images. b) reconstruction of the source image; segmentation mask shown as small inset. c - f) four separate edits; we show the image that provides new style information on top and show the part of the segmentation mask that gets edited as small inset. The results of the successive edits are shown in row two and three. The four edits change hair, mouth and eyes, skin tone, and background, respectively.

Abstract

We propose semantic region-adaptive normalization (SEAN), a simple but effective building block for Generative Adversarial Networks conditioned on segmentation masks that describe the semantic regions in the desired output image. Using SEAN normalization, we can build a network architecture that can control the style of each semantic region individually, e.g., we can specify one style reference image per region. SEAN is better suited to encode, transfer, and synthesize style than the best previous method in terms of reconstruction quality, variability, and visual quality. We evaluate SEAN on multiple datasets and report better quantitative metrics (e.g. FID, PSNR) than the current state of the art. SEAN also pushes the frontier of interactive image editing. We can interactively edit images by changing segmentation masks or the style for any given region. We can also interpolate styles from two reference images per region. Code: https://github.com/ZPdesu/SEAN.

1. Introduction

In this paper we tackle the problem of synthetic image generation using conditional generative adversarial networks (cGANs). Specifically, we would like to control the layout of the generated image using a segmentation mask

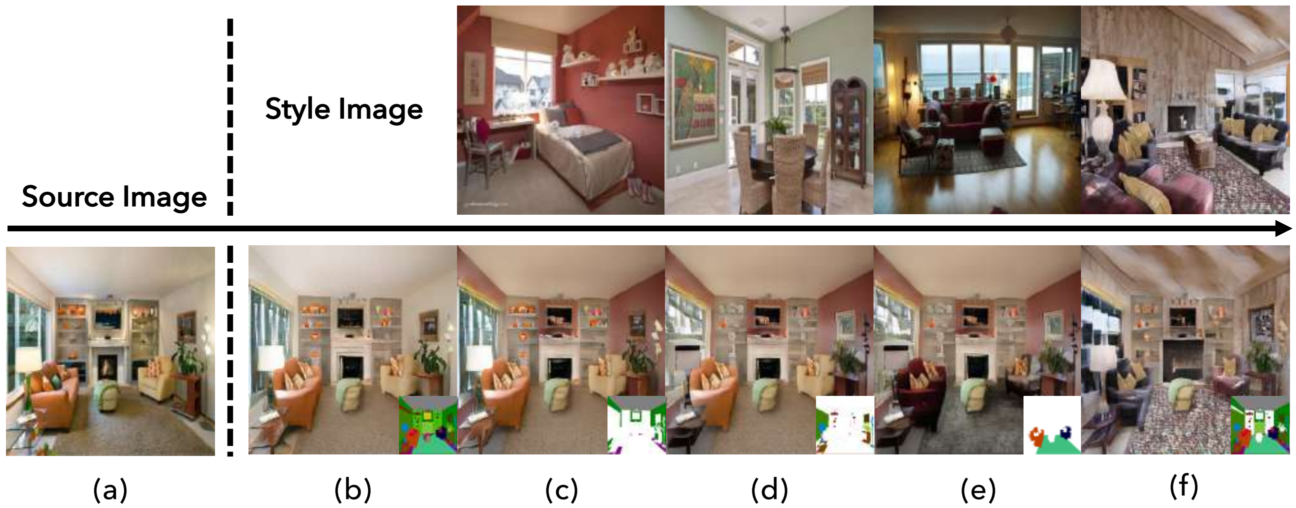


Figure 2: Editing sequence on the ADE20K dataset. (a) source image, (b) reconstruction of the source image, (c-f) various edits using style images shown in the top row. The regions affected by the edits are shown as small insets.

that has labels for each semantic region and “add” realistic styles to each region according to their labels . For example, a face generation application would use region labels like eyes, hair, nose, mouth, *etc*. and a landscape painting application would use labels like water, forest, sky, clouds, *etc*. While multiple very good frameworks exist to tackle this problem [23, 8, 40, 47], the currently best architecture is SPADE [39] (also called GauGAN). Therefore, we decided to use SPADE as starting point for our research. By analyzing the SPADE results, we found two shortcomings that we would like to improve upon in our work.

First, SPADE uses only one style code to control the entire style of an image, which is not sufficient for high quality synthesis or detailed control. For example, it is easily possible that the segmentation mask of the desired output image contains a labeled region that is not present in the segmentation mask of the input style image. In this case, the style of the missing region is undefined, which yields low quality results. Further, SPADE does not allow using a different style input image for each region in the segmentation mask. Our first main idea is therefore to control the style of each region individually, *i.e*., our proposed architecture accepts one style image per region (or per region instance) as input.

Second, we believe that inserting style information only in the beginning of a network is not a good architecture choice. Recent architectures [26, 32, 2] have demonstrated that higher quality results can be obtained if style information is injected as normalization parameters in multiple layers in the network, e.g. using AdaIN [20]. However, none of these previous networks use style information to generate spatially varying normalization parameters. To alleviate this shortcoming, our second main idea is to design a normalization building block, called SEAN, that can use style input images to create spatially varying normalization parameters per semantic region. An important aspect of this work is that the spatially varying normalization parameters are dependent on the segmentation mask as well as the style input images.

Empirically, we provide an extensive evaluation of our method on several challenging datasets: CelebAMask-HQ [29, 25, 33], CityScapes [10], ADE20K [53], and our own Façades dataset. Quantitatively, we evaluate our work on a wide range of metrics including FID, PSNR, RMSE and segmentation performance; qualitatively, we show examples of synthesized images that can be evaluated by visual inspection. Our experimental results demonstrate a large improvement over the current state-of-the-art methods. In summary, we introduce a new architecture building block *SEAN* that has the following advantages:

1. SEAN improves the quality of the synthesized images for conditional GANs. We compared to the state of the art methods SPADE and Pix2PixHD and achieve clear improvements in quantitative metrics (e.g. FID score) and visual inspection.

2. SEAN improves the per-region style encoding, so that reconstructed images can be made more similar to the input style images as measured by PSNR and visual inspection.

3. SEAN allows the user to select a different style input image for each semantic region. This enables image editing capabilities producing much higher quality and providing better control than the current state-of-the-art methods. Example image editing capabilities are interactive region by region style transfer and per-region style interpolation (See Figs. 1, 2, and 5).

2. Related Work

Generative Adversarial Networks. Since their introduction in 2014, Generative Adversarial Networks (GANs) [15] have been successfully applied to various image synthesis tasks, *e.g*. image inpainting [51, 11], image manipulation [55, 5, 1] and texture synthesis [30, 45, 12]. With continuous improvements on GAN architecture [41, 26, 39], loss function [34, 4] and regularization [17, 37, 35], the images synthesized by GANs are becoming more and more realistic. For example, the human face images generated by StyleGAN [26] are of very high quality and are almost indistinguishable from photographs by untrained viewers. A traditional GAN uses noise vectors as the input and thus provides little user control. This motivates the development of conditional GANs (cGANs) [36] where users can control the synthesis by feeding the generator with conditioning information. Examples include class labels [38, 35, 6], text [42, 19, 49] and images [23, 56, 31, 47, 47, 39]. Our work is built on the conditional GANs with image inputs, which aims to tackle image-to-image translation problems.

Image-to-Image Translation. Image-to-image translation is an umbrella concept that can be used to describe many problems in computer vision and computer graphics. As a milestone, Isola *et al*. [23] first showed that image-conditional GANs can be used as a general solution to various image-to-image translation problems. Since then, their method is extended by several following works to scenarios including: unsupervised learning [56, 31], few-shot learning [32], high resolution image synthesis [47], multi-modal image synthesis [57, 22] and multi-domain image synthesis [9]. Among various image-to-image translation problems, semantic image synthesis is a particularly useful genre as it enables easy user control by modifying the input semantic layout image [29, 5, 16, 39]. To date, the SPADE [39] model (also called GauGAN) generates the highest quality results. In this paper, we will improve SPADE by introducing per-region style encoding.

Style Encoding. Style control is a vital component for various image synthesis and manipulation applications [13, 30, 21, 26, 1]. *Style* is generally not manually designed by a user, but extracted from reference images. In most existing methods, styles are encoded in three places: i) statistics of image features [13, 43]; ii) neural network weights (*e.g.* fast style transfer [24, 54, 50]); iii) parameters of a network normalization layer [21, 28]). When applied to style control, the first encoding method is usually time-consuming as it requires a slow optimization process to match the statistics of image features extracted by image classification networks [13]. The second approach runs in real-time but each neural network only encodes the style of selected reference images [24]. Thus, a separate neural network is required to be trained for each style image, which limits its application in practice. To date, the third approach is the best as it enables arbitrary style transfer in real-time [21] and it is used by high-quality networks such as StyleGAN [26], FUNIT [32], and SPADE [39]. Our per-region style encoding also builds on this approach. We will show that our method generates higher quality results and enables more-detailed user control.

3. Per Region Style Encoding and Control

Given an input style image and its corresponding segmentation mask, this section shows i) how to distill the per-region styles from the image according to the mask and ii) how to use the distilled per-region style codes to synthesize photo-realistic images.

3.1. How to Encode Style?

Per-Region Style Encoder. To extract per region styles, we propose a novel style encoder network to distill the corresponding style code from each semantic region of an input image simultaneously (See the subnetwork *Style Encoder* in Figure 4 (A)). The output of the style encoder is a $512 \times s$ dimensional style matrix $\mathbf{ST}$, where s is the number of semantic regions in the input image. Each column of the matrix corresponds to the style code of a semantic region. Unlike the standard encoder built on a simple downscaling convolutional neural network, our per-region style encoder employs a "bottleneck" structure to remove the information irrelevant to styles from the input image. Incorporating the prior knowledge that styles should be independent of the shapes of semantic regions, we pass the intermediate feature maps (512 channels) generated by the network block *TConv-Layers* through a region-wise average pooling layer and reduce them to a collection of 512 dimensional vectors. As implementation detail we would like to remark that we use s as the number of semantic labels in the data set and set columns corresponding to regions that do not exist in an input image to 0. As a variation of this architecture, we can also extract style per region instance for datasets that have instance labels, e.g., CityScapes.

3.2. How to Control Style?

With per-region style codes and a segmentation mask as inputs, we propose a new conditional normalization technique called Semantic Region-Adaptive Normalization (SEAN) to enable detailed control of styles for photo-realistic image synthesis. Similar to existing normalization techniques [21, 39], SEAN works by modulating the scales and biases of generator activations. In contrast to all existing methods, the modulation parameters learnt by SEAN are dependent on both the style codes and segmentation masks. In a SEAN block (Figure 3), a style map is first gen-

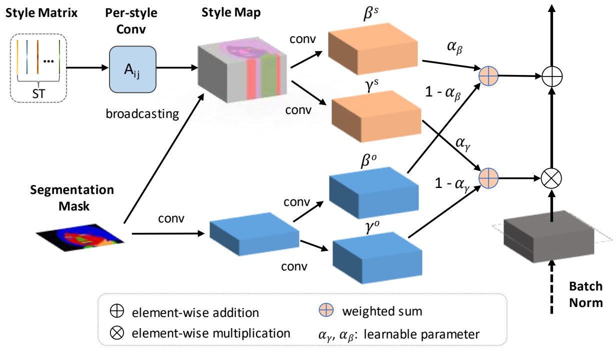


Figure 3: SEAN normalization. The input are style matrix $\mathbf{ST}$ and segmentation mask $\mathbf{M}$. In the upper part, the style codes in $\mathbf{ST}$ undergo a per style convolution and are then broadcast to their corresponding regions according to $\mathbf{M}$ to yield a style map. The style map is processed by conv layers to produce per pixel normalization values γ^s and β^s. The lower part (light blue layers) creates per pixel normalization values using only the region information similar to SPADE.

erated by broadcasting the style codes to their corresponding semantic regions according to the input segmentation mask. This style map along with the input segmentation mask are then passed through two separate convolutional neural networks to learn two sets of modulation parameters. The weighted sums of them are used as the final SEAN parameters to modulate the scales and biases of generator activations. The weight parameter is also learnable during training. The formal definition of SEAN is introduced as follows.

Semantic Region-Adaptive Normalization (SEAN). A SEAN block has two inputs: a style matrix $\mathbf{ST}$ encoding per-region style codes and a segmentation mask $\mathbf{M}$. Let $\mathbf{h}$ denote the input activation of the current SEAN block in a deep convolutional network for a batch of N samples. Let H, W and C be the height, width and the number of channels in the activation map. The modulated activation value at site $(n \in N, c \in C, y \in H, x \in W)$ is given by

$$\gamma_{c,y,x}(\mathbf{ST},\mathbf{M})\frac{h_{n,c,y,x}-\mu_c}{\sigma_c}+\beta_{c,y,x}(\mathbf{ST},\mathbf{M}) \tag{1}$$

where $h_{n,c,y,x}$ is the activation at the site before normalization, the modulation parameters $\gamma_{c,y,x}$ and $\beta_{c,y,x}$ are weighted sums of $\gamma^s_{c,y,x}$, $\gamma^o_{c,y,x}$ and $\beta^s_{c,y,x}$, $\beta^o_{c,y,x}$ respectively (See Fig. 3 for definition of γ and β variables):

$$\begin{aligned}\gamma_{c,y,x}(\mathbf{ST},\mathbf{M}) &= \alpha_\gamma\gamma^s_{c,y,x}(\mathbf{ST})+(1-\alpha_\gamma)\gamma^o_{c,y,x}(\mathbf{M})\\ \beta_{c,y,x}(\mathbf{ST},\mathbf{M}) &= \alpha_\beta\beta^s_{c,y,x}(\mathbf{ST})+(1-\alpha_\beta)\beta^o_{c,y,x}(\mathbf{M})\end{aligned} \tag{2}$$

μ_c and σ_c are the mean and standard deviation of the activation in channel c:

$$\mu_c = \frac{1}{NHW}\sum_{n,y,x} h_{n,c,y,x} \tag{3}$$

$$\sigma_c = \sqrt{\frac{1}{NHW}\left(\sum_{n,y,x} h^2_{n,c,y,x}\right)-\mu_c^2} \tag{4}$$

4. Experimental Setup

4.1. Network Architecture

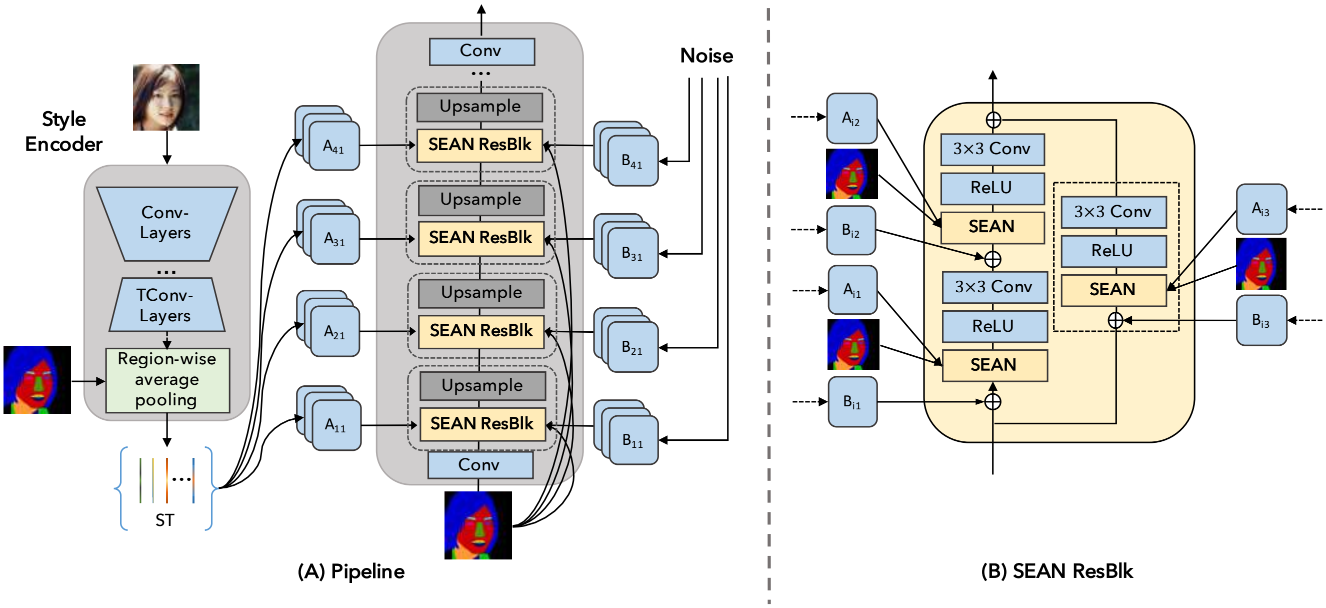
Figure 4 (A) shows an overview of our generator network, which is building on that of SPADE [39]. Similar to [39], we employ a generator consisting of several SEAN ResNet blocks (SEAN ResBlk) with upsampling layers.

SEAN ResBlk. Figure 4 (B) shows the structure of our SEAN ResBlk, which consists of three convolutional layers whose scales and biases are modulated by three SEAN blocks respectively. Each SEAN block takes two inputs: a set of per-region style codes $\mathbf{ST}$ and a semantic mask $\mathbf{M}$. Note that both inputs are adjusted in the beginning: the input segmentation mask is downsampled to the same height and width of the feature maps in a layer; the input style codes from $\mathbf{ST}$ are transformed per region using a 1×1 conv layer A_{ij}. We observed that the initial transformations are indispensable components of the architecture because they transform the style codes according to the different roles of each neural network layer. For example, early layers might control the hair styles (*e.g.* wavy, straight) of human face images while later layers might refer to the lighting and color. In addition, we observed that adding noise to the input of SEAN can improve the quality of synthesized images. The scale of such noise is adjusted by a per-channel scaling factor B that is learnt during training, similar to StyleGAN [26].

4.2. Training and Inference

Training We formulate the training as an image reconstruction problem. That is, the style encoder is trained to distill per-region style codes from the input images according to their corresponding segmentation masks. The generator network is trained to reconstruct the input images with the extracted per-region style codes and the corresponding segmentation masks as inputs. Following SPADE [39] and Pix2PixHD [47], the difference between input images and reconstructed images are measured by an overall loss function consisting of three loss terms: conditional adversarial loss, feature matching loss [47] and perceptual loss [24]. Details of the loss function are included in the supplementary material.

Inference During inference, we take an arbitrary segmentation mask as the mask input and implement per-region style



(A) Pipeline

(B) SEAN ResBlk

Figure 4: SEAN generator. (A) On the left, the style encoder takes an input image and outputs a style matrix **ST**. The generator on the right consists of interleaved SEAN ResBlocks and Upsampling layers. (B) A detailed view of a SEAN ResBlock used in (A).

Method	CelebAMask-HQ			CityScapes			ADE20K			Façades		
	SSIM	RMSE	PSNR	SSIM	RMSE	PSNR	SSIM	RMSE	PSNR	SSIM	RMSE	PSNR
Pix2PixHD [47]	0.68	0.15	17.14	0.69	**0.13**	18.32	0.51	0.22	13.81	0.53	0.16	16.30
SPADE [39]	0.63	0.21	14.30	0.64	0.18	15.77	0.45	0.28	11.52	0.44	0.22	13.87
Ours	**0.73**	**0.12**	**18.74**	**0.70**	**0.13**	**18.61**	**0.58**	**0.17**	**16.16**	**0.58**	**0.14**	**17.14**

Table 1: Quantitative comparison of reconstruction quality. Our method outperforms current leading methods using similarity metrics SSIM, RMSE, and PSNR on all the datasets. For SSIM and PSNR, higher is better. For RMSE, lower is better.

control by selecting a separate 512 dimensional style code for each semantic region as the style input. This enables a variety of high quality image synthesis applications, which will be introduced in the following section.

5. Results

In the following we discuss quantitative and qualitative results of our framework.

Implementation details. Following SPADE [39], we apply Spectral Norm [37] in both the generator and discriminator. Additional normalization is performed by SEAN in the generator. We set the learning rates to 0.0001 and 0.0004 for the generator and discriminator, respectively [18]. For the optimizer, we choose ADAM [27] with $\beta_1 = 0, \beta_2 = 0.999$. All the experiments are trained on 4 Tesla v100 GPUs. To get better performance, we use a synchronized version of batch normalization [52] in the SEAN normalization blocks.

Datasets. We use the following datasets in our experiments: 1) CelebAMask-HQ [29, 25, 33] containing $30,000$ segmentation masks for the CelebAHQ face image dataset. There are 19 different region categories. 2) ADE20K [53] contains $22,210$ images annotated with 150 different region labels. 3) Cityscapes [10] contains 3500 images annotated with 35 different region labels. 4) Façades dataset. we use a dataset of $30,000$ façade images collected from Google Street View [3]. Instead of manual annotation, the segmentation masks are automatically calculated by a pre-trained DeepLabV3+ Network [7].

We follow the recommended splits for each dataset and note that all images in the paper are generated from the test set only. Neither the segmentation masks nor the style images have been seen during training.

Metrics. We employ the following established metrics to compare our results to state of the art: 1) segmentation ac-

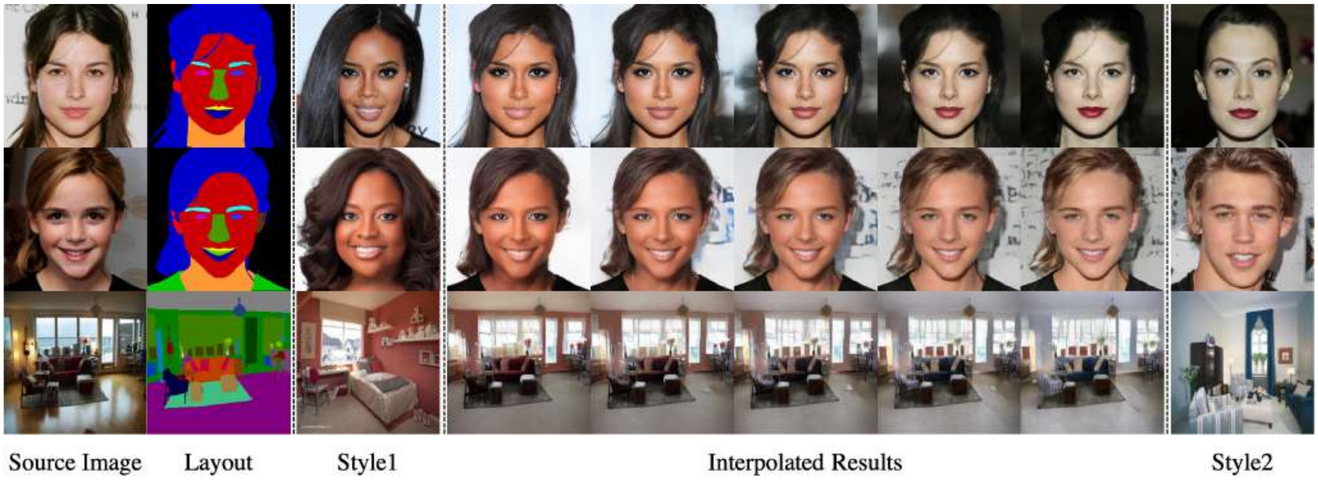


Figure 5: Style interpolation. We take a mask from a source image and reconstruct with two different style images (Style1 and Style2) that are very different from the source image. We then show interpolated results of the per-region style codes.

Method	CelebAMask-HQ			CityScapes			ADE20K			Façades
	mIoU	accu	FID	mIoU	accu	FID	mIoU	accu	FID	FID
Ground Truth	73.14	94.38	9.41	66.21	93.69	32.34	39.38	78.76	14.51	14.40
Pix2PixHD [47]	76.12	95.76	23.69	50.35	92.09	83.24	22.78	73.32	43.0	22.34
SPADE [39]	**77.01**	**95.93**	22.43	56.01	93.13	60.51	**35.37**	**79.37**	34.65	24.04
Ours	75.69	95.69	**17.66**	**57.88**	**93.59**	**50.38**	34.59	77.16	**24.84**	**19.82**

Table 2: Quantitative comparison using semantic segmentation performance measured by mIoU and accu and generation performance measured by FID. Our method outperforms current leading methods in FID on all the datasets.

curacy measured by mean Intersection-over-Union (mIoU) and pixel accuracy (accu), 2) FID [18], 3) peak signal-to-noise ratio (PSNR), 4) structural similarity (SSIM) [48], 5) root mean square error (RMSE).

Quantitative comparisons. In order to facilitate a fair comparison to SPADE, we report reconstruction performance where only a single style image is employed. We train one network for each of the four datasets and report the previously described metrics in Table 1 and 2. We choose SPADE [39] as the currently best state-of-the-art method and Pix2PixHD [47] as the second best method in the comparisons. Based on visual inspection of the results, we found that the FID score is most indicative of visual quality of the results and that a lower FID score often (but not always) corresponds to better visual results. Generally, we can observe that our SEAN framework clearly beats the state of the art on all datasets.

Method	SSIM	RMSE	PSNR	FID
Pix2PixHD [47]	0.68	0.15	17.14	23.69
SPADE [39]	0.63	0.21	14.30	22.43
SPADE++	0.67	0.16	16.71	20.80
SEAN-level encoders	0.74	0.11	19.70	18.17
ResBlk-level encoders	0.72	0.12	18.86	17.98
unified encoder	0.73	0.12	18.74	17.66
w/ downsampling	0.73	0.12	18.74	17.66
w/o downsampling	0.72	0.13	18.32	18.67
w/ noises	0.73	0.12	18.74	17.66
w/o noises	0.72	0.13	18.49	18.05

Table 3: Ablation study on CelebAMask-HQ dataset. See supplementary materials for further details.

Qualitative results. We show visual comparisons for the four datasets in Fig. 6. We can observe that the quality difference between our work and the state of the art can be significant. For example, we notice that SPADE and Pix2Pix cannot handle more extreme poses and older people. We conjecture that our network is more suitable to learn more of the variability in the data because of our improved style encoding. Since a visual inspection of the results is very important for generative modeling, we would like to refer the reader to the supplementary materials and the accompanying video for more results. Our per-region style encoding

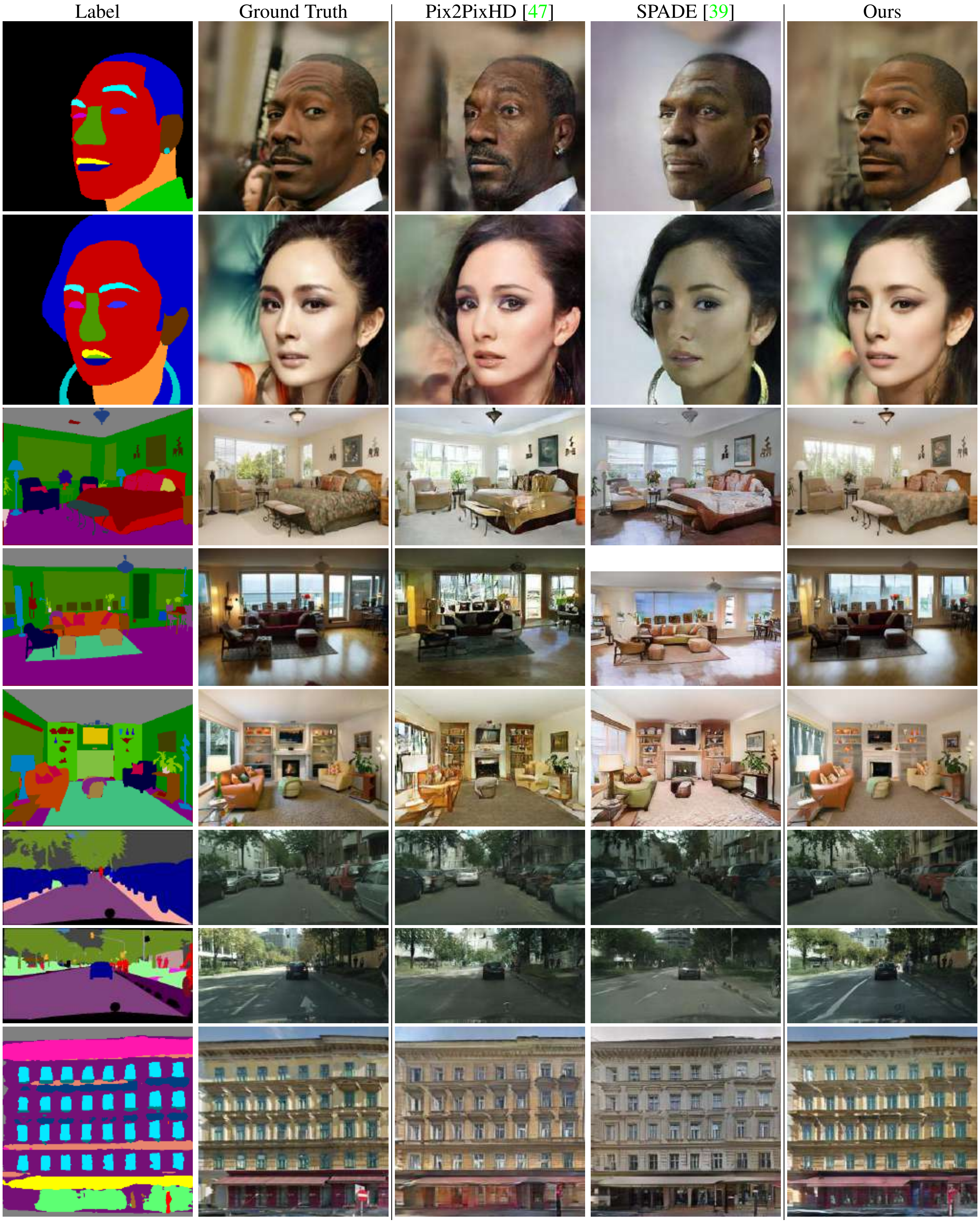


Figure 6: Visual comparison of semantic image synthesis results on the CelebAMask-HQ, ADE20K, CityScapes and Façades dataset. We compare Pix2PixHD, SPADE, and our method. See supplementary materials for further details.

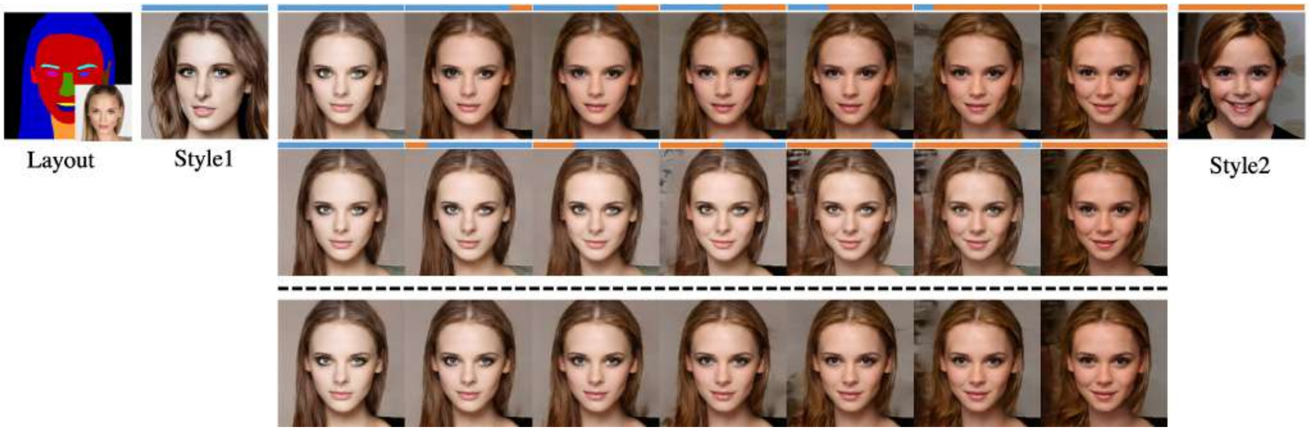


Figure 7: Style crossover. In addition to style interpolation (bottom row), we can perform crossover by selecting different styles per ResBlk. We show two transitions in the top two rows. The blue / orange bars on top of the images indicate which styles are used by the six ResBlks. We can observe that earlier layers are responsible for larger features and later layers mainly determine the color scheme.

also enables new image editing operations: iterative image editing with per-region style control (See Figs. 1 and 2), style interpolation and style crossover (See Fig. 5 and 7).

Variations of SEAN generator (Ablation studies). In Table 3, we compare many variations of our architecture with previous work on the CelebAMask-HQ dataset.

According to our analysis, Pix2PixHD has a better style extraction subnetwork than SPADE (because the PSNR reconstruction values are better), but SPADE has a better generator subnetwork (because the FID scores are better). We therefore build another network that combines the style encoder from Pix2PixHD with the generator from SPADE and call this architecture SPADE++. We can observe that SPADE++ improves SPADE by a bit, but all our architecture variations still have a better FID and PSNR score.

To evaluate our design choices, we report the performance of variations of our encoder. First we compare three variants to encode style: SEAN-level encoder, ResBlk-level encoder, and unified encoder. A fundamental design choice is how to split the style computation into two parts: a shared computation in the style encoder and a per layer computation in each SEAN block. The first two variants perform all style extraction per-layer, while the unified encoder is the architecture described in the main part of the paper. While the other two encoders have better reconstruction performance, they lead to lower FID scores. Based on visual inspection, we also confirmed that the unified encoder leads to better visual quality. This is especially noticeable for difficult inputs (see supplementary materials). Second, we evaluate if downsampling in the style encoder is a good idea and evaluate a style encoder with bottleneck (with downsampling) with another encoder that uses less downsampling. This test confirms that introducing a bottleneck in the style encoder is a good idea. Lastly, we test the effect of the learnable noise. We found the learnable noise can help both similarity (PSNR) and FID performance. More details of the alternative architectures are provided in supplementary materials.

6. Conclusion

We propose semantic region-adaptive normalization (SEAN), a simple but effective building block for Generative Adversarial Networks (GANs) conditioned on segmentation masks that describe the semantic regions in the desired output image. Our main idea is to extend SPADE, the currently best network, to control the style of each semantic region individually, e.g. we can specify one style reference image per semantic region. We introduce a new building block, SEAN normalization, that can extract style from a given style image for a semantic region and processes the style information to bring it in the form of spatially-varying normalization parameters. We evaluate SEAN on multiple datasets and report better quantitative metrics (e.g. FID, PSNR) than the current state of the art. While SEAN is designed to specify style for each region independently, we also achieve big improvements in the basic setting where only one style image is provided for all regions. In summary, SEAN is better suited to encode, transfer, and synthesize style than the best previous method in terms of reconstruction quality, variability, and visual quality. SEAN also pushes the frontier of interactive image editing. In future work, we plan to extend SEAN to processing meshes, point clouds, and textures on surfaces.

Acknowledgement This work was supported by the KAUST Office of Sponsored Research (OSR) under Award No. OSR-CRG2018-3730.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4432–4441, 2019. 3

[2] Yazeed Alharbi, Neil Smith, and Peter Wonka. Latent filter scaling for multimodal unsupervised image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1458–1466, 2019. 2

[3] Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stéphane Lafon, Richard Lyon, Abhijit Ogale, Luc Vincent, and Josh Weaver. Google street view: Capturing the world at street level. *Computer*, 43(6):32–38, 2010. 5

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 3

[5] David Bau, Hendrik Strobelt, William Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. Semantic photo manipulation with a generative image prior. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 38(4), 2019. 3

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018. 3

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018. 5

[8] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks, 2017. 2

[9] 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, 2017. 3

[10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 5, 12, 13

[11] Ugur Demir and Gozde Unal. Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*, 2018. 3

[12] Anna Frhstck, Ibraheem Alhashim, and Peter Wonka. Tilegan. *ACM Transactions on Graphics*, 38(4):111, Jul 2019. 3

[13] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv*, Aug 2015. 3

[14] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. 12

[15] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 3

[16] Shuyang Gu, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen, and Lu Yuan. Mask-guided portrait editing with conditional gans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017. 3

[18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2017. 5, 6

[19] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7986–7994, 2018. 3

[20] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization, 2017. 2

[21] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 3

[22] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation, 2018. 3

[23] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2016. 2, 3, 11

[24] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016. 3, 4, 11

[25] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017. 2, 5, 13

[26] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2018. 2, 3, 4

[27] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. 5

[28] Dmytro Kotovenko, Artsiom Sanakoyeu, Sabine Lang, and Bjorn Ommer. Content and style disentanglement for artistic style transfer. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 3

[29] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. *arXiv preprint arXiv:1907.11922*, 2019. 2, 3, 5, 13

[30] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*, 2016. 3

[31] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks, 2017. 3

[32] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation, 2019. 2, 3

[33] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 2, 5, 13

[34] Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[35] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge?, 2018. 3

[36] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 3

[37] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018. 3, 5, 11

[38] Takeru Miyato and Masanori Koyama. cgans with projection discriminator, 2018. 3

[39] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 3, 4, 5, 6, 7, 11, 13, 16, 17, 18, 19

[40] Xiaojuan Qi, Qifeng Chen, Jiaya Jia, and Vladlen Koltun. Semi-parametric image synthesis, 2018. 2

[41] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 3

[42] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016. 3

[43] Omry Sendik and Daniel Cohen-Or. Deep correlations for texture synthesis. *ACM Transactions on Graphics (TOG)*, 36(5):161, 2017. 3

[44] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. 11

[45] Ron Slossberg, Gil Shamai, and Ron Kimmel. High quality facial surface and texture synthesis via generative adversarial networks. In *European Conference on Computer Vision*, pages 498–513. Springer, 2018. 3

[46] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization, 2016. 11

[47] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 3, 4, 5, 6, 7, 11, 13, 16, 17, 18, 19

[48] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 13(4):600–612, 2004. 6

[49] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1316–1324, 2018. 3

[50] Shuai Yang, Zhangyang Wang, Zhaowen Wang, Ning Xu, Jiaying Liu, and Zongming Guo. Controllable artistic text style transfer via shape-matching gan. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 3

[51] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5505–5514, 2018. 3

[52] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5

[53] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 5, 13

[54] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Non-stationary texture synthesis by adversarial expansion. *arXiv preprint arXiv:1805.04487*, 2018. 3

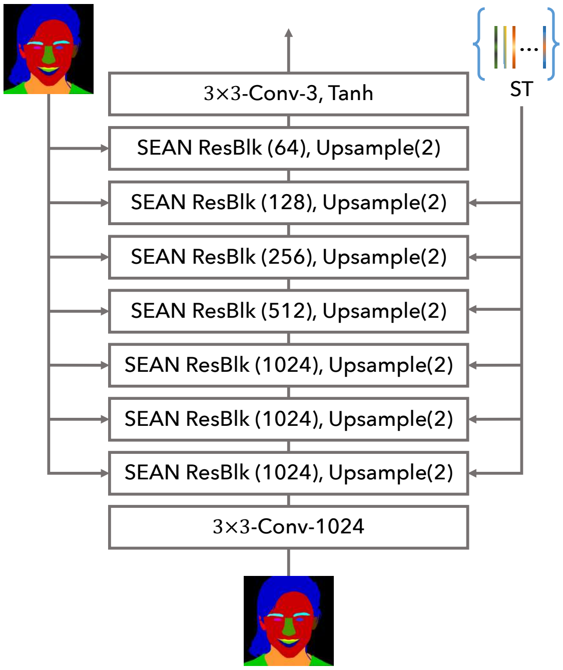
[55] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016. 3

[56] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2017. 3

[57] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems*, 2017. 3

A. Additional Implementation Details

Generator. Our generator consists of several SEAN ResBlks. Each of them is followed by a nearest neighbor upsampling layer. Note that we only inject the style codes **ST** into the first 6 SEAN ResBlks. The other inputs are injected to all SEAN ResBlks. The architecture of our generator is shown in Figure 8.



ST
3×3-Conv-3, Tanh
SEAN ResBlk (64), Upsample(2)
SEAN ResBlk (128), Upsample(2)
SEAN ResBlk (256), Upsample(2)
SEAN ResBlk (512), Upsample(2)
SEAN ResBlk (1024), Upsample(2)
SEAN ResBlk (1024), Upsample(2)
SEAN ResBlk (1024), Upsample(2)
3×3-Conv-1024

Figure 8: SEAN Generator. The style codes **ST** and segmentation mask are passed to the generator through the proposed SEAN ResBlks. The number of feature map channels is shown in the parenthesis after each SEAN ResBlk. To better illustrate the architecture, we omit the learnable noise inputs and per-style Conv layers in this figure. These details are shown in Fig.3 of the main paper (see A_{ij} and B_{ij}).

Discriminator. Following SPADE [39] and Pix2PixHD [47], we employed two multi-scale discriminators with instance normalization (IN) [46] and Leaky ReLU (LReLU). Similar to SPADE, we apply spectral normalization [37] to all the convolutional layers of the discriminator. The architecture of our discriminator is shown in Figure 9.

Style Encoder. Our style encoder consists of a "bottleneck" convolutional neural network and a region-wise average pooling layer (Figure 11). The inputs are the style image and the corresponding segmentation mask, while the outputs are the style codes **ST**.

Loss function. The design of our loss function is inspired by those of SPADE and Pix2PixHD which contains three components:

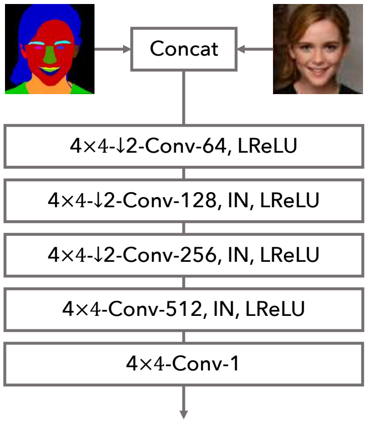


Figure 9: Following SPADE and Pix2PixHD, our discriminator takes the concatenation of a segmentation mask and a style image as inputs. The loss is calculated in the same way as PatchGAN [23].

(1) *Adversarial loss.* Let E be the style encoder, G be the SEAN generator, D_1 and D_2 be two discriminators at different scales [47], $\mathbf{R}$ be a given style image, $\mathbf{M}$ be the corresponding segmentation mask of $\mathbf{R}$, we formulate the conditional adversarial learning part of our loss function as:

$$\min_{E,G} \max_{D_1,D_2} \sum_{k=1,2} \mathcal{L}_{\text{GAN}}\left(E, G, D_k\right) \tag{5}$$

Specifically, $\mathcal{L}_{\text{GAN}}$ is built with the Hinge loss that:

$$\begin{aligned} \mathcal{L}_{\text{GAN}} &= \mathbb{E}\left[\max(0, 1 - D_k(\mathbf{R}, \mathbf{M}))\right] \\ &\quad + \mathbb{E}\left[\max(0, 1 + D_k(G(\mathbf{ST}, \mathbf{M}), \mathbf{M}))\right] \end{aligned} \tag{6}$$

where **ST** is the style codes of $\mathbf{R}$ extracted by E under the guidance of $\mathbf{M}$:

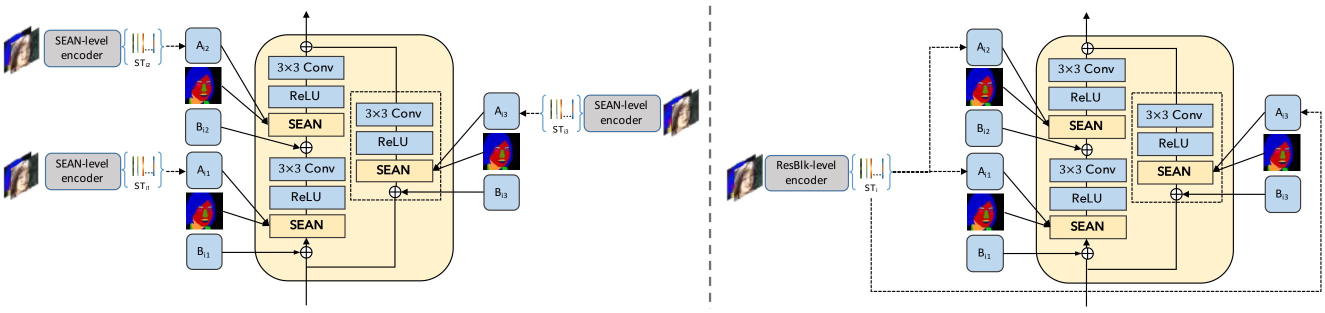
$$\mathbf{ST} = E\left(\mathbf{R}, \mathbf{M}\right) \tag{7}$$

(2) *Feature matching loss* [47]. Let T be the total number of layers in discriminator D_k, $D_k^{(i)}$ and N_i be the output feature maps and the number of elements of the i-th layer of D_k respectively, we denote the feature matching loss term $\mathcal{L}_{FM}$ as:

$$\mathcal{L}_{FM} = \mathbb{E} \sum_{i=1}^{T} \frac{1}{N_i} \left[\left\| D_k^{(i)}(\mathbf{R}, \mathbf{M}) - D_k^{(i)}\left(G(\mathbf{ST}, \mathbf{M}), \mathbf{M}\right) \right\|_1 \right] \tag{8}$$

(3) *Perceptual loss* [24]. Let N be the total number of layers used to calculate the perceptual loss, $F^{(i)}$ be the output feature maps of the i-th layer of the VGG network [44], M_i be the number of elements of $F^{(i)}$, we denote the perceptual loss $\mathcal{L}_{percept}$ as:

$$\mathcal{L}_{percept} = \mathbb{E} \sum_{i=1}^{N} \frac{1}{M_i} \left[\left\| F^{(i)}\left(\mathbf{R}\right) - F^{(i)}\left(G(\mathbf{ST}, \mathbf{M})\right) \right\|_1 \right] \tag{9}$$



(A) SEAN ResBlk -- with SEAN-level encoder

(A) SEAN ResBlk -- with ResBlk-level encoder

Figure 10: Detailed usage of the SEAN-level encoder and the ResBlk-level encoder within a SEAN ResBlk.

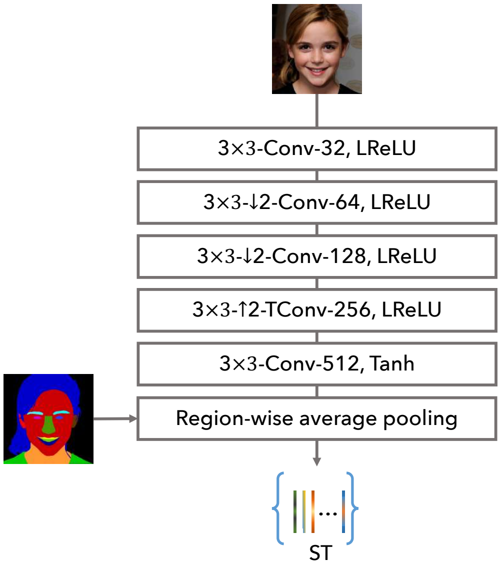


Figure 11: Our style encoder takes the style image and the segmentation mask as inputs to generate the style codes **ST**.

The final loss function used in our experiment is made up of the above-mentioned three loss terms as:

$$\min_{E,G}\left(\left(\max_{D_1,D_2}\sum_{k=1,2}\mathcal{L}_{\text{GAN}}\right)+\lambda_1\sum_{k=1,2}\mathcal{L}_{\text{FM}}+\lambda_2\mathcal{L}_{\text{percept}}\right) \tag{10}$$

Following SPADE and Pix2PixHD, we set $\lambda_1 = \lambda_2 = 10$ in our experiments.

Training details. We perform 50 epochs of training on all the datasets mentioned in the main paper. During training, all input images are resized to a resolution of 256×256, except for the CityScapes dataset [10] whose images are resized to 512×256. We use Glorot initialization [14] to initialize our network weights.

B. Additional Experimental Details

Table 3 (main paper). Supplementing row 5 and 6 in Table 3 of the main paper, Figure 10 shows how the two variants of style encoders (*i.e*. the SEAN-level encoder and the ResBlk-level encoder) are used in a SEAN ResBlk. Specifically, the SEAN-level encoders extract different style codes for each SEAN block while the same style codes extracted by the ResBlk-level encoder are used by all SEAN blocks within a SEAN ResBlk.

Figure 6 (main paper). We used the Ground Truth (second column in Figure 6 of the main paper) as the style input for all methods. For Pix2pixHD, we generate the results by: (i) encoding the style image into a style vector; (ii) broadcasting the style vector and concatenating it to the mask input of the generator.

C. Justification of Encoder Choice

Figure 12 shows that the images generated by the unified encoder are of better visual quality than those generated by the SEAN-level encoder, especially for challenging inputs (e.g. extreme poses, unlabeled regions), which justifies our choice of unified encoder.

D. Additional Analysis

ST-branch *vs*. Mask-branch. The contributions of ST-branch and mask-branch are determined by a linear combination (parameters α_β and α_γ). The resulting parameters are typically in the range of $0.35 - 0.7$ meaning that both branches are actively contributing to the result. See Fig 13 for one example. It is possible to completely drop the mask-branch, but the results will get worse. It was our initial intuition that the mask branch provides the rough structure and the ST-branch additional details. However, in the end, the interaction is quite complicated and cannot be understood by just varying the mixing parameter.

Extreme Cases. To further demonstrate SEAN's power in

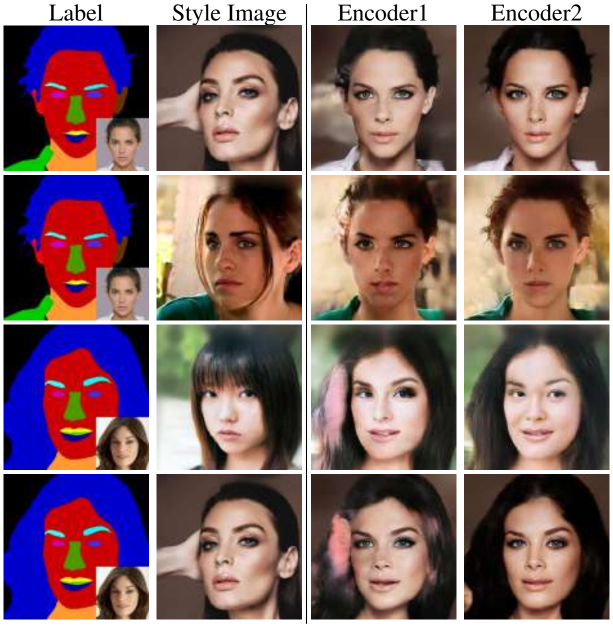


Figure 12: Encoder choice justification. Encoder1 is the SEAN-level encoder and Encoder2 is the unified encoder. SEAN-level encoder is more sensitive to the poses and unlabeled parts of the style image due to the overfitting. Using unified encoder can get more robust style transfer results.

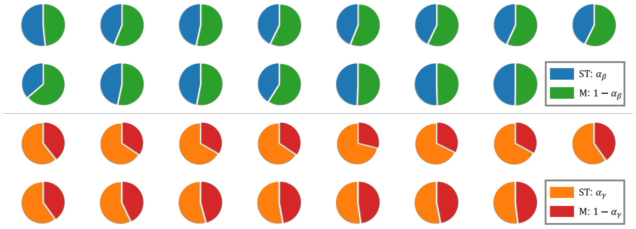


Figure 13: Contributions of ST-branch and Mask-branch for each SEAN normalization block. The pie charts and SEAN normalization blocks are in one-to-one correspondence.

texture transfer, we show that highly complex textures from an artistic image can be transferred to a human face (Fig 14).

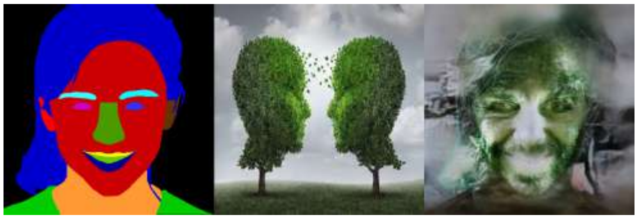


Figure 14: Complex texture transfer.

In addition, our method is highly flexible that enables users to paint a semantic region at a spatially unreasonable location arbitrarily (Fig 15).

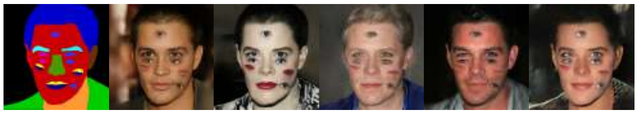


Figure 15: Spatially-flexible painting. Our method allows users to put eyes anywhere on a face.

E. User Study

We conducted a user preference study with Amazon Mechanical Turk (AMT) to illustrate our superior reconstruction results against existing methods (Table 4). Specifically, we created 600 questions for AMT workers to answer. In the end, our questions are answered by 575 AMT workers. For each question, we show the user a set of 5 images: a ground truth image, its corresponding segmentation mask, and 3 reconstruction images obtained by our method, Pix2PixHD [47] and SPADE [39]. The user is then asked to select the reconstructed image closest to the ground truth and with fewest artifacts. To relieve the impact of image orders and make a fair comparison, we picked 100 image sets randomly and created the 600 questions by enumerating all the 6 possible orders of the 3 reconstructed images in each of them.

	Pix2PixHD [47]	SPADE [39]	Ours
Preference (%)	23.17	8.83	68.00

Table 4: User preference study (CelebAMask-HQ dataset). Our method outperforms Pix2PixHD [47] and SPADE [39] significantly.

F. Additional Results

To demonstrate that the proposed per-region style control method builds the foundation of a highly flexible image-editing software, we designed an interactive UI for a demo. Our UI enables high quality image synthesis by transferring the per-region styles from various images to an arbitrary segmentation mask. New styles can be created by interpolating existing styles. Please find the recorded videos of our demo in the supplementary material.

Figure 16 shows additional style transfer results on CelebAMask-HQ [29, 25, 33] dataset. Figure 17 and Figure 18 show additional style interpolation results on CelebAMask-HQ and ADE20K datasets.

Figure 19, 20, 21 and 22 show additional image reconstruction results of our method, Pix2PixHD and SPADE on the CelebAMask-HQ [29, 25, 33], ADE20K [53], CityScapes [10] and our Façades datasets respectively. It can be observed that our reconstructions are of much higher quality than those of Pix2PixHD and SPADE.

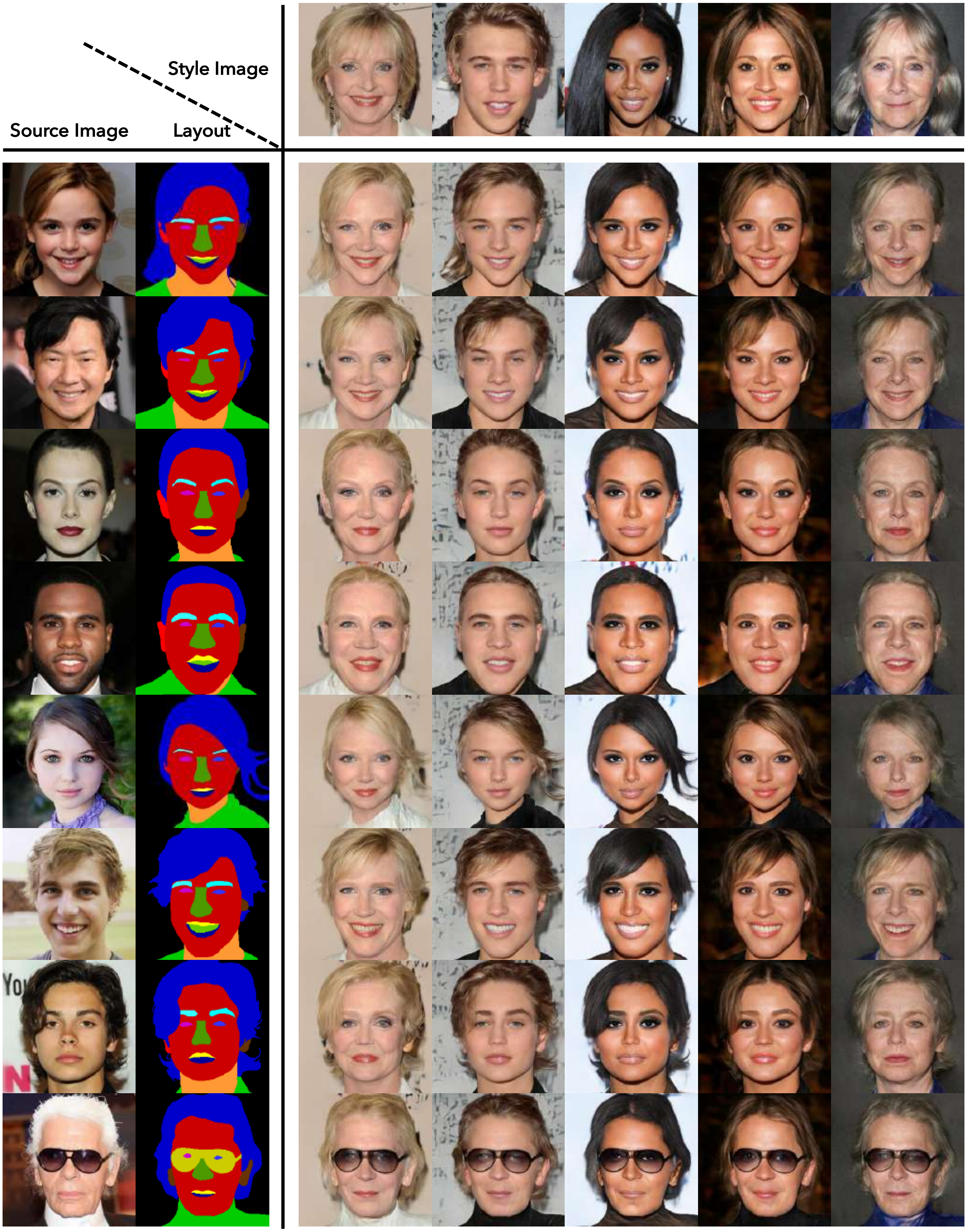


Figure 16: Style transfer on CelebAMask-HQ dataset

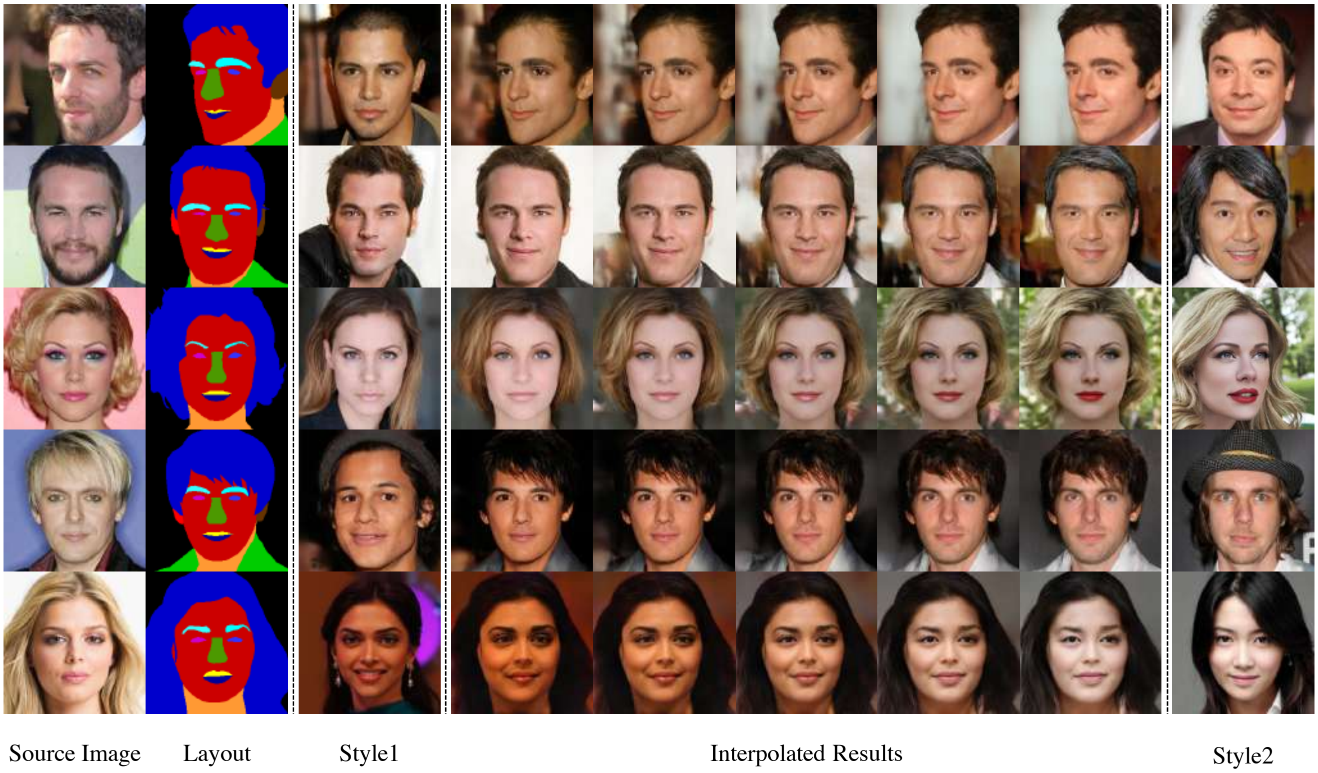


Figure 17: Style interpolation on CelebAMask-HQ dataset

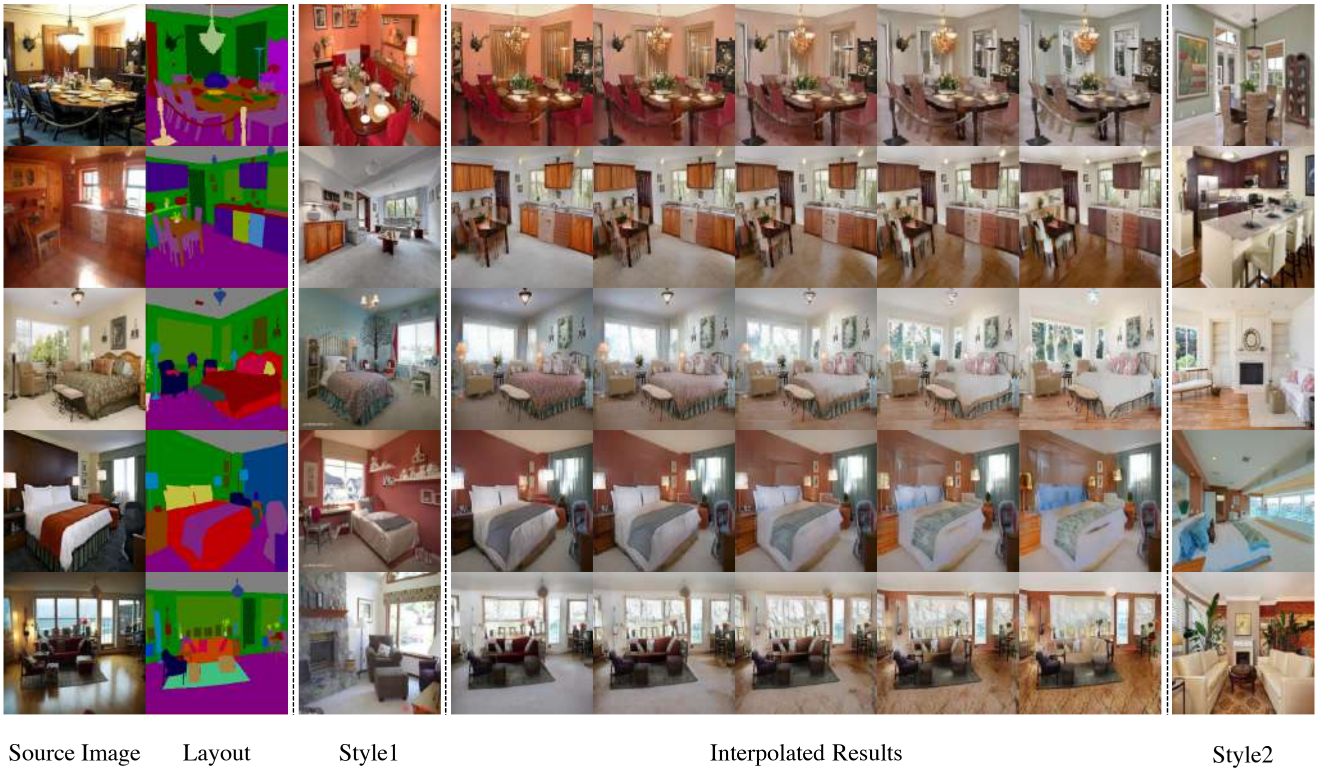


Figure 18: Style interpolation on ADE20K dataset

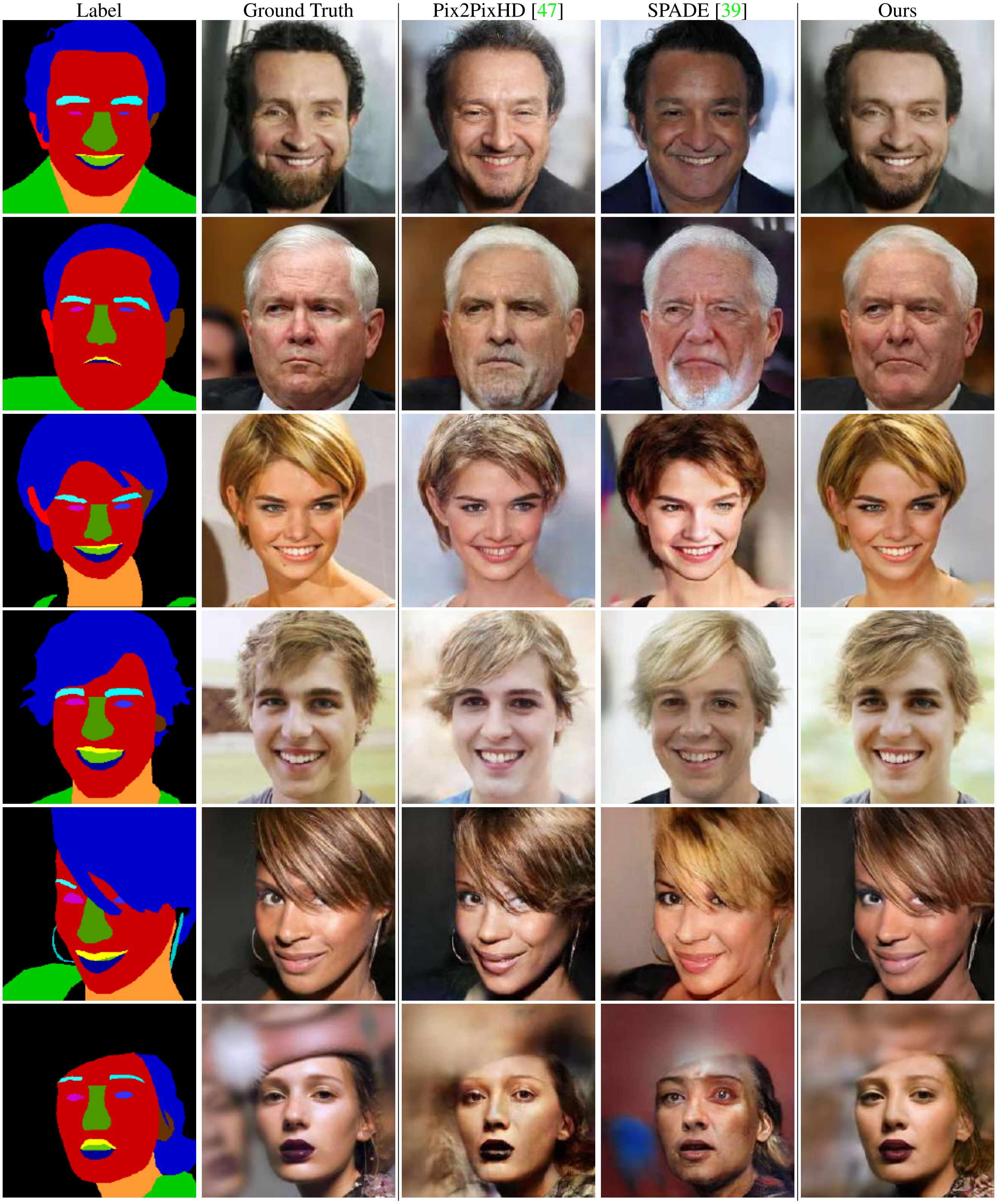


Figure 19: Visual comparison of semantic image synthesis results on the CelebAMask-HQ dataset. We compare Pix2PixHD, SPADE, and our method.

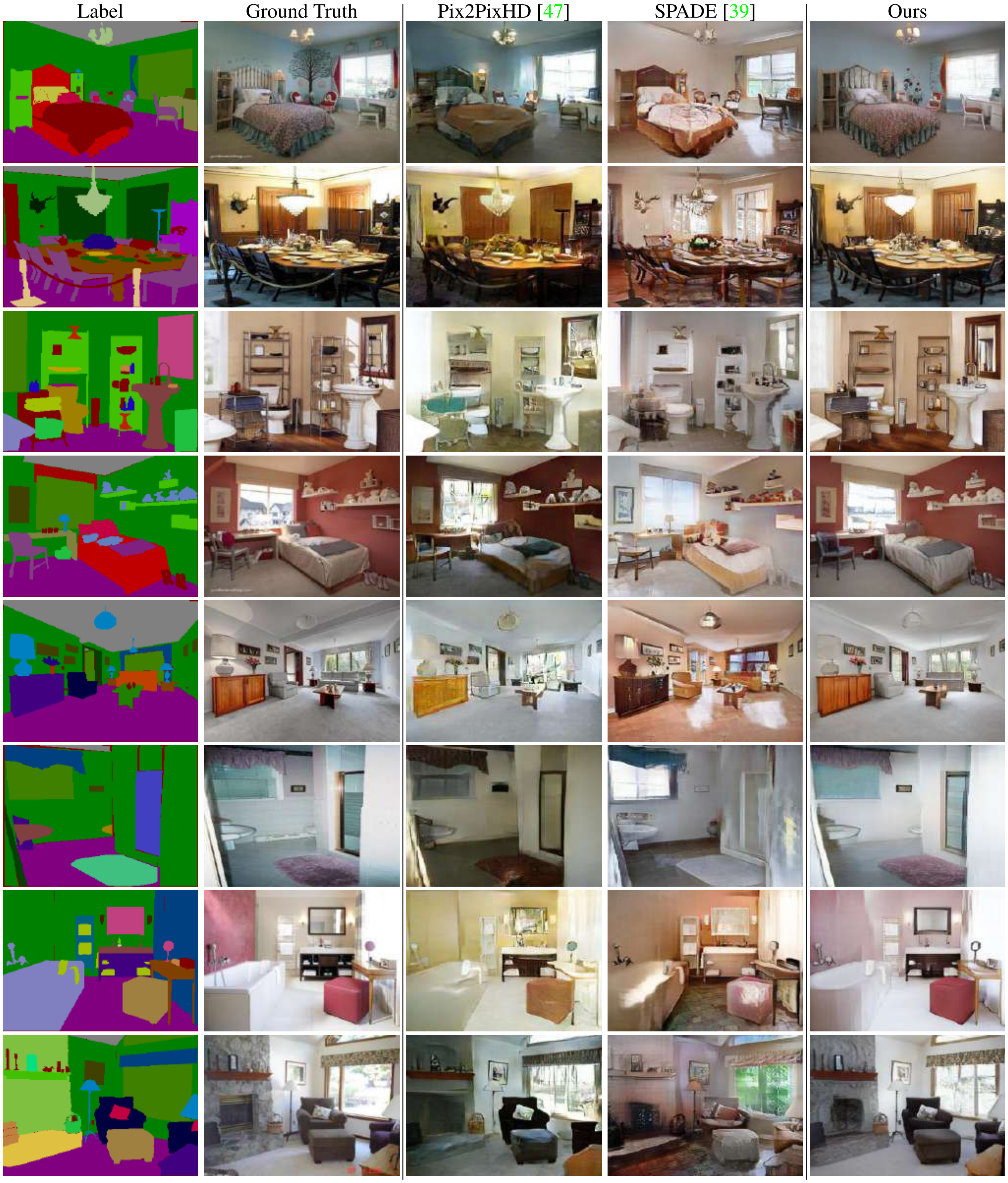


Figure 20: Visual comparison of semantic image synthesis results on the ADE20K dataset. We compare Pix2PixHD, SPADE, and our method.

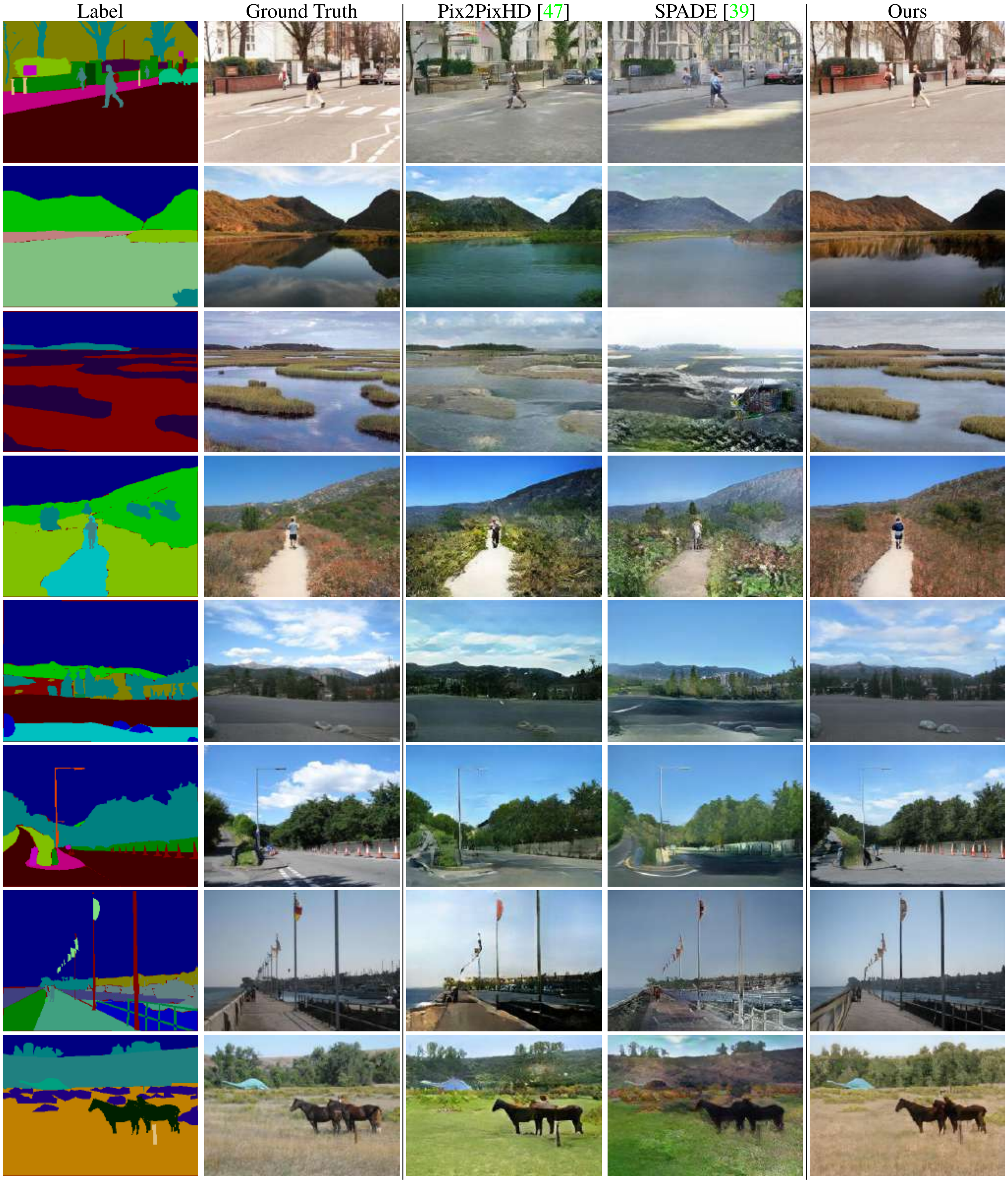


Figure 21: Visual comparison of semantic image synthesis results on the ADE20K dataset. We compare Pix2PixHD, SPADE, and our method.

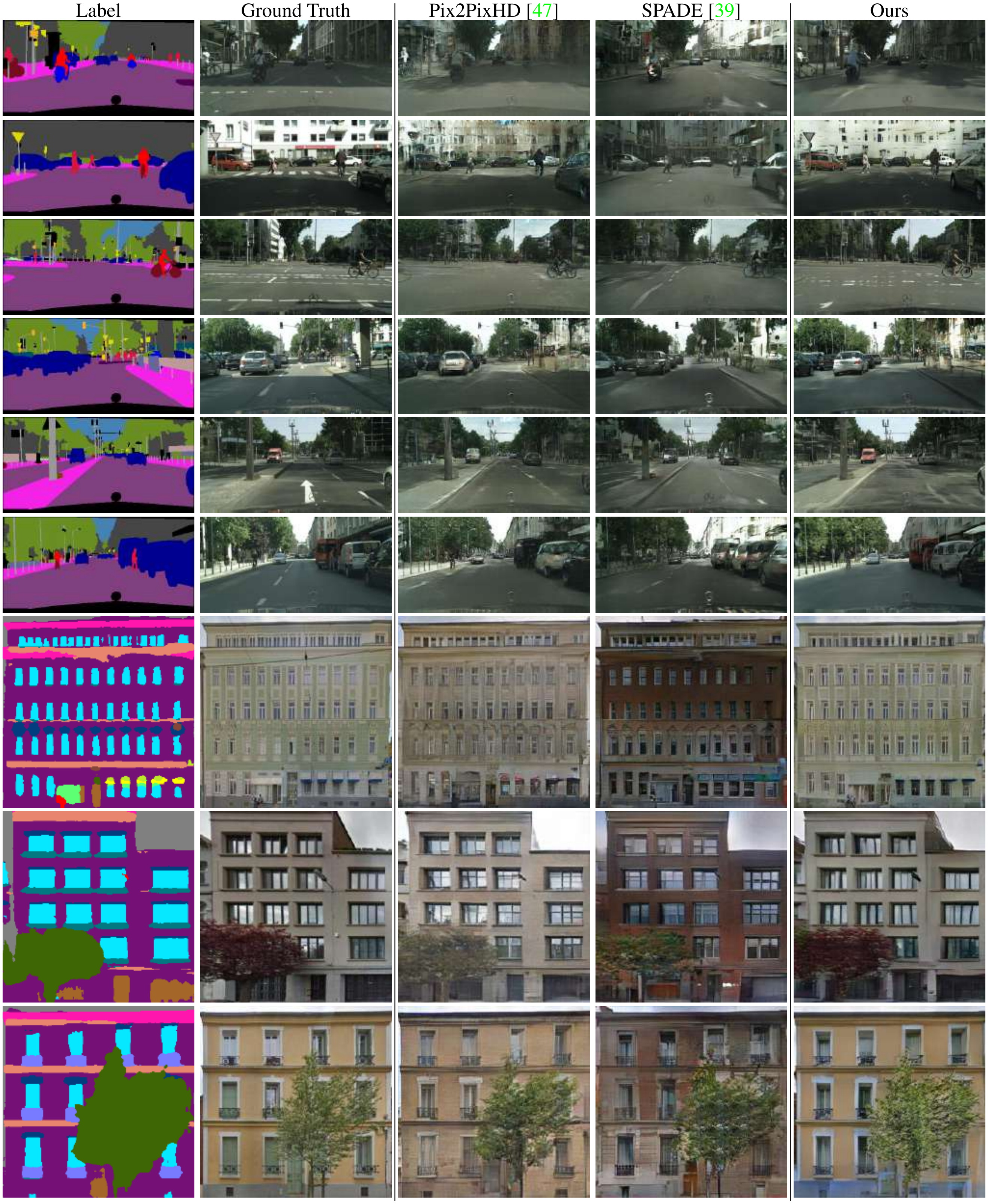


Figure 22: Visual comparison of semantic image synthesis results on the CityScapes and Façades dataset. We compare Pix2PixHD, SPADE, and our method.