Swapping Autoencoder for Deep Image Manipulation

Taesung Park\textsuperscript{1,2} Jun-Yan Zhu\textsuperscript{2} Oliver Wang\textsuperscript{2} Jingwan Lu\textsuperscript{2} Eli Shechtman\textsuperscript{2} Alexei A. Efros\textsuperscript{1,2} Richard Zhang\textsuperscript{2}

\textsuperscript{1}UC Berkeley \hspace{1cm} \textsuperscript{2}Adobe Research

Abstract

Deep generative models have become increasingly effective at producing realistic images from randomly sampled seeds, but using such models for controllable manipulation of existing images remains challenging. We propose the Swapping Autoencoder, a deep model designed specifically for image manipulation, rather than random sampling. The key idea is to encode an image into two independent components and enforce that any swapped combination maps to a realistic image. In particular, we encourage the components to represent structure and texture, by enforcing one component to encode co-occurrent patch statistics across different parts of the image. As our method is trained with an encoder, finding the latent codes for a new input image becomes trivial, rather than cumbersome. As a result, our method enables us to manipulate real input images in various ways, including texture swapping, local and global editing, and latent code vector arithmetic. Experiments on multiple datasets show that our model produces better results and is substantially more efficient compared to recent generative models.

Figure 1: Our Swapping Autoencoder learns to disentangle texture from structure for image editing tasks. One such task is texture swapping, shown here. Please see our project webpage for a demo video of our editing method.

1 Introduction

Deep generative models, such as Generative Adversarial Networks (GANs) \cite{goodfellow2014generative}, have revolutionized image synthesis, enabling photorealistic rendering of complex phenomena \cite{park2019semantic, johnson2016perceptual}. These methods learn a mapping from an easy-to-sample (typically Gaussian) distribution to the image domain, enabling the generation of random images in the target domain. However, in many computer vision and graphics applications, one wants to edit an existing photo. How can we then make deep generative models more useful for these applications in practice?

One promising direction is conditional generative models, which directly synthesize an output conditional on a given input image \cite{rezende2014stochastic, uesato2018connecting, saito2019universal}. However, these approaches require the task to be defined \textit{a priori}, and need extensive training data for each such task. Another method is to retrofit pre-trained unconditional GAN models, by looking for semantically meaningful editing operations in the existing latent space \cite{shrivastava2017learning, lee2018unsupervised, choi2019stargan}. However, it is both challenging and computationally-intensive to project an image in the wild into its corresponding latent code.
Instead, we wish to learn a model designed specifically for image manipulation, rather than random sampling. Our key idea is to use image swapping as a pretext task for learning an embedding space useful for image manipulation. We propose an autoencoder-based model, where a given image is encoded into two separate latent codes, namely structure code and texture code, that are designed to represent structure and texture in a disentangled manner. During training, we swap these two codes between pairs of images and enforce that the resulting hybrid images look realistic. In addition, to encourage a meaningful disentanglement, we enforce images with the same texture code to have the same low-level patch distribution (capturing global texture), by introducing a co-occurrence patch discriminator.

Through a human perceptual study, we validate that the structure code indeed learns to correspond largely to the layout or structure of the scene, while texture codes capture properties about its overall appearance including style. In addition, though only trained for reconstruction, swapping, and patch co-occurrence, the resulting embedding space is dense and amenable to image editing.

We demonstrate three practical applications, all possible with one method: 1) synthesizing new image hybrids given example images (see Figure 1), 2) smooth manipulation of attributes or domain transfer of a given photo through traversing latent "directions", and 3) local manipulation capability. All three applications are possible with the same trained model. Because our architecture contains an encoder, we are able to swap styles in real-time – roughly 4 orders of magnitude faster than previous unconditional models, such as StyleGAN [40].

Finally, we evaluate results on multiple datasets such as LSUN churches and bedrooms [79], FlickrFaces-HQ [40], and newly collected datasets of mountains and waterfalls, using both automatic metrics and human perceptual judgments. We demonstrate the advantage of our method over existing unconditional generative models.

2 Related Work

Conditional generative models, such as image-to-image translation [32, 87], learn to directly synthesize an output image given a user input. Many applications have been successfully built with this framework, including image inpainting [59, 30, 76, 51], photo colorization [82, 47, 84, 21], texture and geometry synthesis [85, 18, 74], sketch2photo [65], semantic image synthesis and editing [73, 62, 10, 58]. Recent methods extend it to multi-domain and multi-modal setting [28, 88, 52, 80, 12]. However, it is challenging to apply such methods to on-the-fly image manipulation, because for each new application and new user input, a new model needs to be trained. We present a framework for both image synthesis and manipulation, in which the task can be defined by one or a small number of examples at run-time. While recent works [66, 67] propose to learn a single-image GANs for image editing, our model can be quickly applied to a test image without extensive computation of single-image training.

Deep image editing via latent space exploration modifies the latent vector of a pre-trained, unconditional generative model (e.g., a GAN [17]) according to the desired user edits. For example, iGAN [86] obtains the latent code using an encoder-based initialization followed by Quasi-Newton optimization, and updates the code according to new user constraints. Similar ideas have been explored in other tasks like image inpainting, face editing, and deblurring [9, 60, 77, 3]. More recently, instead of using the input latent space, GANPaint [4] adapts layers of a pre-trained GAN for each input image and updates layers according to a user’s semantic control [5]. Image2StyleGAN [1] and StyleGAN2 [41] reconstruct the image using an extended embedding space and noise vectors. Our work differs in that we allow the code space to be learned rather than sampled from a fixed distribution, thus making it much more flexible. In addition, we train an encoder together with the generator, which allows for significantly faster reconstruction.

Code swapping in generative models. Deep generative models learn to model the data distribution of natural images [64, 17, 44, 13, 11, 75]. Of special relevance to our work are models that use code swapping during training [55, 26, 35, 68]. Our work differs from them in three aspects. First, while most require human supervision, such as class labels [55], pairwise image similarity [35], or object locations [68], our method is fully unsupervised. Second, our decomposable structure and texture codes allow each factor to control different aspects of the image, and produce higher-quality results when mixed. Note that for our application, image quality and flexible control are critically important, as we focus on image manipulation rather than unsupervised feature learning. Recent image-to-image translation methods also use code swapping but require ground truth domain labels [46, 48, 50]. In concurrent work, Anokhin et al. [2] and ALAE [61] propose models very close to our code swapping scheme for image editing purposes.
Style transfer. Modeling style and content is a classic computer vision and graphics problem [69, 22]. Several recent works revisited the topic using modern neural networks [16, 36, 70], by measuring content using perceptual distance [16, 14], and style as global texture statistics, e.g., a Gram matrix. These methods can transfer low-level styles such as brush strokes, but often fail to capture larger scale semantic structures. Photorealistic style transfer methods further constrain the result to be represented by local affine color transforms from the input image [54, 49, 78], but such methods only allow local color changes. In contrast, our learned decomposition can transfer semantically meaningful structure, such as the architectural details of a church, as well as perform other image editing operations.

3 Method

What is the desired representation for image editing? We argue that such representation should be able to reconstruct the input image easily and precisely. Each code in the representation can be independently modified such that the resulting image both looks realistic and reflects the unmodified code. The representation should also support both global and local image editing.

To achieve the above goals, we train a swapping autoencoder (shown in Figure 2) consisting of an encoder E and a generator G, with the core objectives of 1) accurately reconstructing an image, 2) learning independent components that can be mixed to create a new hybrid image, and 3) disentangling texture from structure by using a patch discriminator that learns co-occurrence statistics of image patches.

3.1 Accurate and realistic reconstruction

In a classic autoencoder [25], the encoder E and generator G form a mapping between image \( x \sim X \subset \mathbb{R}^{D \times W \times 3} \) and latent code \( z \sim Z \). As seen in the top branch of Figure 2, our autoencoder also follows this framework, using an image reconstruction loss:

\[
\mathcal{L}_{\text{rec}}(E,G) = \mathbb{E}_{x \sim X} \left[ \| x - G(E(x)) \|_1 \right].
\]

In addition, we wish for the image to be realistic, enforced by a discriminator D. The non-saturating adversarial loss [17] for the generator G and encoder E is calculated as:

\[
\mathcal{L}_{\text{GAN},\text{rec}}(E,G,D) = \mathbb{E}_{x \sim X} \left[ -\log(D(G(E(x)))) \right].
\]

3.2 Decomposable latent codes

We divide the latent space Z into two components, \( z = (z_s, z_t) \), and enforce that swapping components with those from other images still produces realistic images, using the GAN loss [17].

\[
\mathcal{L}_{\text{GAN,swap}}(E,G,D) = \mathbb{E}_{x^1, x^2 \sim X, x^1 \neq x^2} \left[ -\log(D(G(z^1_s, z^2_t))) \right],
\]

where \( z^1_s, z^2_t \) are the first and second components of \( E(x^1) \), \( E(x^2) \), respectively. Furthermore, as shown in Figure 2, we design the shapes of \( z_s \) and \( z_t \) asymmetrically such that \( z_s \) is a tensor with spatial dimensions, while \( z_t \) is a vector. In our model, \( z_s \) and \( z_t \) are intended to encode structure and texture information, and hence named structure and texture code, respectively, for convenience. At
We first describe our experimental setup. We then evaluate our method on: (1) quickly and accurately generating hybrid images by swapping latent codes. We show that the disentanglement of latent codes into the classic concepts of “style” and “content” is competitive even with style transfer methods that address this specific task [45, 78], while producing more photorealistic results. Furthermore, we observe that even without an explicit objective to encourage it, vector arithmetic in the learned embedding space leads to consistent and plausible image manipulations [8, 40, 34]. This opens up a powerful set of ideas for modeling co-occurrences have been used for propagating a single texture in a supervised setting [74], self-supervised representation learning [33], and identifying image composites [29].

The proposed method can be used to efficiently embed a given image into a factored latent space, and to generate hybrid images by swapping latent codes. We show that the disentanglement of latent codes into the classic concepts of “style” and “content” is competitive even with style transfer methods that address this specific task [45, 78], while producing more photorealistic results. Furthermore, we observe that even without an explicit objective to encourage it, vector arithmetic in the learned embedding space leads to consistent and plausible image manipulations [8, 40, 34]. This opens up a powerful set of operations, such as attribute editing, image translation, and interactive image editing, which we explore.

We first describe our experimental setup. We then evaluate our method on: (1) quickly and accurately embedding a test image, (2) producing realistic hybrid images with a factored latent code that
Figure 3: **Embedding examples and reconstruction quality.** We project images into embedding spaces for our method and baseline GAN models, Im2StyleGAN [1, 40] and StyleGAN2 [41]. Our reconstructions better preserve the detailed outline (e.g., doorway, eye gaze) than StyleGAN2, and appear crisper than Im2StyleGAN. This is verified on average with the LPIPS metric [83]. Our method also reconstructs images much faster than recent generative models that use iterative optimization. See Appendix A for more visual examples.

| Method      | Runtime (sec) | LPIPS Reconstruction |
|-------------|---------------|-----------------------|
|             | Church        | FFHQ                  | Waterfall | Average |
| Ours        | 0.101         | 0.227                 | 0.074     | 0.238   | 0.180   |
| Im2StyleGAN | 495           | 0.186                 | 0.174     | 0.281   | 0.214   |
| StyleGAN2   | 96            | 0.377                 | 0.215     | 0.384   | 0.325   |

Figure 4: **Image swapping.** Each row shows the result of combining the structure code of the leftmost image with the texture code of the top image (trained on LSUN Church and Bedroom). Our model generates realistic images that preserve texture (e.g., material of the building, or the bedsheet pattern) and structure (outline of objects).

corresponds to the concepts of texture and structure, and (3) editability and usefulness of the latent space. We evaluate each aspect separately, with appropriate comparisons to existing methods.

### 4.1 Experimental setup

**Datasets.** For existing datasets, our model is trained on LSUN Churches, Bedrooms [79], Animal Faces HQ (AFHQ) [12], Flickr Faces HQ (FFHQ) [40], all at resolution of 256px except FFHQ at 1024px. In addition, we introduce new datasets, which are Portrait2FFHQ, a combined dataset of 17k portrait paintings from wikiart.org and FFHQ at 256px. Flickr Mountain, 0.5M mountain images from flickr.com, and Waterfall, of 90k 256px waterfall images. Flickr Mountain is trained at 512px resolution, but the model can handle larger image sizes (e.g., 1920×1080) due to the fully convolutional architecture.

**Baselines.** To use a GAN model for downstream image editing, one must embed the image into its latent space [86]. We compare our approach to two recent solutions. Im2StyleGAN [1] present a method for embedding into StyleGAN [40], using iterative optimization into the “W+ -space” of the model. The StyleGAN2 model [41] also includes an optimization-based method to embed into its latent space and noise vectors. One application of this embedding is producing hybrids. StyleGAN and StyleGAN2 present an emergent hierarchical parameter space that allows hybrids to be produced by mixing parameters of two images. We additionally compare to image stylization methods, which aim to mix the “style” of one image with the “content” from another. STROTSS [45] is an optimization-based framework, in the spirit of the classic method of Gatys et al. [16]. We also compare to WCT2 [78], a recent state-of-the-art photorealistic style transfer method based on a feedforward network.
| Method                  | Runtime (sec) | Human Perceptual Study (AMT Fooling Rate) |
|------------------------|---------------|------------------------------------------|
|                        |               | Church       | FFHQ         | Waterfall    | Average |
| Swap Autoencoder (Ours)| 0.113         | 31.3 ± 2.4  | 19.4 ± 2.0  | 41.8 ± 2.2  | 31.0 ± 1.4|
| Im2StyleGAN [1, 30]    |              | 8.5 ± 2.1   | 6.9 ± 1.6   | 12.8 ± 2.4  | 8.4 ± 1.2 |
| StyleGAN2 [41]         | 192           | 24.3 ± 2.2  | 13.8 ± 1.8  | 35.3 ± 2.4  | 24.4 ± 1.4|
| STROTSS [45]           | 166           | 13.7 ± 2.2  | 13.3 ± 1.8  | 23.0 ± 2.4  | 13.5 ± 1.2|
| WCT² [78]              | 1.35          | 27.9 ± 2.3  | 22.3 ± 2.0  | 35.8 ± 2.4  | 28.6 ± 1.3|

Table 1: Realism of swap-generated images. We study how realistic our swap-generated swapped appear, compared to state-of-the-art generative modeling approaches (Im2StyleGAN and StyleGAN2) and stylization methods (STROTSS and WCT²). We run a perceptual study, where each method/dataset is evaluated with 1000 human judgments. We bold the best result per column and bold+italicize methods that are within the statistical significance of the top method. Our method achieves the highest score across all datasets. Note that WCT² is a method tailored especially for photorealistic style transfer and is within the statistical significance of our method in the perceptual study. Runtime is reported for 1024 × 1024 resolution.

4.2 Image embedding

The first step of manipulating an image with a generative model is projecting it into its latent spade. If the input image cannot be projected with high fidelity, the embedded vector cannot be used for editing, as the user would be editing a different image. Figure 3 illustrates both example reconstructions and quantitative measurement of reconstruction quality, using LPIPS [83] between the original and embedded images. Note that our method accurately preserves the doorway pattern (top) and facial features (bottom) without blurriness. Averaged across datasets and on 5 of the 6 comparisons to the baselines, our method achieves better reconstruction quality than the baselines. An exception is on the Church dataset, where Im2StyleGAN obtains a better reconstruction score. Importantly, as our method is designed with test-time embedding in mind, it only requires a single feedforward pass, at least 1000 × faster than the baselines that require hundreds to thousands of optimization steps. Next, we investigate how useful the embedding is by exploring manipulations with the resulting code.

4.3 Swapping to produce image hybrids

In Figure 4, we show example hybrid images with our method, produced by combining structure and texture codes from different images. Note that the textures of the top row of images are consistently transferred; the sky, facade, and window patterns are mapped to the appropriate regions on the structure images on the churches, and similarly for the bedsheets.

Realism of image hybrids. In Table 1, we show results of comparison to existing methods. As well as generative modeling methods [1, 41, 40]. For image hybrids, we additionally compare with SOTA style transfer methods [45, 78], although they are not directly applicable for controllable editing by embedding images (Section 4.5). We run a human perceptual study, following the test setup used in [82, 32, 66]. A real and generated image are shown sequentially for one second each to Amazon
Figure 6: **Style and content.** (Left) Results of our perceptual study where we asked users on AMT to choose which image better reflects the “style” or “content” of a provided reference image, given two results (ours and a baseline). Our model is rated best for capturing style, and second-best for preserving content, behind WCT$^2$ [78], a photorealistic style transfer method. Most importantly, our method was rated strictly better in both style and content matching than both image synthesis models Im2StyleGAN [1, 40] and StyleGAN2 [41]. (Right) Using the self-similarity distance [45] and SIFID [66], we study variations of the co-occurrence discriminator’s patch size in training with respect to the image size. As patch size increases, our model tends to make more changes in swapping (closer to the target style and further from input structure). In addition, we gradually interpolate the texture code, with interpolation ratio $\alpha$, away from a full swapping $\alpha = 1.0$, and observe that the transition is smooth.

Mechanical Turkers (AMT), who choose which they believe to be fake. We measure how often they fail to identify the fake. An algorithm generating perfectly plausible images would achieve a fooling rate of 50%. We gather 15,000 judgments, 1000 for each algorithm and dataset. Our method achieves more realistic results across all datasets. The nearest competitor is the WCT$^2$ [78] method, which is designed for photorealistic style transfer. Averaged across the three datasets, our method achieves the highest fooling rate (31.0±1.4%), with WCT$^2$ closely following within the statistical significance (28.6±1.3%). We show qualitative examples in Figure 5.

**Style and content.** Next, we study how well the concepts of content and style are reflected in the structure and texture codes, respectively. We employ a Two-alternative Forced Choice (2AFC) user study to quantify the quality of image hybrids in content and style space. We show participants our result and a baseline result, with the style or content reference in between. We then ask a user which image is more similar in style, or content respectively. Such 2AFC tests were used to train the LPIPS perceptual metric [83], as well as to evaluate style transfer methods in [45]. As no true automatic perceptual function exists, human perceptual judgments remain the “gold standard” for evaluating image synthesis results [82, 32, 10, 66]. Figure 6 visualizes the result of 3,750 user judgments over four baselines and three datasets, which reveal that our method outperforms all baseline methods with statistical significance in style preservation. For content preservation, our method is only behind WCT$^2$, which is a photorealistic stylization method that makes only minor color modifications to the input. Most importantly, our method achieves the best performance with statistical significance in both style and content among models that can embed images, which is required for other forms of image editing.

### 4.4 Analysis of our method

Next we analyze the behavior of our model using automated metrics. Self-similarity Distance [45] measures structural similarity in deep feature space based on the self-similarity map of ImageNet-pretrained network features. Single-Image FID [66] measures style similarity by computing the Fréchet Inception Distance (FID) between two feature distributions, each generated from a single image. SIFID is similar to Gram distance, a popular metric in stylization methods [16, 15], but differs by comparing the mean of the feature distribution as well as the covariance.

Specifically, we vary the size of cropped patches for the co-occurrence patch discriminator in training. In Figure 6 (right), the max size of random cropping is varied from 1/8 to 3/4 of the image side length, including the default setting of 1/4. We observe that as the co-occurrence discriminator sees larger patches, it enforces stronger constraint, thereby introducing more visual change in both style and content. Moreover, instead of full swapping, we gradually interpolate one texture code to the other. We observe that the SIFID and self-similarity distance both change gradually, in all patch settings. Such gradual visual change can be clearly observed in Figure 7, and the metrics confirm this.

### 4.5 Image editing via latent space operations

Even though no explicit constraint was enforced on the latent space, we find that modifications to the latent vectors cause smooth and predictable transformations to the resulting images. This makes
such a space amenable to downstream editing in multiple ways. First, we find that our representation allows for controllable image manipulations by vector arithmetic in the latent space. Figure 7 shows that adding the same vector smoothly transforms different images into a similar style, such as gradually adding more snow (top). Such vectors can be conveniently derived by taking the mean difference between the embeddings of two groups of images.

In a similar mechanism, the learned embedding space can also be used for image-to-image translation tasks (Figure 7), such as transforming paintings to photos. Image translation is achieved by applying the domain translation vector, computed as the mean difference between the two domains. Compared to most existing image translation methods, our method does not require that all images are labeled, and also allows for multi-domain, fine-grained control simply by modifying the vector magnitude and members of the domain at test time. Finally, the design of the structure code $z_s$ is directly amenable local editing operations, due to its spatial nature; we show additional results in Appendix A.

5 Discussion

The main question we would like to address, is whether unconditional random image generation is required for high-quality image editing tasks. For such approaches, projection becomes a challenging operation, and intuitive disentanglement still remains a challenging question. We show that our method based on an auto-encoder model has a number of advantages over prior work, in that it can accurately embed high-resolution images in real-time, into an embedding space that disentangles texture from structure, and generates realistic output images with both swapping and vector arithmetic. We performed extensive qualitative and quantitative evaluations of our method on multiple datasets. Still, structured texture transfer remains challenging, such as the striped bedsheet of Figure 4. Furthermore, extensive analysis on the nature of disentanglement, ideally using reliable, automatic metrics will be beneficial as future work.

Acknowledgments. We thank Nicholas Kolkin for the helpful discussion on the automated content and style evaluation, Jeongho Seo and Yoseob Kim for advice on the user interface, and William T. Peebles, Tongzhuo Wang, and Yu Sun for the discussion on disentanglement. Taesung Park is supported by a Samsung Scholarship and an Adobe Research Fellowship, and much of this work was done as an Adobe Research intern. This work was supported in part by an Adobe gift.
Broader Impact

Democratizing content creation tools. Tools for creative expression are an important part of human culture. The advent of digital photography and professional editing tools such as Adobe Photoshop, has allowed artists to push creative boundaries. However, many existing tools are complex and require extensive expertise to use effectively. Learning-based content creation tools such as our method, can be used to democratize content creation, allowing novice users to synthesize compelling images.

We built a sample user interface for creative user control over photographs. Figure 8 shows three editing modes that our model supports. Please see a demo video on our webpage. We demonstrate three operations: (1) **global style editing**: the texture code can be transformed by adding predefined manipulation vectors that are computed from PCA on the train set. Like GANSpace [19], the user is provided with knobs to adjust the gain for each manipulation vector. (2) **region editing**: the structure code can also be manipulated the same way of using PCA components, by treating each location as individual, controllable vectors. In addition, masks can be automatically provided to the user based on the self-similarity map at the location of interest to control the extent of structural manipulation. (3) **cloning**: the structure code can be directly edited using a brush that replaces the code from another part of the image, like the Clone Stamp tool of Photoshop.

![UI with input image](image1)
![brush stroke visualization](image2)
![1. remove road](image3)
![2. draw mountain](image4)

**Figure 8: Example Interactive UI.** (top, cloning) using an interactive UI, part of the image is “redrawn” by the user with a brush tool that extracts structure code from user-specified location. (left, region editing) the bottom region is transformed to lake, snow, or different vegetation by adding a manipulation vector to the structure codes of the masked region, which is auto-generated from the self-similarity map at the specified location. (right, global style editing) the overall texture and style can be changed using vector arithmetic with principal directions of PCA, controlled by the sliders on the right pane of the UI. (best viewed zoomed in)

Photo manipulation detection. While our goal is to support artistic and creative applications, the use of such technology for purposes of deception – posing generated images as real photographs – is a concern that is especially relevant today. Work in deep image forensics provides a potential mitigation strategy. In particular, Wang et al. [72] recently showed that a classifier trained to classify between real photographs and synthetic images generated by ProGAN [39], was able to generalize to other generators, among them, StyleGAN [40] and StyleGAN2 [41]. We perform a small investigation and show results in Table 2. We run the off-the-shelf detector from [72], specifically, the Blur-JPEG(0.5) variant on the full, uncropped result images from this paper, and evaluate whether they are correctly classified as “synthesized”. For the most sensitive category, FFHQ faces, both previous generative models and our method have high detectability. We observe similar behavior, albeit with some dropoff on less sensitive categories of “church” and “waterfall”. This indicates that our method shares enough architectural components with previous methods to be detectable. However, these detection methods do not work at 100%, and performance can degrade as the images are degraded in the wild (e.g., compressed, rescanned) or via adversarial attacks. As a result, the problem of verifying image provenance remains a significant challenge to society that requires multiple layers of solutions, from technical (such as learning-based detection systems or authenticity certification chains), to social, such as efforts to increase public awareness of the problem, to regulatory and legislative.
| Method                        | Task       | Church | FFHQ | Waterfall | Average |
|-------------------------------|------------|--------|------|-----------|---------|
| Im2StyleGAN [1, 40]           | reconstruct| 99.3   | 100.0| 92.4      | 97.2    |
|                               | swap       | 100.0  | 100.0| 97.7      | 99.2    |
| StyleGAN2 [41]                | reconstruct| 97.7   | 100.0| 94.3      | 98.0    |
|                               | swap       | 99.8   | 100.0| 96.6      | 98.8    |
| Swap Autoencoder (Ours)       | reconstruct| 93.6   | 95.6 | 73.9      | 87.7    |
|                               | swap       | 96.6   | 94.7 | 80.4      | 90.5    |

Table 2: Detectability. We run the CNN-generated image detector from Wang et al. [72] and report average precision (AP); chance is 50%. The CNN classifier is trained from ProGAN [39], the predecessor to StyleGAN [40]. Because our method shares architectural components, a classifier trained to detect a different method can also generalize to ours, with some dropoff, especially for the waterfall class. Notably, the performance on FFHQ faces remains high. However, performance is not reliably at 100% across all methods, indicating that future detection methods could potentially benefit from training on our method.

References

[1] Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: IEEE International Conference on Computer Vision (ICCV) (2019) 2, 5, 6, 7, 10, 15, 19
[2] Anokhin, I., Solovev, P., Korzhenko, D., Kharlamov, A., Khakhulin, T., Silvestrov, A., Nikolenko, S., Lempitsky, V., Sterkin, G.: High-resolution daytime translation without domain labels. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) 2
[3] Asim, M., Shamshad, F., Ahmed, A.: Blind image deconvolution using deep generative priors. arXiv preprint arXiv:1802.04073 (2018) 2
[4] Bau, D., Strobelt, H., Peebles, W., Wulff, J., Zhou, B., Zhu, J.Y., Torralba, A.: Semantic photo manipulation with a generative image prior. ACM Transactions on Graphics (TOG) 38(4), 1–11 (2019) 1, 2
[5] Bau, D., Zhu, J.Y., Strobelt, H., Bolei, Z., Tenenbaum, J.B., Freeman, W.T., Torralba, A.: Gan dissection: Visualizing and understanding generative adversarial networks. In: International Conference on Learning Representations (ICLR) (2019) 2
[6] Bengio, Y.: Deep learning of representations: Looking forward. In: International Conference on Statistical Language and Speech Processing (2013) 4
[7] Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. In: International Conference on Learning Representations (ICLR) (2019) 1
[8] Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. In: International Conference on Learning Representations (ICLR) (2019) 4
[9] Brock, A., Lim, T., Ritchie, J.M., Weston, N.: Neural photo editing with introspective adversarial networks. In: International Conference on Learning Representations (ICLR) (2017) 1, 2
[10] Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: IEEE International Conference on Computer Vision (ICCV) (2017) 2, 7
[11] Chen, X., Duan, Y., Houhoothft, R., Schulman, J., Sutskever, I., Abbeel, P.: Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In: Advances in Neural Information Processing Systems (2016) 2, 4
[12] Choi, Y., Uh, Y., Yoo, J., Ha, J.W.: Stargan v2: Diverse image synthesis for multiple domains. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020) 2, 5, 8, 15, 22
[13] Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. In: International Conference on Learning Representations (ICLR) (2017) 2, 4
[14] Dosovitskiy, A., Brox, T.: Generating images with perceptual similarity metrics based on deep networks. In: Advances in Neural Information Processing Systems (2016) 3
[15] Gatys, L., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: Advances in Neural Information Processing Systems (2015) 4, 7
[16] 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) 3, 5, 7
[17] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems (2014) 1, 2, 3, 4, 22

[18] Guérin, E., Digne, J., Galin, E., Peytavie, A., Wolf, C., Benes, B., Martinez, B.: Interactive example-based terrain authoring with conditional generative adversarial networks. ACM Transactions on Graphics (TOG) 36(6) (2017) 2

[19] Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: Ganspace: Discovering interpretable gan controls. In: arXiv preprint arXiv:2004.02546 (2020) 9

[20] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 4, 21

[21] He, M., Chen, D., Liao, J., Sander, P.V., Yuan, L.: Deep exemplar-based colorization. ACM Transactions on Graphics (TOG) 37(4), 1–16 (2018) 2

[22] Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image analogies. In: ACM Transactions on Graphics (TOG) (2001) 3

[23] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In: Advances in Neural Information Processing Systems (2017) 20

[24] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., Lerchner, A.: beta-vae: Learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations (ICLR) (2017) 4

[25] Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006) 3

[26] Hu, Q., Szabó, A., Portenier, T., Favaro, P., Zwicker, M.: Disentangling factors of variation by mixing them. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018) 2

[27] Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: IEEE International Conference on Computer Vision (ICCV) (2017) 4

[28] Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. European Conference on Computer Vision (ECCV) (2018) 2

[29] Huh, M., Liu, A., Owens, A., Efros, A.A.: Fighting fake news: Image splice detection via learned self-consistency. In: European Conference on Computer Vision (ECCV) (2018) 4

[30] Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Transactions on Graphics (TOG) 36(4), 107 (2017) 2

[31] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (ICML) (2015) 21

[32] 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) 1, 2, 6, 7

[33] Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Learning visual groups from co-occurrences in space and time. arXiv preprint arXiv:1511.06811 (2015) 4

[34] Jahanian, A., Chai, L., Isola, P.: On the "steerability" of generative adversarial networks. In: International Conference on Learning Representations (ICLR) (2020) 4

[35] Jha, A.H., Anand, S., Singh, M., Veeravasarapu, V.: Disentangling factors of variation with cycle-consistent variational auto-encoders. In: European Conference on Computer Vision (ECCV) (2018) 2

[36] Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision (ECCV) (2016) 3

[37] Julesz, B.: Visual pattern discrimination. IRE transactions on Information Theory 8(2), 84–92 (1962) 4

[38] Julesz, B., Gilbert, E.N., Shepp, L.A., Frisch, H.L.: Inability of humans to discriminate between visual textures that agree in second-order statistics—revisited. Perception 2(4), 391–405 (1973) 4

[39] Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. In: International Conference on Learning Representations (ICLR) (2018) 9, 10

11
[40] Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 1, 2, 4, 5, 6, 7, 9, 10, 15, 19, 22

[41] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 2, 4, 5, 6, 7, 9, 10, 15, 19, 21, 22

[42] Kim, H., Mnih, A.: Disentangling by factorising. In: International Conference on Machine Learning (ICML) (2018) 4

[43] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015) 22

[44] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (ICLR) (2014) 2, 4

[45] Kolkin, N., Salavon, J., Shakhnarovich, G.: Style transfer by relaxed optimal transport and self-similarity. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 4, 5, 6, 7, 15, 19, 20

[46] Kotovenko, D., Sanakoyeu, A., Lang, S., Ommer, B.: Content and style disentanglement for artistic style transfer. In: IEEE International Conference on Computer Vision (ICCV) (2019) 2

[47] Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: European Conference on Computer Vision (ECCV) (2016) 2

[48] Lee, H.Y., Tseng, H.Y., Huang, J.B., Singh, M.K., Yang, M.H.: Diverse image-to-image translation via disentangled representation. In: European Conference on Computer Vision (ECCV) (2018) 2

[49] Li, Y., Liu, M.Y., Li, X., Yang, M.H., Kautz, J.: A closed-form solution to photorealistic image stylization. In: European Conference on Computer Vision (ECCV) (2018) 3

[50] Lin, J., Chen, Z., Xia, Y., Liu, S., Qin, T., Luo, J.: Exploring explicit domain supervision for latent space disentanglement in unpaired image-to-image translation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2019) 2

[51] Liu, G., Reda, F.A., Shih, K.J., Wang, T.C., Tao, A., Catanzaro, B.: Image inpainting for irregular holes using partial convolutions. In: European Conference on Computer Vision (ECCV) (2018) 2

[52] Liu, M.Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz, J.: Few-shot unsupervised image-to-image translation. In: IEEE International Conference on Computer Vision (ICCV) (2019) 2

[53] Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: IEEE International Conference on Computer Vision (ICCV) (2015) 22

[54] Luan, F., Paris, S., Shechtman, E., Bala, K.: Deep photo style transfer. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 3

[55] Mathieu, M.F., Zhao, J.J., Zhao, J., Ramesh, A., Sprechmann, P., LeCun, Y.: Disentangling factors of variation in deep representation using adversarial training. In: Advances in Neural Information Processing Systems (2016) 2

[56] Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: International Conference on Machine Learning (ICML) (2018) 22

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

[58] Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 2

[59] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., Efros, A.A.: Context encoders: Feature learning by inpainting. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 2

[60] Perarnau, G., van de Weijer, J., Raducanu, B., Alvarez, J.M.: Invertible conditional gans for image editing. In: NIPS Workshop on Adversarial Training (2016) 2

[61] Pudhorsky, S., Adjeroh, D.A., Doretto, G.: Adversarial latent autoencoders. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 2

[62] Portenier, T., Hu, Q., Szabó, A., Bigdeli, S.A., Favaro, P., Zwicker, M.: Faceshop: Deep sketch-based face image editing. ACM Transactions on Graphics (TOG) 37(4) (2018) 2
[63] Portilla, J., Simoncelli, E.P.: A parametric texture model based on joint statistics of complex wavelet coefficients. International journal of computer vision 40(1), 49–70 (2000)

[64] Salakhutdinov, R., Hinton, G.: Deep boltzmann machines. In: Artificial intelligence and statistics. pp. 448–455 (2009)

[65] Sangkloy, P., Lu, J., Fang, C., Yu, F., Hays, J.: Scribbler: Controlling deep image synthesis with sketch and color. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[66] Shaham, T.R., Dekel, T., Michaeli, T.: Singan: Learning a generative model from a single natural image. In: IEEE International Conference on Computer Vision (ICCV) (2019)

[67] Shocher, A., Bagon, S., Isola, P., Irani, M.: Ingen: Capturing and remapping the “dna” of a natural image. In: IEEE International Conference on Computer Vision (ICCV) (2019)

[68] Singh, K.K., Ojha, U., Lee, Y.J.: Finegan: Unsupervised hierarchical disentanglement for fine-grained object generation and discovery. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

[69] Tenenbaum, J.B., Freeman, W.T.: Separating style and content with bilinear models. Neural computation 12(6), 1247–1283 (2000)

[70] Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V.: Texture networks: Feed-forward synthesis of textures and stylized images. In: International Conference on Machine Learning (ICML) (2016)

[71] Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)

[72] Wang, S.Y., Wang, O., Zhang, R., Owens, A., Efros, A.A.: Cnn-generated images are surprisingly easy to spot... for now. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)

[73] Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[74] Xian, W., Sangkloy, P., Agrawal, V., Raj, A., Lu, J., Fang, C., Yu, F., Hays, J.: Texturegan: Controlling deep image synthesis with texture patches. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[75] Xing, X., Han, T., Gao, R., Zhu, S.C., Wu, Y.N.: Unsupervised disentangling of appearance and geometry by deformable generator network. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

[76] Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O., Li, H.: High-resolution image inpainting using multi-scale neural patch synthesis. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[77] Yeh, R.A., Chen, C., Yian Lim, T., Schwing, A.G., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with deep generative models. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[78] Yoo, J., Uh, Y., Chun, S., Kang, B., Ha, J.W.: Photorealistic style transfer via wavelet transforms. In: IEEE International Conference on Computer Vision (ICCV) (2019)

[79] Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)

[80] Yu, X., Chen, Y., Liu, S., Li, T., Li, G.: Multi-mapping image-to-image translation via learning disentanglement. In: Advances in Neural Information Processing Systems (NeurIPS) (2019)

[81] Zhang, R.: Making convolutional networks shift-invariant again. In: International Conference on Machine Learning (ICML) (2019)

[82] Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: European Conference on Computer Vision (ECCV) (2016)

[83] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[84] Zhang, R., Zhu, J.Y., Isola, P., Geng, X., Lin, A.S., Yu, T., Efros, A.A.: Real-time user-guided image colorization with learned deep priors. ACM Transactions on Graphics (TOG) 9(4) (2017)
[85] Zhou, Y., Zhu, Z., Bai, X., Lischinski, D., Cohen-Or, D., Huang, H.: Non-stationary texture synthesis by adversarial expansion. ACM Transactions on Graphics (TOG) 37(4) (2018) 2

[86] Zhu, J.Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Generative visual manipulation on the natural image manifold. In: European Conference on Computer Vision (ECCV) (2016) 1, 2, 5

[87] Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: IEEE International Conference on Computer Vision (ICCV) (2017) 1, 2

[88] Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shechtman, E.: Toward multimodal image-to-image translation. In: Advances in Neural Information Processing Systems (2017) 2
Appendix A  Results and Comparisons

A.1 Additional visual results

In Figure 1, 4, and 7 of the main paper, we have shown our results of swapping the texture and structure codes as well as manipulation results of the latent space. Here we show additional swapping and editing results.

Swapping. Here we show additional results of swapping on FFHQ (Figure 9), Mountains (Figure 12), and LSUN Church and Bedroom (Figure 14) dataset. For test images, the input images for the models trained on FFHQ (Figure 9, 10, and 11) and Mountains (Figure 12 and 13) are separately downloaded from pixabay.com using relevant keywords. The results on LSUN (Figure 14) are from the validation sets [79].

Editing. The latent space of our method can be used for image editing. For example, in Figure 11 and 13, we show the result of editing the texture code using an interactive UI that performs vector arithmetic using the PCA components. Editing the texture code results in changing global attributes like age, wearing glasses, lighting, and background in the FFHQ dataset (Figure 11), and time of day and grayscale in the Mountains dataset (Figure 13). On the other hand, editing the structure code can manipulate locally isolated attributes such as eye shape, gaze direction (Figure 10), or texture of the grass field (Figure 13). These results are generated by performing vector arithmetic in the latent space of the flattened structure code, masked by the region specified by the user in the UI, similar to region editing of Figure 8. In addition, the pond of Figure 13 is created by overwriting the structure code with the code of a lake from another image (cloning of Figure 8). More editing results of using the interactive UI can be found on our project webpage: https://taesungp.github.io/SwappingAutoencoder.

User-guided image translation. In Figure 16, we show the results of user-guided image translation, trained on Portrait2FFHQ and Animal Faces HQ [12]. For each dataset, the results are produced using the model trained on the mix of all domains and hence without any domain labels. By adjusting the gains on the principal components of the texture code with the interactive UI, the user controls the magnitude and style of translation. Interestingly, we found that the first principal axis of the texture code largely corresponds to the domain translation vector in the case of Portrait2FFHQ and AFHQ dataset, with the subsequent vectors controlling more fine-grained styles. Therefore, our model is suitable for the inherent multi-modal nature of image translation. For example, in Figure 16, the input cat and dog images are translated into six different plausible outputs.

A.2 Additional comparison to existing methods

In Table 3, we report the FIDs of the swapping results of our model and baselines on LSUN Church, FFHQ, and Waterfall datasets using the validation set. More visual comparison results that extend Figure 3 and 5 of the main paper are in Figure 15. Note that using FID to evaluate the results of this task is not sufficient, as it does not capture the relationship to input content and style images. For example, a low FID can be achieved simply by not making large changes to the input content image. Our model achieves the second-best FID, behind the photorealistic style transfer method WCT [78]. However, the visual results of Figure 15 and human perceptual study of Figure 6 reveal that our method better captures the details of the reference style. In Table 4, we compare the FIDs of swapping on the training set with unconditionally generated StyleGAN and StyleGAN2 outputs. Note that randomly sampled images of StyleGAN and StyleGAN2 are not suitable for image editing, as it ignores the input image. The FID of swap-generated images of our method is placed between the FID of unconditionally generated StyleGAN and StyleGAN2 images.

| Method                  | Church | FFHQ | Waterfall | Mean |
|-------------------------|--------|------|-----------|------|
| Swap Autoencoder (Ours) | 3.91   | 3.48 | 3.04      | 3.04 |
| StyleGAN [40]           | 4.21   | 4.40 | 6.09      | 5.03 |
| StyleGAN2 [41]          | 3.86   | 2.84 | 2.67      | 2.80 |

Table 3: FID of swapping on the validation set. We compare the FIDs of content-style mixing on the validation sets. Note that the utility of FID is limited in our setting, since it does not capture the quality of embedding or disentanglement. Our method achieves second-lowest FID, behind WCT [78], a photorealistic style transfer method. Note that the values are not directly comparable to different datasets or to the training splits (Table 4), since the number of samples is different. Please see Figure 15 for visual results.

| Method                  | Church | FFHQ | Waterfall |
|-------------------------|--------|------|-----------|
| Swap Autoencoder (Ours) | 3.91   | 3.48 | 3.04      |
| StyleGAN [40]           | 4.21   | 4.40 | 6.09      |
| StyleGAN2 [41]          | 3.86   | 2.84 | 2.67      |

Table 4: FID of swapping on the training set, in the context of unconditional GAN. We compute the FID of swapped images on the training set, and compare it with FIDs of unconditionally generated images of StyleGAN [40] and StyleGAN2 [41]. The result conveys how much realism the swap-generated images convey. Note that randomly sampled images of StyleGAN [40] and StyleGAN2 [41] models are not suitable for image editing. Asterisk(∗) denotes FIDs reported in the original papers.
Figure 9: **Swapping results of our FFHQ model.** The input photographs are collected from pixabay.com.

Figure 10: **Region editing.** The results are generated by performing vector arithmetic on the structure code. The vectors are discovered by a user with our UI, with each goal in mind.

Figure 11: **Global editing.** The results are generated using vector arithmetic on the texture code. The vectors are discovered by a user with our UI, with each goal in mind.
Figure 12: **Swapping results of our method trained on Flickr Mountains.** The model is trained and tested at 512px height.

Figure 13: **User editing results of our method trained on Flickr Mountains.** For the input image in red, the top and bottom rows show examples of editing the structure and texture code, respectively. Please refer to Figure 8 on how editing is performed. The image is of 1536×1020 resolution, using a model trained at 512px resolution.
Figure 14: Swapping results of LSUN Churches (top) and Bedrooms (bottom) validation set. The model is trained with 256px-by-256px crops and tested at 256px resolution on the shorter side, keeping the aspect ratio.
Figure 15: **Comparison to existing methods.** Random results on LSUN Churches and Flickr Waterfall are shown. In each block, we show both the reconstruction and swapping for ours, Im2StyleGAN [1, 40], and StyleGAN2 [41], as well as the style transfer results of STROTSS [45] and WCT2 [78]. Im2StyleGAN has a low reconstruction error but performs poorly on the swapping task. StyleGAN2 generates realistic swappings, but fails to capture the input images faithfully. Both style transfer methods makes small changes to the input structure images.
A.3 Corruption study of Self-Similarity Distance and SIFID

In Figure 17, we validate our usage of Self-Similarity Matrix Distance [45] and Single-Image FID (SIFID) [66] as automated metrics for measuring distance in structure and style. Following FID [23], we study the change in both metrics under predefined corruptions. We find that the self-similarity distance shows a larger variation for image translation and rotation than blurring or adding white noise. In contrast, SIFID is more sensitive to blurring or white noise than translation or rotation. This confirms that the self-similarity captures structure, and SIFID captures style.
Appendix B  Implementation details

We show our architecture designs, additional training details, and provide information about our datasets.

B.1 Architecture

The encoder maps the input image to structure and texture codes, as shown in Figure 18 (left). For the structure code, the network consists of 4 downsampling residual blocks [20], followed by two convolution layers. For the texture code, the network branches off and adds 2 convolutional layers, followed by an average pooling (to completely remove spatial dimensions) and a dense layer. The asymmetry of the code shapes is designed to impose an inductive bias and encourage decomposition into orthogonal tensor dimensions. Given an 256 $\times$ 256 image, the structure code is of dimension 16 $\times$ 16 $\times$ 8 (large spatial dimension), and texture code is of dimension 1 $\times$ 1 $\times$ 2048 (large channel dimension).

The texture code is designed to be agnostic to positional information by using reflection padding or no padding (“valid”) in the convolutional layers (rather than zero padding) followed by average pooling. On the other hand, each location of the structure code has a strong inductive bias to encode information in its neighborhood, due to its fully convolutional architecture and limited receptive field.

The generator maps the codes back to an image, as shown in Figure 18 (right). The network uses the structure code in the main branch, which consists of 4 residual blocks and 4 upsampling residual blocks. The texture code is injected using the weight modulation/demodulation layer from StyleGAN2 [41]. We generate the output image by applying a convolutional layer at the end of the residual blocks. This is different from the default setting of StyleGAN2, which uses an output skip, but more similar to the residual net setting of StyleGAN2 discriminator. Lastly, to enable isolated local editing, we avoid normalizations such as instance or batch normalization [71, 31].

The discriminator architecture is identical to StyleGAN2, except with no minibatch discrimination, to enable easier fine-tuning at higher resolutions with smaller batch sizes.

The co-occurrence patch discriminator architecture is shown in Figure 19 and is designed to determine if a patch in question (“real/fake patch”) is from the same image as a set of reference patches. Each patch is first independently encoded with 5 downsampling residual blocks, 1 residual block, and 1 convolutional layer. The representations for the reference patches are averaged together and concatenated with the representation of the real/fake patch. The classification applies 3 dense layers to output the final prediction.

The detailed design choices of the layers in all the networks follow StyleGAN2 [41], including weight demodulation, antialiased bilinear down/upsampling [81], equalized learning rate, noise injection at every layer, adjusting variance of residual blocks by the division of $\sqrt{2}$, and leaky ReLU with slope 0.2.

Figure 18: Encoder and generator architecture. The encoder network first applies 4 downsampling residual blocks [20] to produce an intermediate tensor, which is then passed to two separate branches, producing the structure code and texture code. The structure code is produced by applying 1-by-1 convolutions to the intermediate tensor. The texture code is produced by applying strided convolutions, average pooling, and then a dense layer. Given an $H \times H$ image, the shapes of the two codes are $H/16 \times H/16 \times 8$, and 1 $\times$ 1 $\times$ 2048, respectively. The case for a 512 $\times$ 512 image is shown. To prevent the texture code from encoding positional information, we apply reflection padding for the residual blocks, and then no padding for the conv blocks. The generator consists of 4 residual blocks and then 4 upsampling residual blocks, followed by 1-by-1 convolution to produce an RGB image. The structure code is given in the beginning of the network, and the texture code is provided at every layer as modulation parameters. We use zero padding for the generator. The detailed architecture follows StyleGAN2 [41], including weight demodulation, bilinear upsampling, equalized learning rate, noise injection at every layer, adjusting variance of residual blocks by the division of $\sqrt{2}$, and leaky ReLU with slope 0.2.
B.2 Training details

At each iteration, we sample a minibatch of size \( N \) and produce \( N/2 \) reconstructed images and \( N/2 \) hybrid images. The reconstruction loss is computed using \( N/2 \) reconstructed images. The loss for the image discriminator is computed on the real, reconstructed, and hybrid images, using the adversarial loss \( \mathbb{E}[-\log(D(x))] + \mathbb{E}[-\log(1 - D(x_{\text{fake}}))] \), where \( x \) and \( x_{\text{fake}} \) are real and generated (both reconstructed and hybrid) images, respectively. For the details of the GAN loss, we follow the setting of StyleGAN2 \cite{Karras2020}, including the non-saturating GAN loss \cite{Goodfellow2014} and lazy R1 regularization \cite{Karras2020, Karras2020-2}. In particular, R1 regularization is also applied to the co-occurrence patch discriminator. The weight for R1 regularization was 10.0 for the image discriminator (following the setting of \cite{Karras2020, Karras2020-2}) and 1.0 for the co-occurrence discriminator. Lastly, the co-occurrence patch discriminator loss is computed on random crops of the real and swapped images. The size of the crops are randomly chosen between 1/8 and 1/4 of the image dimensions for each side, and are then resized to 1/4 of the original image. For each image (real or fake), 8 crops are made, producing 8\( N \) predictions at each iteration. Both discriminators use the binary cross-entropy GAN loss.

We use ADAM \cite{Kingma2014} with 0.002 learning rate, \( \beta_1 = 0.0 \) and \( \beta_2 = 0.99 \). We use the maximum batch size that fits in memory on 8 16GB Titan V100 GPUs: 64 for images of 256\( \times \)256 resolution, 16 for 512\( \times \)512 resolution, and 16 for 1024\( \times \)1024 resolution (with smaller network capacity). Note that only the FFHQ dataset was trained at 1024\( \times \)1024 resolution; for the landscape datasets, we take advantage of the fully convolutional architecture and train with cropped images of size 512\( \times \)512, and test on the full image. The weights on each loss term are simply set to be all 1.0 among the reconstruction, image GAN, and co-occurrence GAN loss.

B.3 Datasets

Here we describe our datasets in more detail.

**LSUN Church** \cite{Leddin2015} consists of 126,227 images of outdoor churches. The images are in the dataset are 256\( \times \)256 resolution, and finetuned at 1024\( \times \)1024 resolution. The dataset designated 10,000 images for validation, but we train our model on the entire 70,000 images, following the practice of StyleGAN \cite{Karras2018} and StyleGAN2 \cite{Karras2020}. For evaluation, we used randomly selected 200 images from the validation set, although the models are trained with these images.

**Flickr Faces HQ** \cite{Karras2018} consists of 70,000 high resolution aligned face images from flickr.com. Our model is initially trained at 512\( \times \)512 resolution, and finetuned at 1024 resolution. The dataset designated 10,000 images for validation, but we train our model on the entire 70,000 images, following the practice of StyleGAN \cite{Karras2018} and StyleGAN2 \cite{Karras2020}. For evaluation, we used randomly selected 200 images from the validation set, although the models are trained with these images.

**Animal Faces HQ** \cite{Karras2018} contains a total of 15,000 images equally split between cats, dogs, and a wildlife category. Our method is trained at 256\( \times \)256 resolution on the combined dataset without domain labels. The results are shown with a separate validation set.

**Portrait2FFHQ** consists of FFHQ \cite{Karras2020} and a newly collected 19,863 portrait painting images from vikiar.t.org. The model is trained at 512\( \times \)512 resolution on the combined dataset. The results of the paper are generated from separately collected sample paintings. We did not check if the same painting belongs in the training set. The test photographs are from CelebA \cite{Kanade2015}. All images are aligned to match the facial landmarks of FFHQ dataset.
Flickr Waterfall is a newly collected dataset of 90,345 waterfall images. The images are downloaded from the user group “Waterfalls around the world” on flickr.com. The validation set is 399 images collected from the user group “*Waterfalls*”. Our model is trained at 256 × 256 resolution.

Flickr Mountains is a newly collected dataset of 517,980 mountain images from Flickr. The images are downloaded from the user group “Mountains Anywhere” on flickr.com. For testing, separately downloaded sample images were used. Our model is trained at 512 × 512 resolution.