Image Generators with Conditionally-Independent Pixel Synthesis

I. Anokhin[1,2], K. Demochkin[1,2], T. Khakhulin[1,2], G. Sterkin[1], V. Lempitsky[1,2], D. Korzhenkov[1]

[1]Samsung AI Center, Moscow
[2]Skolkovo Institute of Science and Technology, Moscow

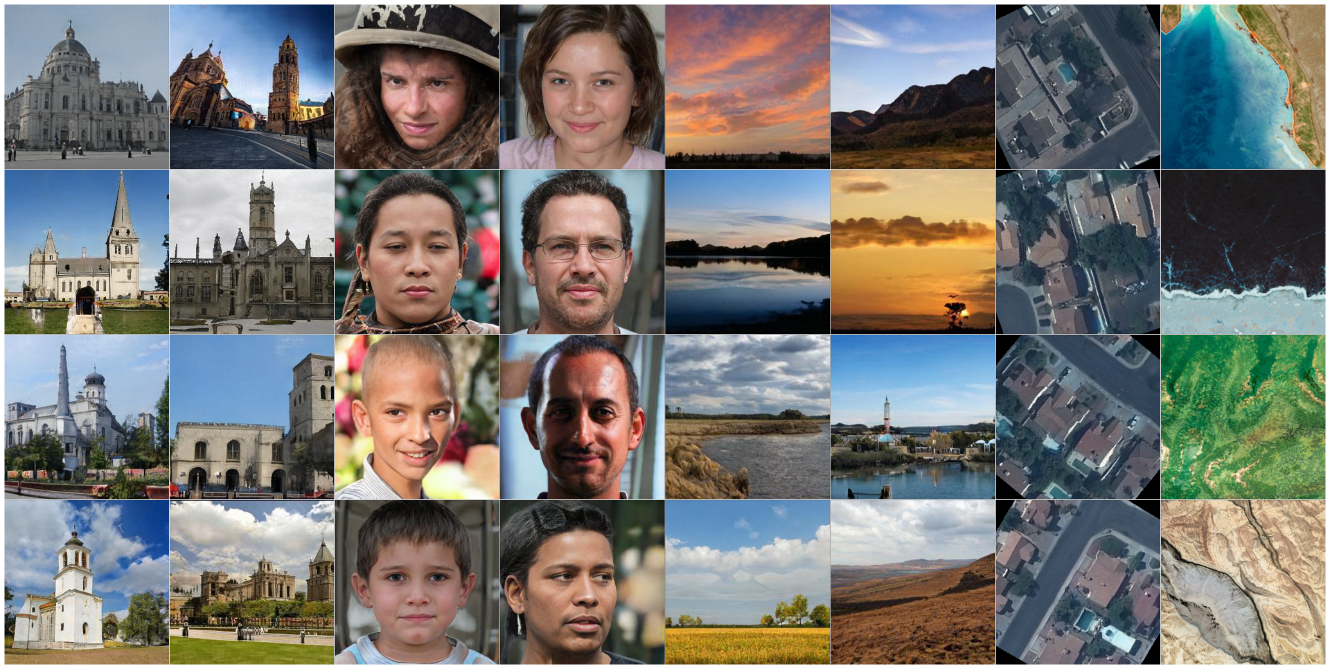


Figure 1: Samples from our generators trained on several challenging datasets (LSUN Churches, FFHQ, Landscapes, Satellite-Buildings, Satellite-Landscapes) at resolution 256 $\times$ 256. The images are generated without spatial convolutions, upsampling, or self-attention operations. No interaction between pixels takes place during inference.

Abstract

Existing image generator networks rely heavily on spatial convolutions and, optionally, self-attention blocks in order to gradually synthesize images in a coarse-to-fine manner. Here, we present a new architecture for image generators, where the color value at each pixel is computed independently given the value of a random latent vector and the coordinate of that pixel. No spatial convolutions or similar operations that propagate information across pixels are involved during the synthesis. We analyze the modeling capabilities of such generators when trained in an adversarial fashion, and observe the new generators to achieve similar generation quality to state-of-the-art convolutional generators. We also investigate several interesting properties unique to the new architecture.

1. Introduction

State-of-the-art in unconditional image generation is achieved using large-scale convolutional generators trained against adversarial discriminators [10, 11, 1]. While lots of nuances and ideas have contributed to the state-of-the-art recently, for many years since the introduction of DC-GAN [21] such generators are based around spatial convolutional layers, also occasionally using the spatial self-attention blocks [32]. Spatial convolutions are also invariably present in other popular generative architectures for images, including autoencoders [13], autoregressive generators [30], or flow models [3, 12]. Thus, it may seem that spatial convolutions (or at least spatial self-attention) is an unavoidable building block for state-of-the-art image generators.

Recently, a number of works have shown that individual images or collections of images of the same scene can be encoded/synthesized using rather different deep architectures (deep multi-layer perceptrons) of a special kind [19, 25]. Such architectures do not use spatial convolutions or spatial self-attention and yet are able to reproduce images rather well. They are, however, restricted to individual scenes. In this work, we investigate whether deep generators for unconditional image class synthesis can be built using similar architectural ideas, and, more importantly, whether the quality of such generators can be pushed to achieve state-of-the-art.

Perhaps surprisingly, we come up with a positive answer (Fig. 1), at least for the medium image resolution (of 256×256). We have thus designed and trained deep generative architectures for diverse classes of images that achieve similar quality of generation to state-of-the-art convolutional generator StyleGANv2 [11], even surpassing this quality for some datasets. Crucially, our generators are not using any form of spatial convolutions or spatial attention in their computational graphs. Instead, they use coordinate encodings of individual pixels, as well as multiplicative conditioning (weight modulation) on random latent vectors. Aside from such conditioning, the color of each pixel in our architecture is predicted independently (hence we call our image generator architecture *Conditionally-Independent Pixel Synthesis* (CIPS) generators).

In addition to suggesting this class of image generators and comparing its quality with state-of-the-art convolutional generators, we also investigate the extra flexibility that is permitted by the independent processing of pixels. This includes easy extention of synthesis to non-trivial topologies (e.g. cylindrical panoramas), for which the extension of spatial convolutions is known to be non-trivial [16, 2]. Furthermore, the fact that pixels are synthesized independently within our generators, allows sequential synthesis for memory-constrained computing architectures. It enables our model to both improve the quality of photos and generate more pixel values in a specific areas of image (i.e. to perform foveated synthesis).

2. Related Work

Feeding pixel coordinates as an additional input to the neural network previously was successfully used in the widely known CoordConv technique [17] to introduce the spatial position-related biases. Recently, the same idea was employed by the COCO-GAN [16] to generate images part-by-part or to create "looped" images like spherical panoramas. However, those models still use standard convolutions as the main synthesis operation. The synthesis process for neighboring pixels in such architectures is therefore not independent.

To the best of our knowledge, the problem of regressing a given image from pixel coordinates with a perceptron (that calculates each pixel's value independently) started from creating compositional patterns with an evolutionary approach [28]. Those patterns, appealing for digital artists, were also treated as kind of differentiable image parametrization [20]. However, this approach was not capable of producing photorealistic high resolution outputs (e.g. see the demo [9]).

Some machine learning blogs reported experiments with adversarial learning, where the generator takes a form of a perceptron accepting a random vector and pixel coordinates as an input, and returning that pixel's value as an output [5, 6]. The described model was successfully trained on MNIST, but has not been scaled to more complex image data.

Scene-representation networks [26] and later neural radiance fields (NeRF) networks [19] have demonstrated how 3D content of individual scenes can be encoded with surprising accuracy using deep perceptron networks. Following this realization, systems [25] and [29] considered the usage of periodic activation functions and so-called Fourier features to encode the pixel (or voxel) coordinates. In particular, the ability to fit high-resolution individual images with perceptron architetures using these features and activation functions was demonstrated. All these works however have not considered the task of learning image generators, which we address here.

The very recent (and independent) Generative Radiance Fields (GRAF) system [24] showed promising results at embedding the NeRF generator into an image generator for 3D aware image synthesis. Results for such 3D aware synthesis (still limited in diversity and resolution) for certain have been demonstrated. Here, we do not consider 3D-aware synthesis and instead investigate whether perceptron-based architectures can achieve high 2D image synthesis quality.

The independent and parallel work on implicit neural representations (INR) [27] achieves results similar to ours. Their work differs in the use of the nearest neighbor upsampling inside the generator (which "entangles" nearby positions in the high-resolution output) and in the reliance on another functional form of conditioning on the style vector.

3. Method

Our generator network synthesizes images of a fixed resolution $H \times W$ and has the multi-layer perceptron-type architecture G (see Fig. 2). In more detail, the synthesis of each pixel takes a random vector $\mathbf{z} \in \mathcal{Z}$ shared across all pixels, as well the pixel coordinates $(x, y) \in \{0 \dots W-1\} \times \{0 \dots H-1\}$ as input. It then returns the RGB value $\mathbf{c} \in [0,1]^3$ of that pixel $G : (x, y, \mathbf{z}) \mapsto \mathbf{c}$. Therefore, to compute the whole output image I, the generator G is evaluated at each position (x, y) of the coordinate grid, while keeping the random part $\mathbf{z}$ fixed:

$$I = \{G(x, y; \mathbf{z}) \mid (x, y) \in \texttt{mgrid}(H, W)\}, \tag{1}$$

where

$$\texttt{mgrid}(H, W) = \{(x, y) \mid 0 \le x < W, 0 \le y < H\}$$

is a set of integer pixel coordinates.

Following [10], the mapping network M (also a perceptron) turns $\mathbf{z}$ into a *style* vector $\mathbf{w} \in \mathcal{W}$, $M : \mathbf{z} \mapsto \mathbf{w}$. We then follow the StyleGANv2 [11] approach of injecting the style $\mathbf{w}$ into the process of generation via weight modulation. To make the paper self-contained, we describe the procedure in brief here.

Any modulated fully-connected (ModFC) layer of our generator (see Fig. 2) can be written in the form $\psi = \hat{B}\phi + \mathbf{b}$, where $\phi \in \mathbb{R}^n$ is an input, $\hat{B}$ is a learnable weight matrix $B \in \mathbb{R}^{m \times n}$ modulated with the style $\mathbf{w}$, $\mathbf{b} \in \mathbb{R}^m$ is a learnable bias, and $\psi \in \mathbb{R}^m$ is an output. The modulation takes place as follows: at first, the style vector $\mathbf{w}$ is mapped with an affine transform (referred to as A in Fig. 2) to a scale vector $\mathbf{s} \in \mathbb{R}^n$. Then, the (i, j)-th entry of $\hat{B}$ is computed as

$$\hat{B}_{ij} = \frac{s_j B_{ij}}{\sqrt{\epsilon + \sum_{k=1}^{n} (s_k B_{ik})^2}}, \tag{2}$$

where ϵ is a small constant.

To apply the spatial noise introduced by [10], we sample a separate random value n for each of $H \times W$ spatial positions from the standard normal distribution and add it (using the tensor broadcasting) to the output of ModFC layer $\psi + \alpha n$, where α is a learnable scalar parameter shared across both channels and spatial positions [11]. Afterwards, a LeakyReLU function [18] is applied to that sum. Interestingly, in our experiments α learns a value close to zero within all the layers, and therefore the spatial noise has almost no influence on the generation process.

Finally, in our default configuration we add skip connections for every two layers from the intermediate feature maps to RGB values and sum the contributions of RGB outputs corresponding to different layers. These skip connections naturally add values corresponding to the same pixel, and do not introduce interactions between pixels.

We note that the independence of the pixel generation process, makes our model parallelizable at inference time and, additionally, provides flexibility in the latent space. E.g., as we show in Supplementary material, for some modified variants of synthesis, each pixel can be computed with a different style vector $\mathbf{w}$, though gradual variation in $\mathbf{w}$ is required to achieve consistently looking images.

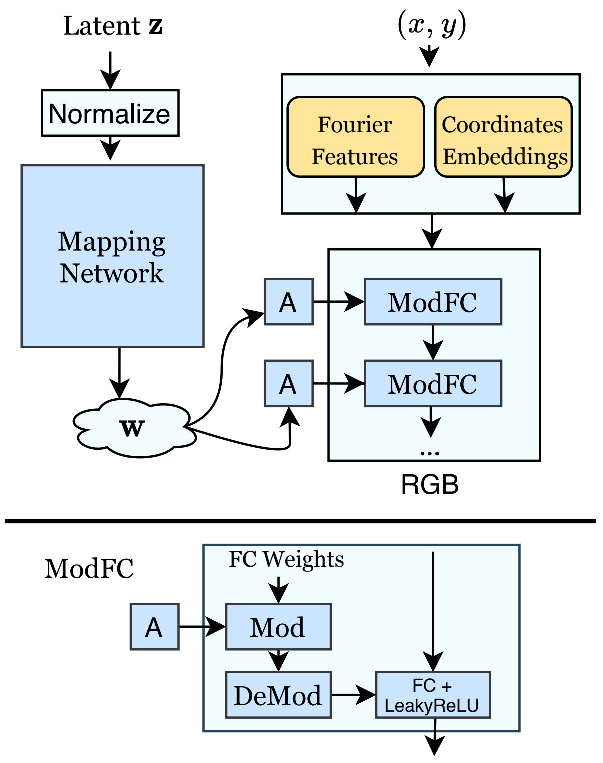


Figure 2: The Conditionally-Independent Pixel Synthesis (CIPS-base) generator architecture. Top: the generation pipeline, in which the coordinates (x, y) of each pixel are encoded (yellow) and processed by a fully-connected (FC) network with weights, modulated with a latent vector $\mathbf{w}$, shared for all pixels. The network returns the RGB value of that pixel. Bottom: The architecture of the modulated fully-connected layer (ModFC). Note that our *default* configuration also includes skip connections to the output and per-layer noise maps (not shown here).

3.1. Positional encoding

The architecture described above needs an important modification in order to achieve the state-of-the-art synthesis quality. Recently two slightly different versions of positional encoding for coordinate-based multi-layer perceptrons (MLP), producing images, were described in literature. Firstly, SIREN [25] proposed perceptrons with a principled weight initialization and sine as an activation function, used throughout all the layers. Secondly, [29] advocated the use of Fourier features as inputs to perceptrons. There, sine waves were used as activation functions in the very first layer only, which encoded non-learned affine transformation (in the spirit of [22]). In our experiments, we apply a somewhat in-between scheme: the sine function is used in the first layer that has *learnable* affine transformation resulting in Fourier embedding e_{fo}, while other layers use a standard LeakyReLU function:

$$e_{\text{fo}}(x,y) = \sin\left[B_{\text{fo}}(x',y')^T\right], \qquad (3)$$

where $x' = \frac{2x}{W-1} - 1$ and $y' = \frac{2y}{H-1} - 1$ are pixel coordinates, uniformly mapped to the range $[-1,1]$ and the weight matrix $B_{\text{fo}} \in \mathbb{R}^{2\times n}$ is learnable.

However, using only Fourier positional encoding turned out to be insufficient to produce plausible images. In particular, we have found out that the outputs of the synthesis tend to have multiple wave-like artifacts. Therefore, we also train a separate vector $e_{\text{co}}^{(x,y)}$ for each spatial position and call them *coordinate embeddings*. They represent $H \times W$ learnable vectors in total. For comparison of these two embedding from the spectral point of view, see Sec. 4.5. The full positional encoding $e(x,y)$ is a concatenation of Fourier features and the coordinate embedding:

$$e(x,y) = \text{concat}\left[e_{\text{fo}}(x,y), e_{\text{co}}^{(x,y)}\right]. \qquad (4)$$

It serves as an input for the next perceptron layer: $G(x,y;\mathbf{z}) = G'(e(x,y); M(\mathbf{z}))$.

4. Experiments

4.1. Architecture details

In our experiments, both Fourier features and coordinate embeddings had the dimension of 512. For ablation study we use the generator with 14 modulated fully-connected layers with 512 hidden units. For the final experiments, in order to reduce memory usage, we use 512 hidden units for the first ten layers, 256 units for layers 11 and 12, and 128 units for layers 13-14. We implement our experiments[1] on top of the public code[2] for StyleGANv2. We trained our model on eight P40 GPUs and showed approximately 22.4 mln images to the discriminator during training. With a straightforward PyTorch implementation, CIPS is 4.3 times slower than StyleGANv2 and has 3.3x more MACs (at 256×256 resolution). Further experimental details are provided in the supplementary material.

4.2. Evaluation

We now evaluate CIPS generators and their variations on a range of datasets. For the sake of efficiency, most evaluations are restricted to 256×256 resolution. The following datasets were considered:

- The *Flickr Faces-HQ* (FFHQ) [10] dataset contains 70,000 high quality well-aligned, mostly near frontal human faces. This dataset is the most regular in terms of geometric alignement and the StyleGAN variants are known to perform very well in this setting.

Dataset	StyleGANv2	CIPS (ours)
FFHQ	**3.83**	4.38
LSUN Churches	3.86	**2.92**
Landscapes	**2.36**	3.61
Satellite-Buildings	73.67	**69.67**
Satellite-Landscapes	51.54	**48.47**

Table 1: FID on multiple datasets at resolution of 256^2 for CIPS-skips model. Note that CIPS is of comparable quality with state-of-the-art StyleGANv2, and even exceeds it for some datasets. The value for CIPS model on FFHQ differs from the one reported in Tab. 3 as we trained this model for more time and with larger batch size. For Landscape dataset we employ generator with residual blocks rather than our default setting as it produced slightly better FID in this case.

Model	Precision↑	Recall↑	PPL↓
StyleGANv2	0.609	**0.513**	**270.2**
CIPS (ours)	**0.613**	0.493	336.4

Table 2: Precision & Recall and PPL measured on FFHQ at 256^2. The resulting quality of our model is better in terms of precision (corresponds to plausibility of images) and worse in recall (this points to the greater number of dropped modes). Perceptual path length (measures both disentanglement and quality of samples) is better for StyleGANv2 eventhough it was trained without the corresponding regularization in this case (as was our method).

- The *LSUN Churches* [31] contains 126,000 outdoor photographs of churches of diverse architectural style. The dataset is less regular/aligned, yet all images share upright orientation.
- The *Landscapes* dataset contains 60,000 manually collected landscape photos from the Flickr website.
- The *Satellite-Buildings*[3] dataset contains 280,741 images of 300×300 pixels (which we crop to 256×256 resolution and randomly rotate). This dataset has large size, and is approximately aligned in terms of scale, yet lacks consistent orientation.
- Finally, the *Satellite-Landscapes*[4] contains a smaller curated collection of 2,608 images of 512×512 resolution of satellite images depicting various unusual landscapes found on Google Earth (which we crop to 256×256 resolution). This is the most “textural” dataset, that lacks consistent scale or orientation.

For evaluation, we relied on commonly used metrics for image generation: Frechet Inception Distance (FID) [7]

[1]https://github.com/saic-mdal/CIPS
[2]https://github.com/rosinality/stylegan2-pytorch
[3]https://www.crowdai.org/challenges/mapping-challenge
[4]https://earthview.withgoogle.com/

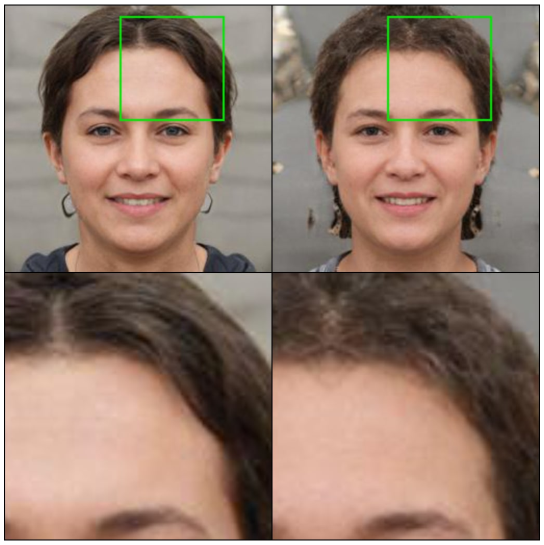


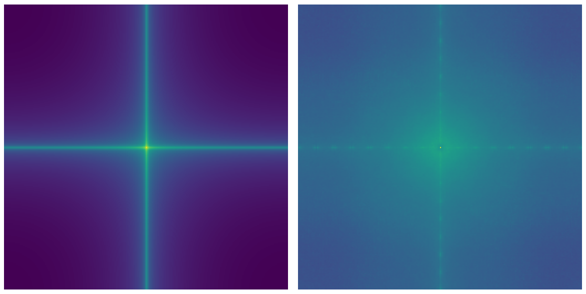
Figure 3: Image corresponding to the mean style vector in the space of $\mathcal{W}$ for CIPS (left) and CIPS-NE (no embeddings) generators (right). Left image has more plausible details like hair which confirms the results in Tab. 3.

as well as more recently introduced generative Precision and Recall measures [23, 14] and Perceptual path length (PPL) [10].

Our main evaluation is thus against the state-of-the-art StyleGANv2 generator [11]. We took FID value for LSUN-Churches directly from the original paper [11] and trained StyleGANv2 on other datasets. We trained StyleGANv2 in authors' setup but without style-mixing and path regularization of generator – as noted in the original paper, these changes do not influence the FID metric. The results of this key comparison are presented in Tab. 1 and 2. Neither of the two variants of the generator dominates the other, with StyleGANv2 achieving lower (better) FID score on FFHQ, Landscapes, while CIPS generator achieving lower score on LSUN Churches and Satellite datasets.

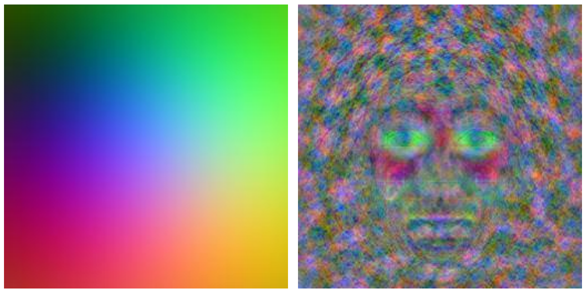
4.3. Ablations

We then evaluate the importance of different parts of our model by its ablation on the FFHQ dataset (Tab. 3). We thus consider removing Fourier features, coordinate embeddings (config referred to as *CIPS-NE*) and replace LeakyReLU activation with sine function in all layers. We also compare the variants with residual connections (we follow StyleGANv2 [11] implementation adjusting variance of residual blocks with the division by $\sqrt{2}$) with our main choice of cumulative projections to RGB. Additionally, the "base" configuration without skip connections and residual connections is considered. Finally, to estimate the importance of conditional independence of CIPS approach we replace each ModFC layer with a combination of a pointwise convolution and a depthwise convolution (we use filters 3×3)



(a) Fourier features (b) Coordinate embeddings

Figure 4: The spectrum magnitude for our two kinds of positional encoding (color scale is equal for both plots). The output of coordinate embeddings clearly has more higher frequencies.



(a) Fourier features (b) Coordinate embeddings

Figure 5: PCA plot (three components) for two kinds of positional encoding of CIPS-base. Interestingly, learned coordinate embeddings contain both high-frequency patterns and eye shaped features (emerging from the registration of the FFHQ dataset).

from MobileNet [8]. In this comparison, all models were trained for 300K iterations with batch size 16.

As the results show, coordinate embeddings, residual blocks and cumulative projection to RGB significantly improve the quality of the model. Interestingly, adding 3×3 convolutions ("MobileNet") degrades performance, we suggest it may have such an effect as it makes the model less rotation-invariant. The removal of coordinate embeddings most severely worsens the FID value, and affects the quality of generated images (Fig. 3). We further investigate the importance of coordinate embeddings for the CIPS model below.

4.4. Influence of positional encodings

To analyze the difference between Fourier features e_{fo} and coordinate embeddings e_{co}, we plotted the spectrum of these codes for the generator CIPS-base, trained on FFHQ. As shown in Fig. 4, Fourier encoding generally carries low-frequency components, whereas coordinate embeddings resemble more high-frequency details. The Principal Component Analysis (PCA) of the two encodings supports the

CIPS	"base"	"no embed" (NE)	"no Fourier"	"**default**"	"residual"	"sine"	"MobileNet"
Fourier features	+	+	−	+	+	−	+
Coordinate embeddings	+	−	+	+	+	+	+
Residual blocks	−	−	−	−	+	−	−
Skip connections	−	−	−	+	−	−	−
Sine activation	−	−	−	−	−	+	−
Conv3×3	−	−	−	−	−	−	+
FID	6.71	12.71	10.18	**6.31**	6.52	10.0	11.39

Table 3: Effects of the modifications of CIPS generator on the FFHQ dataset in terms of Frechet Inception Distance (FID) score. Each column corresponds to a certain configuration, while rows correspond to present/missing features. The simultaneous usage of Fourier features and coordinate embeddings is necessary for a good FID score. Also, both residual connections and cumulative skip connections (default configuration) to the output outperform the plain multilayer perceptron.

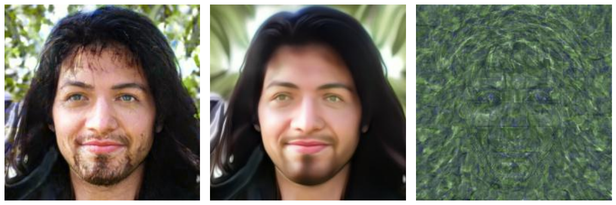


Figure 6: Influence of different types of positional encoding on the resulting image. Left: original image. Center: coordinate embeddings zeroed out (the image contains no fine-grained details). Right: Fourier features zeroed out (only high-frequency details are present).

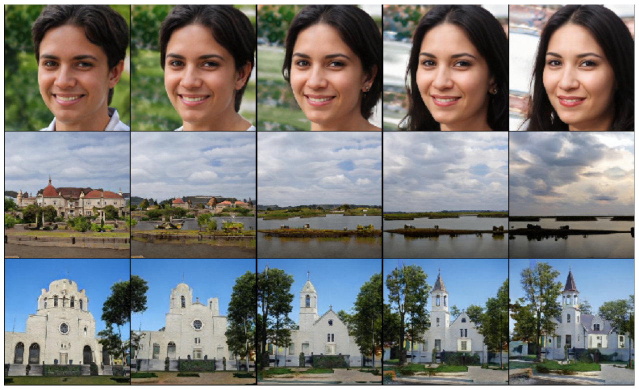


Figure 7: Latent linear morphing between two sampled images – the left-most and right-most ones.

same conclusion (Fig. 5) The possible explanation is simple: coordinate embeddings are trained independently for each pixel, while $e_{\text{fo}}(x, y)$ is a learned function of the coordinates. However, the next layers of the network could transform the positional codes and, for example, finally produce more fine-grained details, relying on Fourier features. To demonstrate that this is not the case, we conducted the following experiment. We have zeroed out either the output of Fourier features or coordinate embeddings and showed the obtained images in Fig. 6. We notice that the information about the facial hair's details as well as the forelock is contained in the coordinate embeddings. Thus, coordinate

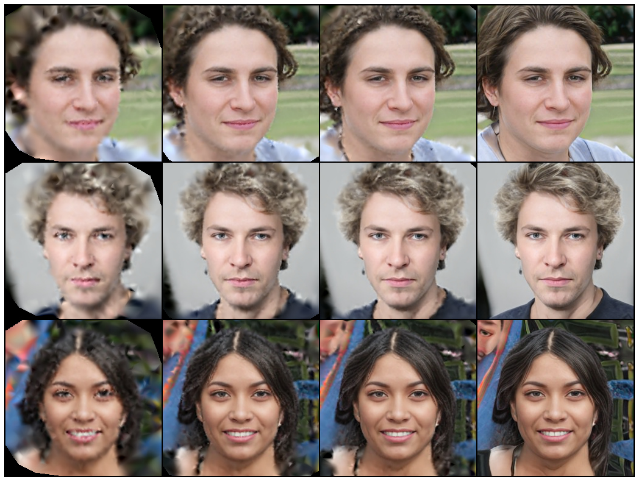


Figure 8: Images generated using foveated synthesis. In each case, the CIPS generator was sampled on a 2D Gaussian distribution concentrated in the center of an image (standard deviation = $0.4*$image size). Left to right: sampled pattern covers 5% of all pixels, 25%, 50%, 100% (full coordinate grid). Missing color values have been filled via bicubic interpolation.

embeddings turn out to be key for high-frequency details in the resulting image.

4.5. Spectral analysis of generated images

Recently, [4] observed that the common convolutional upsampling operations can lead to the inability to learn the spectral distribution of real images. In contrast, CIPS operates explicitly with the coordinate grid and has no upsampling modules, which should lead to improved reproduction of the spectrum. Indeed, we compare the spectrum of our models (CIPS-base without residual and skip connections, and CIPS-NE) to StyleGANv2 and the comparison reveals the advantage of CIPS generators in the spectral domain.

The analysis of magnitude spectra for produced images is given in Fig. 9 (average results for $5,000$ randomly sampled images). The spectrum of StyleGANv2 has artifacts

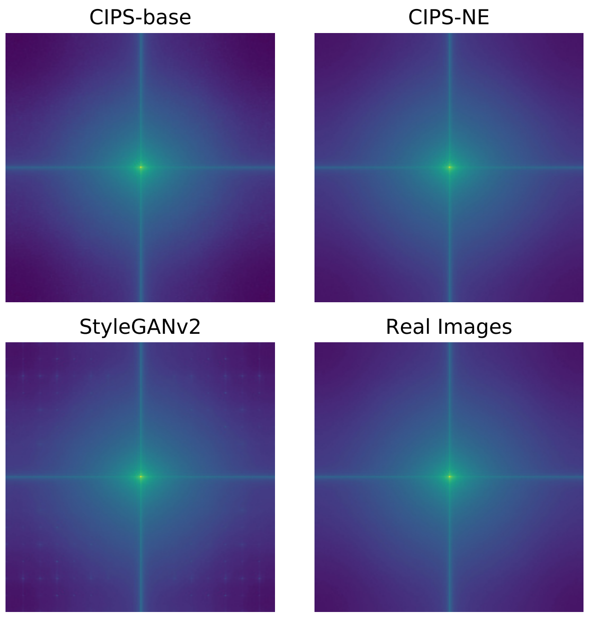


Figure 9: Magnitude spectrum. Our models produce less artifacts in high frequency components (note the grid pattern in StyleGANv2 spectrum). Two CIPS models are difficult to distinguish between (zoom-in required).

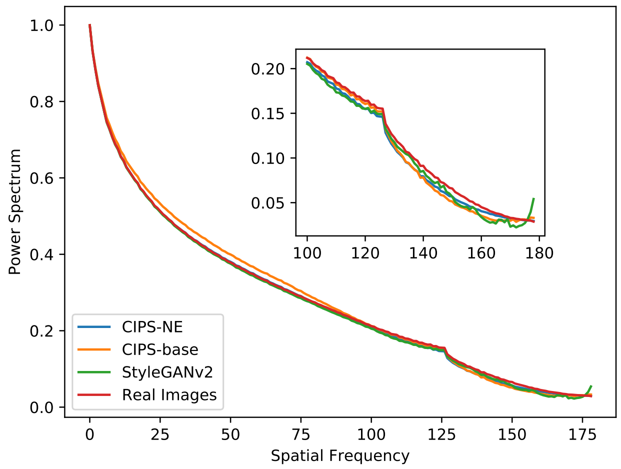


Figure 10: Azimuthal integration over Fourier power spectrum. The curve of StyleGANv2 has heavy distortions in most high frequency components. Surprisingly, CIPS-NE has more realistic and smoother tail than CIPS-base, while being worse in terms of FID.

in high-frequency regions, not present in both considered variants of CIPS generators. Following prior works [4], we also use the azimuthal integration (AI) over the Fourier power spectrum (Fig. 10). It is worth noting that AI statistics of CIPS-NE are very close to the ones of real images. However, adding the coordinate embeddings degrades a realistic spectrum while improving the quality in terms of FID (Tab. 3).

We note that the introduction of skip connections in fact makes the spectra less similar to those of natural images.

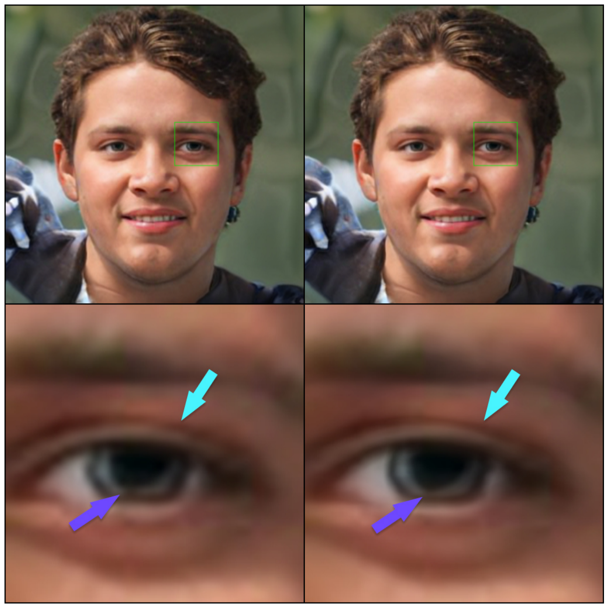


Figure 11: Left: the generated image of resolution 256×256, upscaled with Lanczos upsampling scheme [15] to 1024×1024. Right: the image, synthesized by CIPS, trained at resolution of 256×256 on the coordinate grid of resolution 1024×1024. Note the sharpness/plausibility of the eyelid and the more proper shape of the pupil.

4.6. Latent interpolation

We conclude the experimental part with the demonstration of the flexibility of CIPS. As well as many other generators, CIPS generators have the ability to interpolate between latent vectors with meaningful morphing (Fig. 7). As expected, the change between the two images occurs smoothly as in StyleGANv2.

4.7. Foveated rendering and upsampling

One of the inspiring applications of our per-pixel generator is the foveated synthesis. The foveated synthesis ability can be beneficial for computer graphics and other applications, and it also mimics human visual system. In foveated synthesis, sampling is irregular and more dense in the area, where the gaze is assumed to be directed to, and more sparse outside of that region. CIPS is evaluated on this irregular set of pixels (that contains less pixels that the full image). The color for missing pixels is filled using interpolation. The demonstration of this method is provided in Fig. 8.

Alongside the foveated rendering, we are also able to interpolate the image beyond the training resolution by simply sampling denser grids. Here we use a model, trained on images of 256×256 resolution to process a grid of 1024×1024

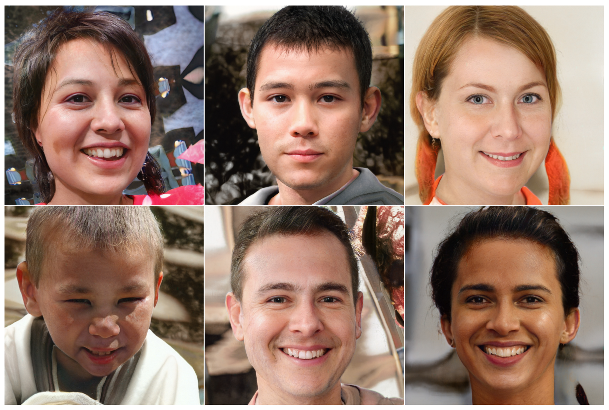


Figure 12: Samples from CIPS trained progressively on FFHQ-1024.

Generator	Landscapes-512	FFHQ-512	FFHQ-1024
StyleGANv2	**3.94**	**3.41**	**2.84**
CIPS (ours)	6.90	6.18	10.07

Table 4: FID value for for CIPS ans StyleGANv2 models trained on hi-res FFHQ and Landscapes datasets. Quality and diversity of images produced by CIPS in high resolution is worse than in case of StyleGANv2.

pixels (the learned tensor of coordinate embeddings is bilinearly upscaled). We compare the result of such synthesis with the results of upsampling the image, synthesized at the 256×256 resolution, using the Lanczos filter [15]. As Fig. 11 suggests, more plausible details are obtained with denser synthesis than with Lanczos upsampling.

4.8. High-resolution image generation

Finally, to show the ability of our architecture to generalized to hi-res images we trained CIPS on FFHQ with resolution 512×512 and 1024×1024 and Landscapes with resolution 512×512. These models were trained progressively: we initialized our default generator with weights trained on 256×256 resolution images. In this setting we trained the networks with batch size of 8 showing about 1.9 mln images to the discriminator. At resolution of 1024×1024, training in full size did not fit the GPU memory (on P40 GPU), so we used the patch-based approach with patches of size 512×512 which explains the relatively bad FID score (see Supplementary for details). In Tab. 4 we report FID metrics showing that hi-res datasets are feasible for our model. At higher resolution, CIPS produces worse result than StyleGANv2. Examples of outputs are shown in Fig. 12.

4.9. Typical artifacts

Finally, we show the typical artifacts that keep recurring in the results of CIPS generators (Fig. 13). We attribute the wavy texture (in hair) and repeated lines pattern (in buildings) to the periodic nature of sine activation function within the Fourier features. Also we note that sometimes CIPS produces a realistic image with a small part of the image being inconsistent with the rest and out of the domain. Our belief is that this behaviour is caused by the LeakyReLU activation function that divides the coordinate grid into parts. For each part, CIPS effectively applies its own inverse discrete Fourier transform. As CIPS generators do not use any upsampling or other pixel coordination, it is harder for the generator to safeguard against such behaviour.

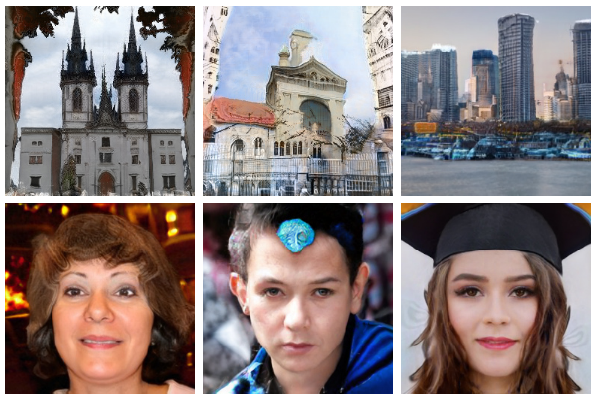


Figure 13: Examples of the most common kinds of artifacts on different datasets. They are best described as wavy textures on hair, background, and glowing blobs. See text for discussion.

5. Conclusion

We have presented a new generator model called CIPS, a high-quality architecture with conditionally independent pixel synthesis, such that the color value is computed using only random noise and coordinate position.

Our key insight is that the proposed architecture without spatial convolutions, attention or upsampling operations is nevertheless competitive and obtains reasonable quality in terms of FID and precision & recall. Furthermore, in the spectral domain outputs of CIPS match real images more closely than StyleGANv2 generators. Interestingly, CIPS-NE modification is weaker in terms of plausibility, yet has a more realistic spectrum.

Direct usage of a coordinate grid allows us to work with more complex structures, such as cylindrical panoramas, just by replacing the underlying coordinate system (see Supplementary).

In summary, our generator demonstrates quality on par with state-of-the-art model StyleGANv2; moreover, it has applications in various diverse scenarios. We have shown that the considered model could be successfully applied to foveated rendering and super-resolution problems in their generative interpretations. Future development of our approach assumes researching these problems in their image-to-image formulations.

References

[1] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 1

[2] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling. Spherical cnns. In *International Conference on Learning Representations*, 2018. 2

[3] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016. 1

[4] R. Durall, M. Keuper, and J. Keuper. Watch Your Up-Convolution: CNN Based Generative Deep Neural Networks Are Failing to Reproduce Spectral Distributions. In *Proc. CVPR*, pages 7887–7896, 2020. 6, 7

[5] D. Ha. Generating large images from latent vectors. *blog.otoro.net*, 2016. 2

[6] D. Ha. Generating large images from latent vectors - part two. *blog.otoro.net*, 2016. 2

[7] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proc. NIPS*, NIPS'17, page 6629–6640, Red Hook, NY, USA, 2017. Curran Associates Inc. 4

[8] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 5

[9] A. Karpathy. Convnetjs demo: Image "painting". https://cs.stanford.edu/people/karpathy/convnetjs/demo/image_regression.html. Accessed: 2020-11-05. 2

[10] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, pages 4396–4405, 2019. 1, 3, 4, 5

[11] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. In *Proc. CVPR*, pages 8107–8116, 2020. 1, 2, 3, 5

[12] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Proc. NeurIPS*, pages 10215–10224, 2018. 1

[13] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[14] T. Kynkäänniemi, T. Karras, S. Laine, J. Lehtinen, and T. Aila. Improved precision and recall metric for assessing generative models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Proc. NeurIPS*, volume 32, pages 3927–3936. Curran Associates, Inc., 2019. 5

[15] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of research of the National Bureau of Standards*, 45:255–282, 1950. 7, 8

[16] C. H. Lin, C. Chang, Y. Chen, D. Juan, W. Wei, and H. Chen. Coco-gan: Generation by parts via conditional coordinating. In *Proc. ICCV*, pages 4511–4520, 2019. 2

[17] R. Liu, J. Lehman, P. Molino, F. Petroski Such, E. Frank, A. Sergeev, and J. Yosinski. An intriguing failing of convolutional neural networks and the CoordConv solution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Proc. NeurIPS*, pages 9627–9638. Curran Associates, Inc., 2018. 2

[18] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, volume 30, page 3. Citeseer, 2013. 3

[19] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Proc. ECCV*, pages 405–421, Cham, 2020. Springer International Publishing. 2

[20] A. Mordvintsev, N. Pezzotti, L. Schubert, and C. Olah. Differentiable image parameterizations. *Distill*, 2018. https://distill.pub/2018/differentiable-parameterizations. 2

[21] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016. 1

[22] A. Rahimi, B. Recht, et al. Random features for large-scale kernel machines. In *Proc. NIPS*, volume 3, page 5. Citeseer, 2007. 3

[23] M. S. M. Sajjadi, O. Bachem, M. Lucic, O. Bousquet, and S. Gelly. Assessing generative models via precision and recall. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Proc. NIPS*, volume 31, pages 5228–5237. Curran Associates, Inc., 2018. 5

[24] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger. GRAF: Generative Radiance Fields for 3D-Aware Image Synthesis. In *Proc. NeurIPS*. Curran Associates, Inc., 2020. 2

[25] V. Sitzmann, J. N. P. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein. Implicit Neural Representations with Periodic Activation Functions. In *Proc. NeurIPS*. Curran Associates, Inc., 2020. 2, 3

[26] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Proc. NeurIPS*. 2019. 2

[27] I. Skorokhodov, S. Ignatyev, and M. Elhoseiny. Adversarial generation of continuous images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[28] K. O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines*, 8(2):131–162, Jun 2007. 2

[29] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. In *Proc. NeurIPS*. Curran Associates, Inc., 2020. 2, 3

[30] A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *Proc. ICML*, pages 1747–1756, 2016. 1

[31] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. 2016. 4

[32] H. Zhang, I. J. Goodfellow, D. N. Metaxas, and A. Odena. Self-attention generative adversarial networks. In *Proc. ICML*, 2019. 1