Generative Model
StyleGAN

Suk-Hwan Lee

Artificial Intelligence

Creating the Future

Dong-A University

Division of Computer Engineering &
Artificial Intelligence

References

Main

- https://theaisummer.com/cnn-architectures/

blog Sub

- https://deepbaksuvision.github.io/Modu_ObjectDetection/

PyTorch tutorial

- website : https://pytorch.org/
- Korea website : https://pytorch.kr/

github

- https://github.com/pytorch
- https://github.com/9bow/PyTorch-tutorials-kr
- torchvision : https://github.com/pytorch/vision
- https://github.com/weiaicunzai/pytorch-cifar100

StyleGAN3 : Alias-Free Generative Adversarial Networks

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, Timo Aila,
NeurIPS 2021
Aalto University and NVIDIA

https://github.com/NVlabs/stylegan3
https://nvlabs.github.io/stylegan3/
PR12-338, Jaejun Yoo 발표 (유투브)
Paper Review, https://slowstarter.tistory.com/12

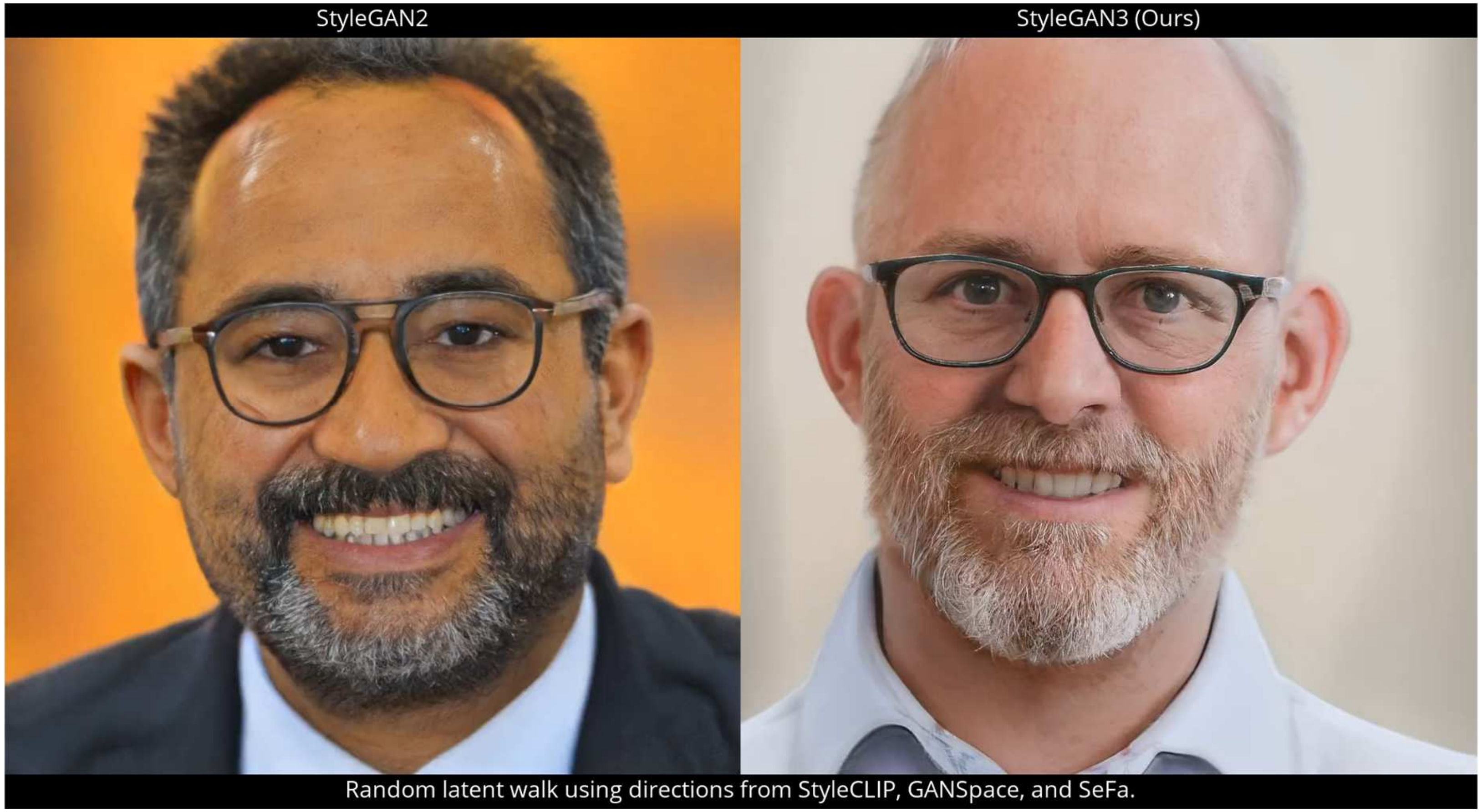
Abstract

- We observe that despite their hierarchical convolutional nature, the synthesis process of typical generative adversarial networks depends on **absolute pixel coordinates** in an unhealthy manner. This manifests itself as, e.g., detail appearing to be glued to image coordinates instead of the surfaces of depicted objects.
- We trace the root cause to careless signal processing that **causes aliasing in the generator network**. Interpreting all signals in the network as continuous, we derive generally applicable, small architectural changes that guarantee that unwanted information cannot leak into the hierarchical synthesis process.
- The resulting networks match the FID of StyleGAN2 but differ dramatically in their internal representations, and they are fully equivariant to translation and rotation even at subpixel scales. Our results pave the way for generative models better suited for video and animation.

- 기존의 StyleGAN2은 texture sticking 이라는 문제를 가지고 있음. 이미지는 구조적으로 학습되어야 하는데(ex. 턱에 해당하는 위치에 수염이 있어야 함), StyleGAN 의 Generator는 이미지의 각 특징들을 hierarchical 방식으로 학습하지 않고 고정된 픽셀 단위로 학습함
- Interpolation 영상을 보면, StyleGAN2의 영상에서는 턱 수염이 인물을 따라가지 않고, 픽셀 단위로 고정되어 있는 것을 확인할 수 있음. 위의 문제를 해결하기 위해서 hiearchical하게 이미지를 합성할 수 있도록 alias-free 한 network 제안함.
- 그 결과, translation이나 rotation에 대해 **equivariance**를 만족하였으며, video나 animation을 만들기에 적합하다고 볼 수 있음

StyleGAN3 : Alias-Free Generative Adversarial Networks

https://github.com/NVlabs/stylegan3
https://nvlabs.github.io/stylegan3/



StyleGAN3 : Alias-Free Generative Adversarial Networks

https://github.com/NVlabs/stylegan3
https://nvlabs.github.io/stylegan3/

StyleGAN3 : Alias-Free Generative Adversarial Networks

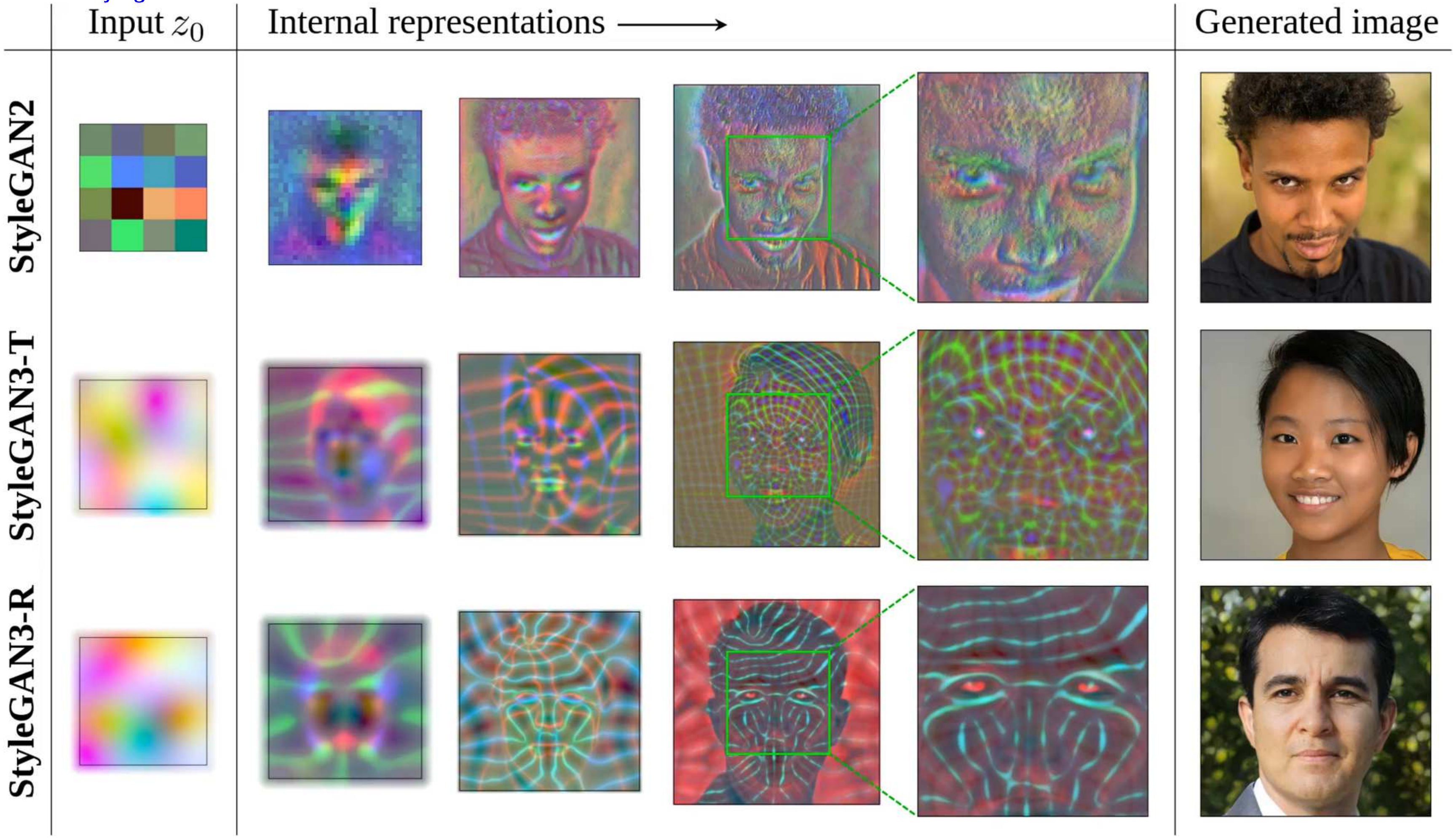
https://github.com/NVlabs/stylegan3
https://nvlabs.github.io/stylegan3/

StyleGAN3 : Alias-Free Generative Adversarial Networks

https://github.com/NVlabs/stylegan3
https://nvlabs.github.io/stylegan3/

StyleGAN3 : Alias-Free Generative Adversarial Networks

https://github.com/NVlabs/stylegan3
https://nvlabs.github.io/stylegan3/



1. Introduction

Texture Sticking

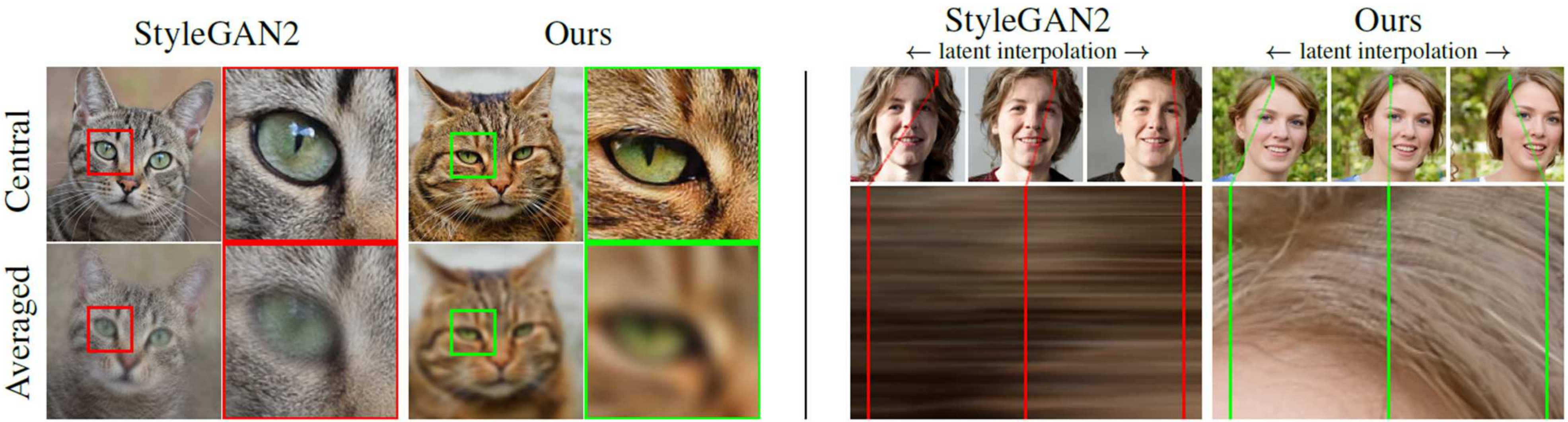


Figure 1: Examples of "texture sticking". **Left:** The average of images generated from a small neighborhood around a central latent (top row). The intended result is uniformly blurry because all details should move together. However, with StyleGAN2 many details (e.g., fur) stick to the same pixel coordinates, showing unwanted sharpness. **Right:** From a latent space interpolation (top row), we extract a short vertical segment of pixels from each generated image and stack them horizontally (bottom). The desired result is hairs moving in animation, creating a time-varying field. With StyleGAN2 the hairs mostly stick to the same coordinates, creating horizontal streaks instead.

StyleGAN3 : Alias-Free Generative Adversarial Networks

1. Introduction

Texture Sticking

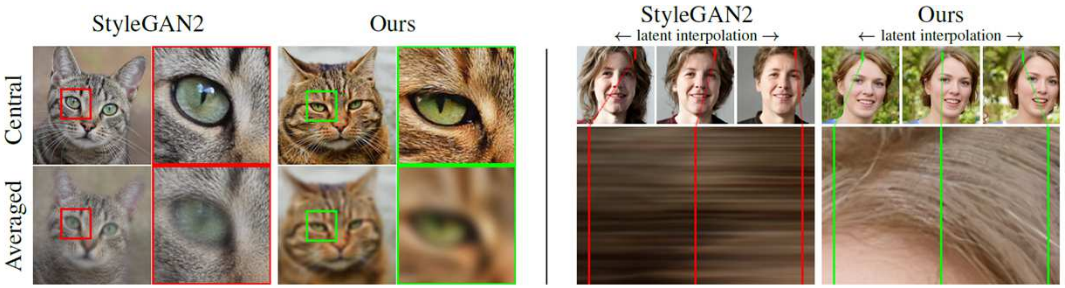


Figure 1: Examples of "texture sticking". **Left:** The average of images generated from a small neighborhood around a central latent (top row). The intended result is uniformly blurry because all details should move together. However, with StyleGAN2 many details (e.g., fur) stick to the same pixel coordinates, showing unwanted sharpness. **Right:** From a latent space interpolation (top row), we extract a short vertical segment of pixels from each generated image and stack them horizontally (bottom). The desired result is hairs moving in animation, creating a time-varying field. With StyleGAN2 the hairs mostly stick to the same coordinates, creating horizontal streaks instead.

- 기존 StyleGAN generator: coarse, low-resolution feature에서 시작하여 upsampling하고, convolution으로 local하게 mixing하고, non-linear function을 거쳐 detail들을 찾아가는 방식으로 학습해왔음
  - coarse feature들이 finer feature의 여부에 대해서는 조절을 하지만, 정확한 위치까지 control하지는 못함.
  - 결과적으로, fine detail이 hiearchical하게 학습되는 것이 아니라 pixel coordinate에 고착화된 상태로 학습이 됨.
  - 이러한 **texture sticking** 문제는 위 이미지를 보면 확인할 수 있음.
- 위 이미지 중 고양이 이미지를 보면, StyleGAN2은 눈을 제외한 영역이 블러처리 되지 않고, StyleGAN3에서는 모든 영역이 고르게 블러처리가 됨을 확인할 수 있음. interpolation 된 이미지를 보면, StyleGAN2와 다르게 3에서는 translation에 따라 자연스럽게 interpolation 됨을 확인 할 수 있음
  - latent interpolation을 통해 자연스러운 transformation을 만들었을 때, 이 transformation이 hierarchy하게 조절되는 것이 아니라 각 feature들이 특정 pixel에 고착화되어 있음을 확인 할 수 있음

1. Introduction

Texture Sticking

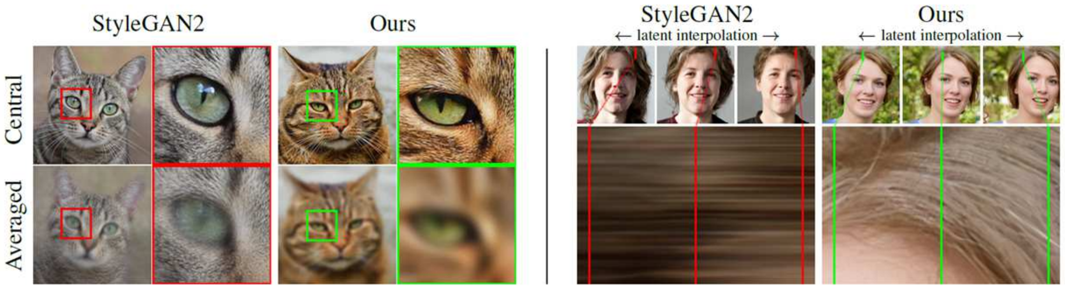


Figure 1: Examples of "texture sticking". **Left:** The average of images generated from a small neighborhood around a central latent (top row). The intended result is uniformly blurry because all details should move together. However, with StyleGAN2 many details (e.g., fur) stick to the same pixel coordinates, showing unwanted sharpness. **Right:** From a latent space interpolation (top row), we extract a short vertical segment of pixels from each generated image and stack them horizontally (bottom). The desired result is hairs moving in animation, creating a time-varying field. With StyleGAN2 the hairs mostly stick to the same coordinates, creating horizontal streaks instead.

Observations

- 어디선가 의도치 않게 **spatial information**이 주입되고 있다.
- **Translation Equivariance representation**이 필요한데 무엇인가가 이를 깨고 있다

"Our goal is an architecture that exhibits a more natural transformation hierarchy, where the exact sub-pixel position of each feature is exclusively inherited from the underlying coarse features."

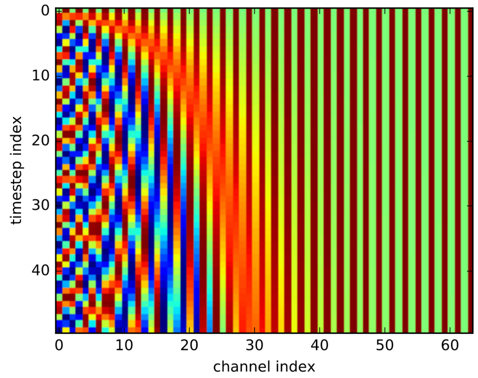
Our goal is to make every layer of G equivariant w.r.t. the continuous signal, so that all finer details transform together with the coarser features of a local neighborhood. If this succeeds, the entire network becomes similarly equivariant.

1. Introduction

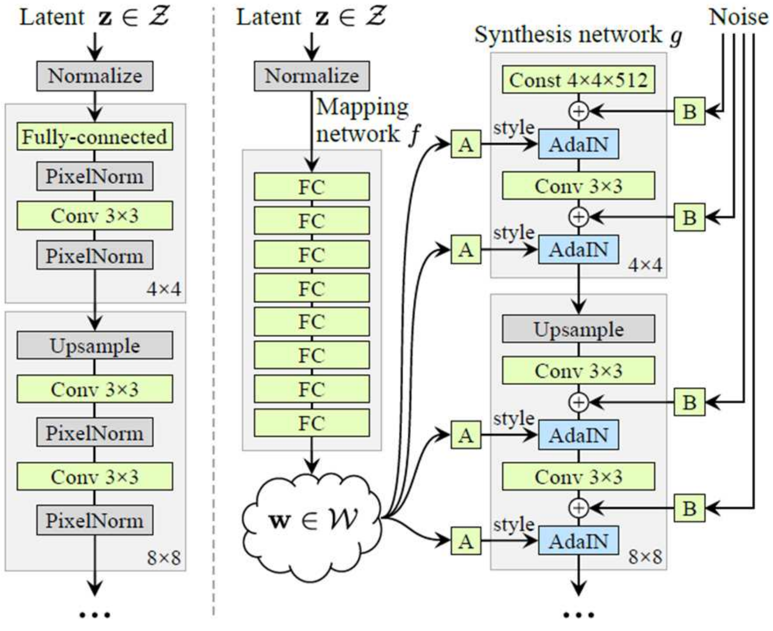
Motivation and Main Problem

- ➢ **Image borders**
- ➢ **Per-pixel noise inputs**
- ➢ **Positional encoding**
- ➢ **Aliasing**

Image borders (boundary)



Positional Encoding



(a) Traditional (b) Style-based generator

Per-pixel noise inputs

- 본 논문의 저자들은 이 중에서 aliasing이 가장 critical한 issue라고 주장함
- network는 aliasing이 조금만 존재해도 이를 증폭하는 경향이 있어서 학습이 진행되면서 scale이 커질 수록 픽셀에 특정 texture가 고착되기 때문임

1. Introduction

Motivation and Main Problem

- **Image borders**
- **Per-pixel noise inputs**
- **Positional encoding**
- **Aliasing**

(Prerequisites)

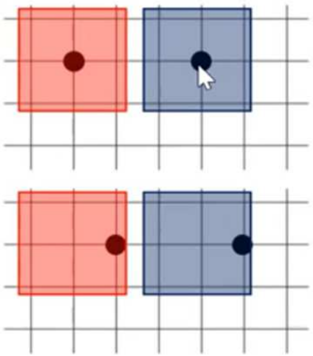
Simoncelli et al. 1992

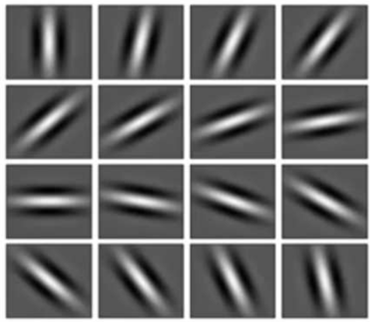
Zhang et al. 2019, Azulay et al. 2019

CNN이 Translation Equivariant Representation을 학습하려면 aliasing이라는 현상이 생기지 않도록 해줘야 함

CNN classifier w/ global avr. pooling

"To make CNN representations (before GAP) translation equivariant (so that the final representation becomes translation invariant), we need to make sure that any feature map that uses stride does not contain frequencies above the Nyquist frequency."





1. Introduction

Motivation and Main Problem

- **Image borders**
- **Per-pixel noise inputs**
- **Positional encoding**
- **Aliasing**

❖ Aliasing는 발생 이유

- Non-ideal upsampling filters (ex. nearest, bilinear, strided conv): Generator에서 upsampling을 하는 과정에서 low-path filtering을 하지 않기 때문임. 즉 ideal하지 않은 upsampling filter 때문에 원치 않은 high-frequency들이 계속 더해져서 aliasing이 일어남.
- Pointwise application nonlinearities such as ReLU: 예를 들어 음수일때 0으로 만들어주는 relu가 있으면 갑자기 값이 확 튀게 되기 때문에 aliasing이 일어남

- 또한, 저자들은 이러한 aliasing에서 비롯되는 문제가 styleGAN 뿐만 아니라 deep learning에서 전반적으로 발생한다고 보고함

❖ Aliasing는 해결 방안

- 이론적으로 aliasing은 Nyquist-Shannon sampling theorem로 해결할 수 있음. 저자들은 StyleGAN2의 Generator를 신호론적으로 분석하여 **upsampling filter**랑 **pointwise nonlinearties**에서 생기는 aliasing을 해결하고자 함

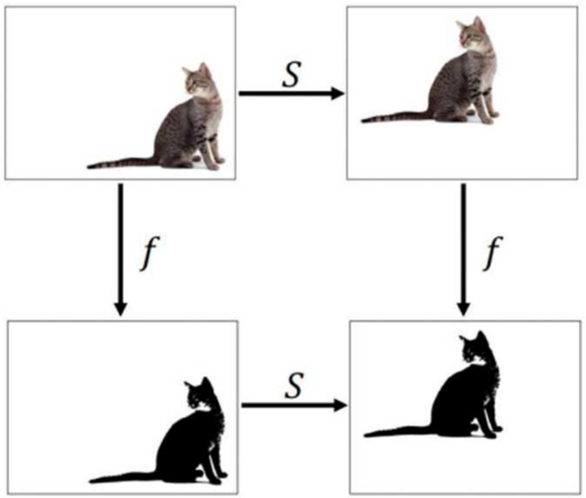
1. Introduction

Equivariance

- 본 논문은 이미지를 생성하기 위한 다양한 operation(ex, CNN, ReLU, Upsampling / Downsampling..)에서 Equivariance 해야한다고 주장함.
- Generator 모델을 Equivariance 하게 만드는 것이 논문의 핵심이라고 볼 수 있음

- 본 논문에서 지향하는 바는 각각의 특징들을 hierarchical 하게 학습하게 만드는 것임.
- 따라서 각각의 layer들을 translation equivariant 하게 만들어서 변형된 input으로 생성된 output이 일반 output을 변형한 것과 같도록 만든다면, 이미지의 각 특징들이 자연스럽게 hierarchical 하게 학습될 것이라는 아이디어를 말하고자 함.

Equivariance



예를 들어 어떤 이미지를 회전시키고 싶을 때,

- latent code z 로 부터 생성된 이미지를 회전시켰을 때랑
- 회전된 latent code z 에서 생성된 이미지가
- 같다면, 이것이 바로 rotation에 대해 equivariant 함

2. Equivariance via continuous signal interpretation

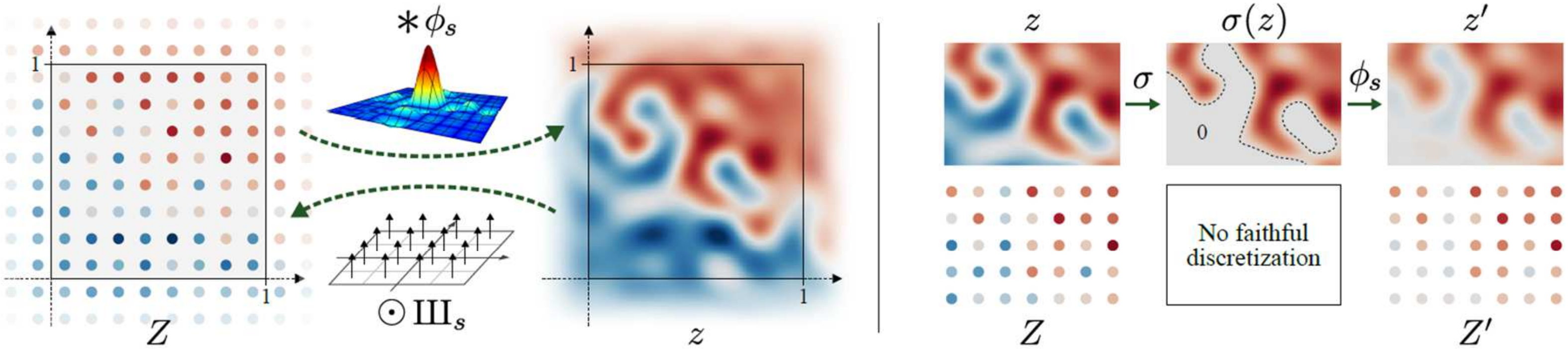


Figure 2: **Left:** Discrete representation Z and continuous representation z are related to each other via convolution with ideal interpolation filter ϕ_s and pointwise multiplication with Dirac comb Ш_s. **Right:** Nonlinearity σ, ReLU in this example, may produce arbitrarily high frequencies in the continuous-domain $\sigma(z)$. Low-pass filtering via ϕ_s is necessary to ensure that Z' captures the result.

Discrete and continuous representation of network layers Practical neural networks operate on the discretely sampled feature maps. Consider operation $\mathbf{F}$ (convolution, nonlinearity, etc.) operating on a discrete feature map: $Z' = \mathbf{F}(Z)$. The feature map has a corresponding continuous counterpart, so we also have a corresponding mapping in the continuous domain: $z' = \mathbf{f}(z)$. Now, an operation specified in one domain can be seen to perform a corresponding operation in the other domain:

$$\mathbf{f}(z) = \phi_{s'} * \mathbf{F}(\text{Ш}_s \odot z), \qquad \mathbf{F}(Z) = \text{Ш}_{s'} \odot \mathbf{f}(\phi_s * Z), \tag{1}$$

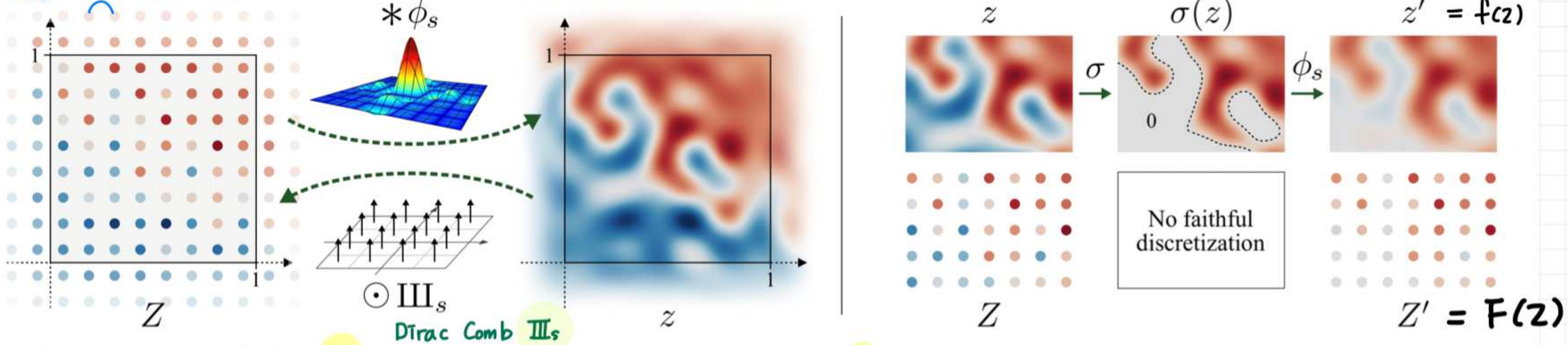
where $\odot$ denotes pointwise multiplication and s and s' are the input and output sampling rates. Note that in the latter case $\mathbf{f}$ must not introduce frequency content beyond the output bandlimit $s'/2$.

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \operatorname{sinc}\left(\frac{t - nT}{T}\right)$$

- ϕ_s : width · height 모두 $\frac{s}{2}$의 bandlimit
  → continuous signal 이 모두 주파수 표현가능

$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$ (normalized sinc func)

2차원 $\frac{1}{s}$ (s : sample rate) ideal interpolation filter ϕ_s : $\phi_s(x) = \operatorname{sinc}(sx_0) \cdot \operatorname{sinc}(sx_1)$



Dirac Comb III_s

discrete representation Z

continuous representation z

- feature map $Z[x]$
  : 다양한 크기의 Dirac impulse 의 정규 그리드로 구성됨.
  (≈ infinit 2D grid)

- $z(x) = (\phi_s * Z)(x)$
  ↑
  continous convolution

$$\mathbf{f}(z) = \phi_{s'} * \mathbf{F}(\text{III}_s \odot z)$$
$$\mathbf{F}(Z) = \text{III}_{s'} \odot \mathbf{f}(\phi_s * Z),$$

$$\text{III}_s(x) = \sum_{x \in \mathbb{Z}^2} \delta\left(x - \left(x + \tfrac{1}{2}\right)\tfrac{1}{s}\right)$$

$$Ш_T(t) \triangleq \sum_{k=-\infty}^{\infty} \delta(t - kT) = \frac{1}{T} Ш\left(\frac{t}{T}\right)$$

2. Equivariance via continuous signal interpretation

➢ 본 논문에서는 discrete domin과 continous domain 사이를 자유롭게 넘나들 수 있도록 도와주는 operation에 대해서 소개함

- sampling: continuous → discrete (by Dirac Comb)
- interpolation: discrete → continuous (by ideal interpolation filter)
- Nyquist-Shannon Samping Theorem
- 만약 신호가 bandlimited 신호이고, 표본화 주파수가 신호의 대역의 두 배 이상이라면 표본으로부터 연속 시간 기저 대역 신호를 완전히 재구성할 수 있음
- 입력 신호의 최고 주파수 f_max 의 2배 이상으로 모든 신호들을 균일하게 sampling 한다면, 원래 신호를 완벽하게 복원할 수 있음
- aliasing 현상: 아날로그 신호를 디지털 신호에 적용할 때, sampling 속도가 2f_max 보다 작을 경우 아날로그 입력 신호에서 일부 최고 주파수 성분이 디지털 출력에 올바르게 출력되지 않음. 따라서 이 디지털 신호를 다시금 아날로그 신호로 변환하고자 할 때, 원래 주파수에 없던 잘못된 주파수 성분이 나타남.

- 이 이론에 따라 신호를 sampling 하고 나면, sampling 된 discrete feature map Z(x) 이 나중에 continuous domain으로 복원하기 위한 충분한 정보를 가지고 있음을 시사함

time domain

$$\Rightarrow \quad x(t) = h_r(t) * x_\delta(t)$$

$$H_r(\omega) = T_s \,\mathrm{rect}\left(\frac{\omega}{2\cdot\frac{\omega_s}{2}}\right)$$

$$x_\delta(t) = x(t)\cdot\sum_{n=-\infty}^{\infty}\delta(t-nT_s) = \sum_{n=-\infty}^{\infty} x(nT_s)\,\delta(t-nT_s)$$

$$\mathrm{rect}\left(\frac{\omega}{2a}\right) \Leftrightarrow \frac{a}{\pi}\,\mathrm{sinc}(at), \quad \mathrm{sinc}(x) = \frac{\sin x}{x}$$

$$\Rightarrow \quad h_r(\omega) = T_s\,\frac{\omega_s}{2\pi}\,\mathrm{sinc}\left(\frac{\omega_s t}{2}\right) = \mathrm{sinc}\left(\frac{\omega_s t}{2}\right)$$

$$\omega_s = 2\pi f_s = \frac{2\pi}{T_s}$$

$$\Rightarrow \quad x(t) = \mathrm{sinc}\left(\frac{\omega_s t}{2}\right) * \sum_{n=-\infty}^{\infty} x(nT_s)\,\delta(t-nT_s) = \sum_{n=-\infty}^{\infty} x(nT_s)\,\mathrm{sinc}\left[\omega_s(t-nT_s)/2\right]$$

"infinit weighted sum of time-shifted sinc function"

2.1 Equivariant network layers

Operation $\mathbf{f}$ is equivariant with respect to a spatial transformation $\mathbf{t}$ of the 2D plane if it commutes with it in the continuous domain: $\mathbf{t} \circ \mathbf{f} = \mathbf{f} \circ \mathbf{t}$. We note that when inputs are bandlimited to $s/2$, an equivariant operation must not generate frequency content above the output bandlimit of $s'/2$, as otherwise no faithful discrete output representation exists.

We focus on two types of equivariance in this paper: translation and rotation. In the case of rotation the spectral constraint is somewhat stricter — rotating an image corresponds to rotating the spectrum, and in order to guarantee the bandlimit in both horizontal and vertical direction, the spectrum must be limited to a disc with radius $s/2$. This applies to both the initial network input as well as the bandlimiting filters used for downsampling, as will be described later.

- 본 논문에서는 2가지 tranformation(translation, rotation)과 전형적인 generator network의 4가지 operations(convolution, upsampling, downsampling, nonlinearity)에 대해서 equivariant 한 지 확인해주어야 함
- 또한, aliasing이 없으려면 nyquist sampling을 했을 때 이상한 high frequency가 없어야 함. 즉 low-path filtering이 output까지 유지되고 있는지를 확인해줘야 함.

2.1 Equivariant network layers

Convolution

Convolution Consider a standard convolution with a discrete kernel K. We can interpret K as living in the same grid as the input feature map, with sampling rate s. The discrete-domain operation is simply $\mathbf{F}_{\text{conv}}(Z) = K * Z$, and we obtain the corresponding continuous operation from Eq. 1:

$$\mathrm{f}_{\text{conv}}(z) = \phi_s * \left(K * (\text{Ш}_s \odot z)\right) = K * \left(\phi_s * (\text{Ш}_s \odot z)\right) = K * z \tag{2}$$

due to commutativity of convolution and the fact that discretization followed by convolution with ideal low-pass filter, both with same sampling rate s, is an identity operation, i.e., $\phi_s * (\text{Ш}_s \odot z) = z$. In other words, the convolution operates by continuously sliding the discretized kernel over the continuous representation of the feature map. This convolution introduces no new frequencies, so the bandlimit requirements for both translation and rotation equivariance are trivially fulfilled.

Convolution also commutes with translation in the continuous domain, and thus the operation is equivariant to translation. For rotation equivariance, the discrete kernel K needs to be radially symmetric. We later show in Section 3.2 that trivially symmetric 1×1 convolution kernels are, despite their simplicity, a viable choice for rotation equivariant generative networks.

2.1 Equivariant network layers

Upsampling and downsampling Ideal upsampling does not modify the continuous representation. Its only purpose is to increase the output sampling rate ($s' > s$) to add headroom in the spectrum where subsequent layers may introduce additional content. Translation and rotation equivariance follow directly from upsampling being an identity operation in the continuous domain. With $\mathrm{f_{up}}(z) = z$, the discrete operation according to Eq. 1 is $\mathbf{F}_{\mathrm{up}}(Z) = \mathrm{III}_{s'} \odot (\phi_s * Z)$. If we choose $s' = ns$ with integer n, this operation can be implemented by first interleaving Z with zeros to increase its sampling rate and then convolving it with a discretized filter $\mathrm{III}_{s'} \odot \phi_s$.

In downsampling, we must low-pass filter z to remove frequencies above the output bandlimit, so that the signal can be represented faithfully in the coarser discretization. The operation in continuous domain is $\mathrm{f_{down}}(z) = \psi_{s'} * z$, where an ideal low-pass filter $\psi_s := s^2 \cdot \phi_s$ is simply the corresponding interpolation filter normalized to unit mass. The discrete counterpart is $\mathbf{F}_{\mathrm{down}}(Z) = \mathrm{III}_{s'} \odot (\psi_{s'} * (\phi_s * Z)) = 1/s^2 \cdot \mathrm{III}_{s'} \odot (\psi_{s'} * \psi_s * Z) = (s'/s)^2 \cdot \mathrm{III}_{s'} \odot (\phi_{s'} * Z)$. The latter equality follows from $\psi_s * \psi_{s'} = \psi_{\min(s,s')}$. Similar to upsampling, downsampling by an integer fraction can be implemented with a discrete convolution followed by dropping sample points. Translation equivariance follows automatically from the commutativity of $\mathrm{f_{down}}(z)$ with translation, but for rotation equivariance we must replace $\phi_{s'}$ with a radially symmetric filter with disc-shaped frequency response. The ideal such filter [9] is given by $\phi_s^\circ(\boldsymbol{x}) = \mathrm{jinc}(s\|\boldsymbol{x}\|) = 2J_1(\pi s\|\boldsymbol{x}\|)/(\pi s\|\boldsymbol{x}\|)$, where J_1 is the first order Bessel function of the first kind.

2.1 Equivariant network layers

Nonlinearity Applying a pointwise nonlinearity σ in the discrete domain does not commute with fractional translation or rotation. However, in the continuous domain, any pointwise function commutes trivially with geometric transformations and is thus equivariant to translation and rotation. Fulfilling the bandlimit constraint is another question — applying, e.g., ReLU in the continuous domain may introduce arbitrarily high frequencies that cannot be represented in the output.

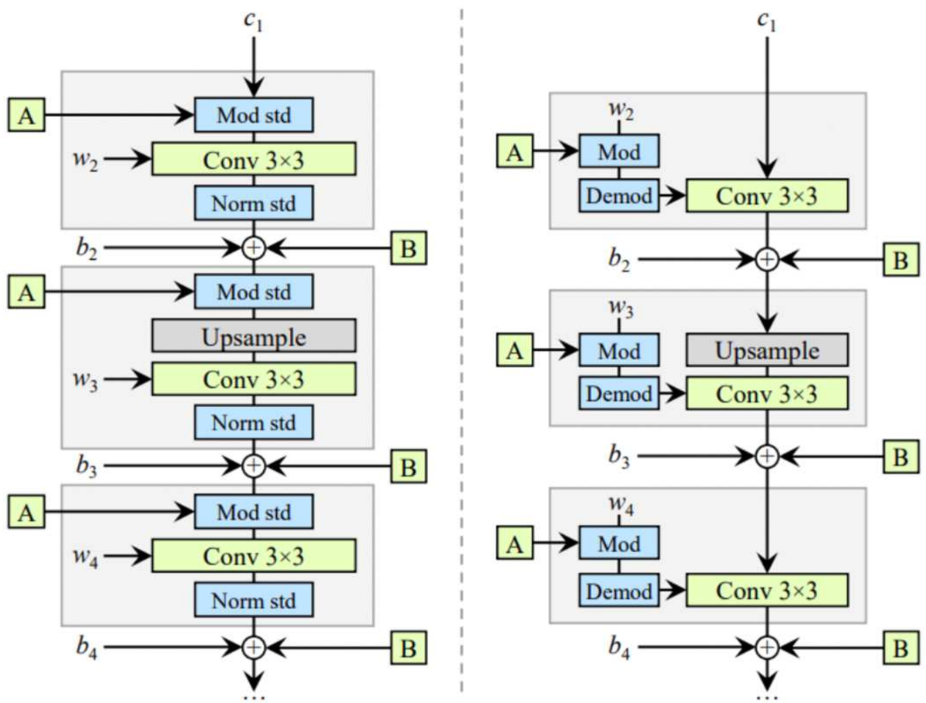
A natural solution is to eliminate the offending high-frequency content by convolving the continuous result with the ideal low-pass filter ψ_s. Then, the continuous representation of the nonlinearity becomes $\mathrm{f}_\sigma(z) = \psi_s * \sigma(z) = s^2 \cdot \phi_s * \sigma(z)$ and the discrete counterpart is $\mathrm{F}_\sigma(Z) = s^2 \cdot \mathrm{III}_s \odot (\phi_s * \sigma(\phi_s * Z))$ (see Figure 2, right). This discrete operation cannot be realized without temporarily entering the continuous representation. We approximate this by upsampling the signal, applying the nonlinearity in the higher resolution, and downsampling it afterwards. Even though the nonlinearity is still performed in the discrete domain, we have found that only a $2\times$ temporary resolution increase is sufficient for high-quality equivariance. For rotation equivariance, we must use the radially symmetric interpolation filter ϕ_s° in the downsampling step, as discussed above.

Note that nonlinearity is the only operation capable of generating novel frequencies in our formulation, and that we can limit the range of these novel frequencies by applying a reconstruction filter with a lower cutoff than $s/2$ before the final discretization operation. This gives us precise control over how much new information is introduced by each layer of a generator network (Section 3.2).

3. Practical application to generator network

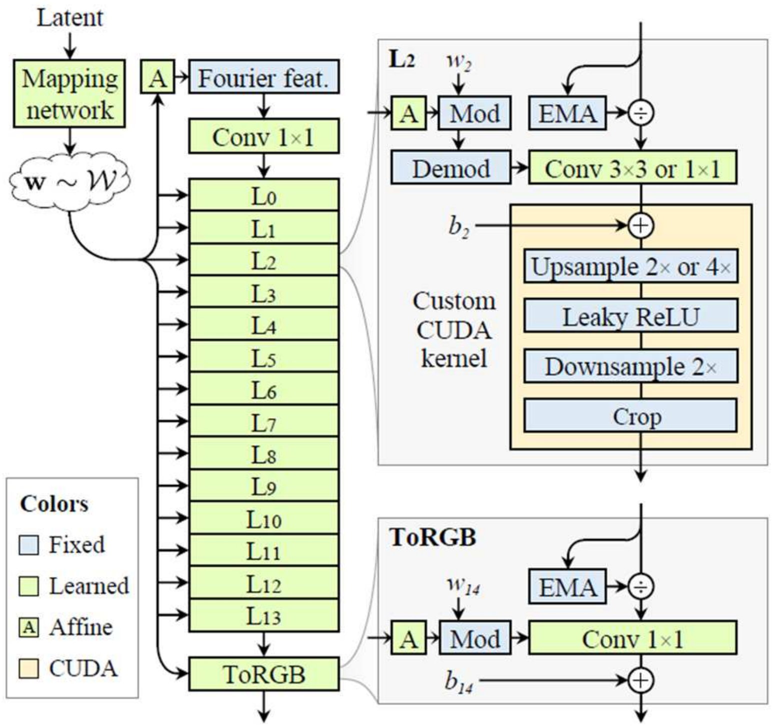
Discriminator

- alias-free-gan에서는 stylegan2의 discriminator 구조를 유지함



(c) Revised architecture (d) Weight demodulation

Generator



3. Practical application to generator network

Generator

The StyleGAN2 generator consists of two parts. First, a *mapping network* transforms an initial, normally distributed latent to an intermediate latent code $\mathbf{w} \sim \mathcal{W}$. Then, a *synthesis network* $\mathbf{G}$ starts from a learned $4\times4\times512$ constant Z_0 and applies a sequence of N layers — consisting of convolutions, nonlinearities, upsampling, and per-pixel noise — to produce an output image $Z_N = \mathbf{G}(Z_0; \mathbf{w})$. The intermediate latent code $\mathbf{w}$ controls the modulation of the convolution kernels in $\mathbf{G}$. The layers follow a rigid $2\times$ upsampling schedule, where two layers are executed at each resolution and the number of feature maps is halved after each upsampling. Additionally, StyleGAN2 employs skip connections, mixing regularization [33], and path length regularization.

- *Mapping Network*: initial, normally distributed latent code z 를 intermediate latent code w ~ W 로 transform 하는 기법을 사용
- *Synthesis network G*: learned constant input 4x4x412 Z_0 에서 N개의 layer를 거쳐 output image Z_n = G(Z_0;w) 를 생성함
- N개의 layer: consisting of convolutions, nonlinearities, upsampling, and per-pixel noise
- skip connection, mixing regularization, path length regularization 기법들도 도입함

- Generator의 operation들을 equivariance 하게 만드는 것이 이 논문의 핵심임

3. Practical application to generator network

Generator

- 각 operation이 얼마나 equivariance 한지 평가할 수 있는 방법도 함께 report함
- EQ-T / EQ-R : 이 score가 높을 수록 translation / rotation에 대해 equivariance 하는 것

For translation equivariance, we report the peak signal-to-noise ratio (PSNR) in decibels (dB) between two sets of images, obtained by translating the input and output of the synthesis network by a random amount, resembling the definition by Zhang [69]:

$$\text{EQ-T} = 10 \cdot \log_{10}\left(I_{max}^2 / \mathbb{E}_{\mathbf{w}\sim\mathcal{W}, x\sim\mathcal{X}^2, p\sim\mathcal{V}, c\sim\mathcal{C}}\left[\left(\mathbf{g}(\mathbf{t}_x[z_0]; \mathbf{w})_c(p) - \mathbf{t}_x[\mathbf{g}(z_0; \mathbf{w})]_c(p)\right)^2\right]\right) \quad (3)$$

Each pair of images, corresponding to a different random choice of $\mathbf{w}$, is sampled at integer pixel locations p within their mutually valid region $\mathcal{V}$. Color channels c are processed independently, and the intended dynamic range of generated images $-1 \ldots +1$ gives $I_{max} = 2$. Operator $\mathbf{t}_x$ implements spatial translation with 2D offset x, here drawn from distribution $\mathcal{X}^2$ of integer offsets. We define an analogous metric EQ-R for rotations, with the rotation angles drawn from $\mathcal{U}(0°, 360°)$. Appendix E gives implementation details and our accompanying videos highlight the practical relevance of different dB values.

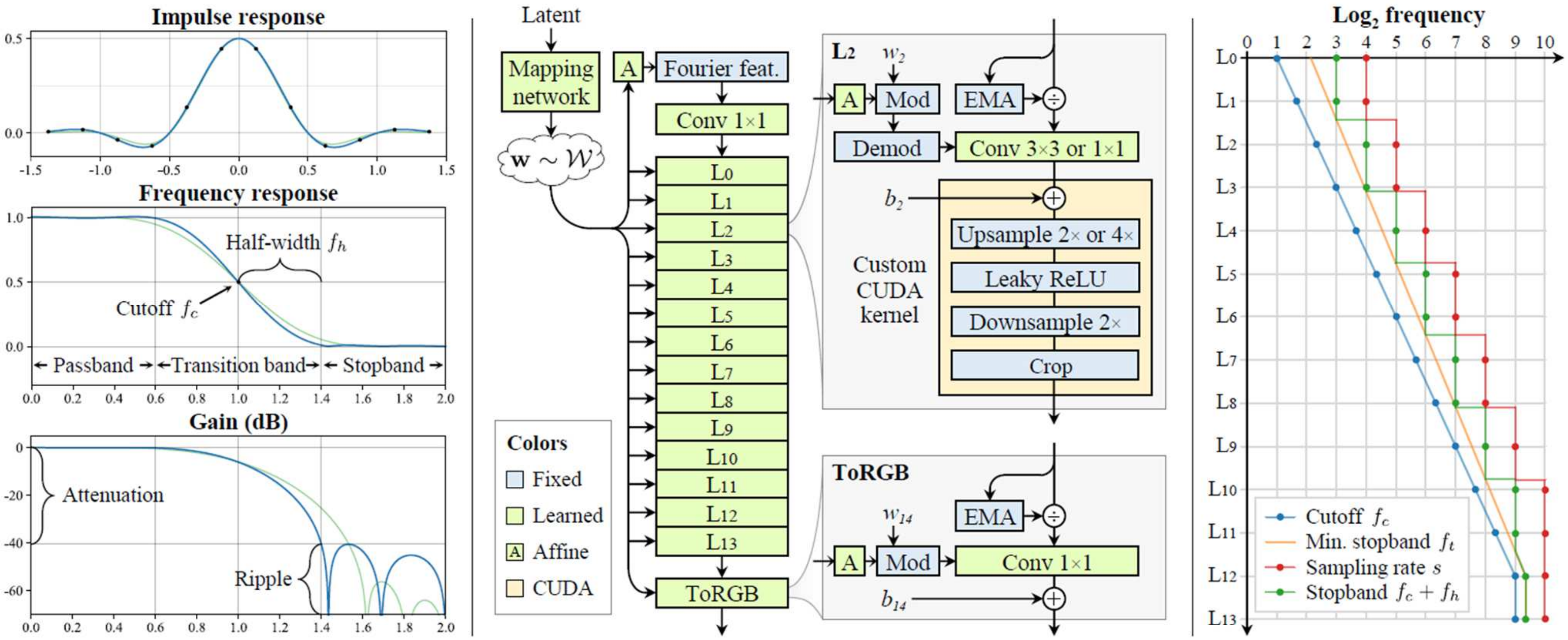
3. Practical application to generator network

Generator

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

	Parameter	FID↓	EQ-T↑	EQ-R↑	Time	Mem.
	Filter size $n = 4$	4.72	57.49	39.70	**0.84**×	**0.99**×
*	Filter size $n = 6$	**4.50**	**66.65**	40.48	1.00×	1.00×
	Filter size $n = 8$	4.66	65.57	**42.09**	1.18×	1.01×
	Upsampling $m = 1$	**4.38**	39.96	36.42	**0.65**×	**0.87**×
*	Upsampling $m = 2$	4.50	66.65	40.48	1.00×	1.00×
	Upsampling $m = 4$	4.57	**74.21**	**40.97**	2.31×	1.62×
	Stopband $f_{t,0} = 2^{1.5}$	4.62	51.10	29.14	**0.86**×	**0.90**×
*	Stopband $f_{t,0} = 2^{2.1}$	**4.50**	66.65	40.48	1.00×	1.00×
	Stopband $f_{t,0} = 2^{3.1}$	4.68	**73.13**	**41.63**	1.36×	1.25×

Figure 3: Results for FFHQ-U (unaligned FFHQ) at 256^2. **Left:** Training configurations. FID is computed between 50k generated images and all training images [26, 32]; lower is better. EQ-T and EQ-R are our equivariance metrics in decibels (dB); higher is better. **Right:** Parameter ablations using our final configuration (R) for the filter's support, magnification around nonlinearities, and the minimum stopband frequency at the first layer. * indicates our default choices.



(a) Filter design concepts (b) Our alias-free StyleGAN3 generator architecture (c) Flexible layers

Figure 4: **(a)** 1D example of a 2× upsampling filter with $n = 6$, $s = 2$, $f_c = 1$, and $f_h = 0.4$ (blue). Setting $f_h = 0.6$ makes the transition band wider (green), which reduces the unwanted stopband ripple and thus leads to stronger attenuation. **(b)** Our alias-free generator, corresponding to configs T and R in Figure 3. The main datapath consists of Fourier features and normalization (Section 3.1), modulated convolutions [34], and filtered nonlinearities (Section 3.2). **(c)** Flexible layer specifications (config T) with $N = 14$ and $s_N = 1024$. Cutoff f_c (blue) and minimum acceptable stopband frequency f_t (orange) obey geometric progression over the layers; sampling rate s (red) and actual stopband $f_c + f_h$ (green) are computed according to our design constraints.

3.1 Fourier features and baseline simplifications (configs B–D)

To facilitate exact continuous translation and rotation of the input z_0, we replace the learned input constant in StyleGAN2 with Fourier features [56, 66], which also has the advantage of naturally defining a spatially infinite map. We sample the frequencies uniformly within the circular frequency band $f_c = 2$, matching the original 4×4 input resolution, and keep them fixed over the course of training. This change (configs A and B in Figure 3, left) slightly improves FID and, crucially, allows us to compute the equivariance metrics without having to approximate the operator $\mathbf{t}$. This baseline architecture is far from being equivariant; our accompanying videos show that the output images deteriorate drastically when the input features are translated or rotated from their original position.

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

➢ 이미지를 continuous 하게 transformation 하기 위해 learned constant input을 Fourier fearture로 변경함

- 기존의 learned const input은 positional encoding 정보를 주기는 하지만, 좌표계가 좋지 않아 어떠한 방식으로 transformation이 작동되는지 알기 어려웠으며 이미지 공간을 explicit하게 표현했다고 하기에도 부족한 점이 많았음
- StyleGAN2에서는 signal의 크기만을 encoding했다면, StyleGAN3에서는 새로운 coordinate system (SPE)을 도입하여 signal 뿐만 아니라 phase에 대한 정보도 잘 encoding하고자 함.
- StyleGAN3는 Fourier Feature (in continous한 frequency domain)으로 input을 변경하여 infinit domain으로 확장하였고, 동시에 Postional Encoding 정보도 explicit하게 줄 수 있게 됨

➢ Fourier features는 stylegan3의 코드에서 *SyntehsisInput*에 구현되어있음

- SyntehsisInput block
- intermediate latent code w 를 input으로 받아 affine 변환을 한 후,
- 이 값을 learned transformation: (1) 먼저 image를 rotation한 후 (2) translation (3) 마지막으로는 user-specified transform
- sampling grid를 만들어서 fourier feature로 변환

3.1 Fourier features and baseline simplifications (configs B–D)

➢ Baseline Simplification

Next, we remove the per-pixel noise inputs because they are strongly at odds with our goal of a natural transformation hierarchy, i.e., that the exact sub-pixel position of each feature is exclusively inherited from the underlying coarse features. While this change (config C) is approximately FID-neutral, it fails to improve the equivariance metrics when considered in isolation.

To further simplify the setup, we decrease the mapping network depth as recommended by Karras et al. [32] and disable mixing regularization and path length regularization [34]. Finally, we also eliminate the output skip connections. We hypothesize that their benefit is mostly related to gradient magnitude dynamics during training and address the underlying issue more directly using a simple normalization before each convolution. We track the exponential moving average $\sigma^2 = \mathbb{E}[x^2]$ over all pixels and feature maps during training, and divide the feature maps by $\sqrt{\sigma^2}$. In practice, we bake the division into the convolution weights to improve efficiency. These changes (config D) bring FID back to the level of original StyleGAN2, while leading to a slight improvement in translation equivariance.

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

(1) per-pixel noise inputs 제거

- StyleGAN2에 삽입되는 per-pixel noise는 이미지의 세부적인 요소들을 독립적이게 학습하도록 만들기 때문에, 이미지가 hierarchical하게 학습되지 못함
- noise를 제거하면, figure5 (논문의 figure3)를 보면 FID가 개선되지는 않지만 훨씬 equivariance 할 수 있음

(2) StyleGAN2-ADA에서 처럼 the mapping network depth를 줄임

(3) disable mixing regularization and path length regularization

(4) output skip connections 제거

- FID score를 높이기 위해 2,3,4를 했었지만, 모델을 단순화하기 위해 FID는 약간 포기하고 2,3,4를 제거하였다고 함

3.2 Step-by-step redesign motivated by continuous interpretation

➢ Boundaries and upsampling (config E)

- **Boundaries** : 본 논문에서는 feature map을 무한한 공간으로 확장했다고 가정함. 따라서 target canva에 어느 정도의 margi을 준 후, high-layer로 갈수록 이 확장된 canva를 crop함
  - ✓ border padding이 내부 이미지의 coordinate 값을 어느정도 갖고 있기 때문에 border를 explicit하게 extension하는 과정이 필요함
  - ✓ 실험 결과, 10-pixel margin 정도면 충분하여 이를 사용했다 한다.

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

- **Upsampling**: 기존의 bilinear 2X upsampling filter를 windowed sinc filter로 대체하여 low-pass filtering도 함께 하도록 함.

Motivated by our theoretical model, we replace the bilinear $2\times$ upsampling filter with a better approximation of the ideal low-pass filter. We use a windowed sinc filter with a relatively large Kaiser window [41] of size $n = 6$, meaning that each output pixel is affected by 6 input pixels in upsampling and each input pixel affects 6 output pixels in downsampling. Kaiser window is a particularly good choice for our purposes, because it offers explicit control over the transition band and attenuation (Figure 4a). In the remainder of this section, we specify the transition band explicitly and compute the remaining parameters using Kaiser's original formulas (Appendix C). For now, we choose to employ *critical sampling* and set the filter cutoff $f_c = s/2$, i.e., exactly at the bandlimit, and transition band half-width $f_h = (\sqrt{2} - 1)(s/2)$. Recall that sampling rate s equals the width of the canvas in pixels, given our definitions in Section 2.

The improved handling of boundaries and upsampling (config E) leads to better translation equivariance. However, FID is compromised by 16%, probably because we started to constrain what the feature maps can contain. In a further ablation (Figure 3, right), smaller resampling filters ($n = 4$) hurt translation equivariance, while larger filters ($n = 8$) mainly increase training time.

3.2 Step-by-step redesign motivated by continuous interpretation

➢ Filtered nonlinearities (config F)

- ReLU가 당연히 equivariance는 만족하지만, bandlimit를 지키지 않으면 aliasing이 생길 수도 있다고 보고함. 따라서 non-linearity function을 지날 때 low-path filtering을 꼭 해야 한다.
- upsample-leaky ReLU-downsample의 sequence가 CUDA kernel에서 효과적으로 연산되도록 최적화함 (10배 빨라짐 + memory saving)
- upsampling + downsampling는 실험결과 m = 2 면 충분하다고 함

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

	Parameter	FID↓	EQ-T↑	EQ-R↑	Time	Mem.
	Filter size $n = 4$	4.72	57.49	39.70	**0.84**×	**0.99**×
*	Filter size $n = 6$	**4.50**	**66.65**	40.48	1.00×	1.00×
	Filter size $n = 8$	4.66	65.57	**42.09**	1.18×	1.01×
	Upsampling $m = 1$	**4.38**	39.96	36.42	**0.65**×	**0.87**×
*	Upsampling $m = 2$	4.50	66.65	40.48	1.00×	1.00×
	Upsampling $m = 4$	4.57	**74.21**	**40.97**	2.31×	1.62×
	Stopband $f_{t,0} = 2^{1.5}$	4.62	51.10	29.14	**0.86**×	**0.90**×
*	Stopband $f_{t,0} = 2^{2.1}$	**4.50**	66.65	40.48	1.00×	1.00×
	Stopband $f_{t,0} = 2^{3.1}$	4.68	**73.13**	**41.63**	1.36×	1.25×

3.2 Step-by-step redesign motivated by continuous interpretation

➢ Non-critical sampling (config G)

- aliasing은 generator의 equivariance를 망치는 원인이기도 하다. 따라서 각 각의 layer를 지날 때 aliasing이 생기지 않도록 해야 합니다.
- config G에서는 저해상도의 layer에서 aliasing이 안생기도록 cutoff frequency를 f_c = s/2 − f_h 로 낮추어 주었습니다.

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

Non-critical sampling (config G) The critical sampling scheme — where filter cutoff is set exactly at the bandlimit — is ideal for many image processing applications as it strikes a good balance between antialiasing and the retention of high-frequency detail [58]. However, our goals are markedly different because aliasing is highly detrimental for the equivariance of the generator. While high-frequency detail is important in the output image and thus in the highest-resolution layers, it is less important in the earlier ones given that their exact resolutions are somewhat arbitrary to begin with.

To suppress aliasing, we can simply lower the cutoff frequency to $f_c = s/2 - f_h$, which ensures that all alias frequencies (above $s/2$) are in the stopband.[3] For example, lowering the cutoff of the blue filter in Figure 4a would move its frequency response left so that the the worst-case attenuation of alias frequencies improves from 6 dB to 40 dB. This *oversampling* can be seen as a computational cost of better antialiasing, as we now use the same number of samples to express a slower-varying signal than before. In practice, we choose to lower f_c on all layers except the highest-resolution ones, because in the end the generator must be able to produce crisp images to match the training data. As the signals now contain less spatial information, we modify the heuristic used for determining the number of feature maps to be inversely proportional to f_c instead of the sampling rate s. These changes (config G) further improve translation equivariance and push FID below the original StyleGAN2.

3.2 Step-by-step redesign motivated by continuous interpretation

➢ Transformed Fourier Features(config H)

- StyleGAN3 Generator의 layer들은 equivariant 하기 때문에 unaligned dataset이나 임의로 변형시킨 dataset에 대해서도 학습이 잘됨. (만약 intermediate feature z_i를 변형시키면 final image z_N 도 변형되어 생성함)
- 그러나 layer 자체에서 global 하게 transformation 하기에는 layer의 capability가 작다. 따라서 Input Fourier Features 자체를 변형시키는 방식으로 생성되는 이미지도 transformation 되도록 함.
- learned affine layer를 통해 input Fourier Features 가 global translation or rotation 되도록 만들어줌.

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

3.2 Step-by-step redesign motivated by continuous interpretation

➢ **Flexible layer (config T)**

- aliasing을 없애는 것은 중요하나 이미지를 학습할 때 상위 layer로 갈수록 detail을 학습하는 것도 중요함.
- 즉, high-frequenc도 적절히 학습을 해야 하는데, low-path filterin을 너무 강하게 걸어주다 보면 aliasing은 안 생기겠지만 high-frequency (detail)가 학습되지 못함

- config T에서는 layer를 flexible하게 조절함

- 저해상도 layer에서는 aliasing이 안 생기도록 lower cutoff frequency를 통해 low-path filtering을 강하게 걸어주고
- 고해상도 layer에서는 이미지의 detail을 학습하는 게 중요하므로 flexible 하게 조절하여 high-frequency를 학습하도록 함

The new layer specifications again improve translation equivariance (config T), eliminating the remaining artifacts. A further ablation (Figure 3, right) shows that $f_{t,0}$ provides an effective way to trade training speed for equivariance quality. Note that the number of layers is now a free parameter that does not directly depend on the output resolution. In fact, we have found that a fixed choice of N works consistently across multiple output resolutions and makes other hyperparameters such as learning rate behave more predictably. We use N = 14 in the remainder of this paper

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

	Parameter	FID↓	EQ-T↑	EQ-R↑	Time	Mem.
	Filter size $n = 4$	4.72	57.49	39.70	**0.84×**	**0.99×**
*	Filter size $n = 6$	**4.50**	**66.65**	40.48	1.00×	1.00×
	Filter size $n = 8$	4.66	65.57	**42.09**	1.18×	1.01×
	Upsampling $m = 1$	**4.38**	39.96	36.42	**0.65×**	**0.87×**
*	Upsampling $m = 2$	4.50	66.65	40.48	1.00×	1.00×
	Upsampling $m = 4$	4.57	**74.21**	**40.97**	2.31×	1.62×
	Stopband $f_{t,0} = 2^{1.5}$	4.62	51.10	29.14	**0.86×**	**0.90×**
*	Stopband $f_{t,0} = 2^{2.1}$	**4.50**	66.65	40.48	1.00×	1.00×
	Stopband $f_{t,0} = 2^{3.1}$	4.68	**73.13**	**41.63**	1.36×	1.25×

3.2 Step-by-step redesign motivated by continuous interpretation

➢ Flexible layer (config T)

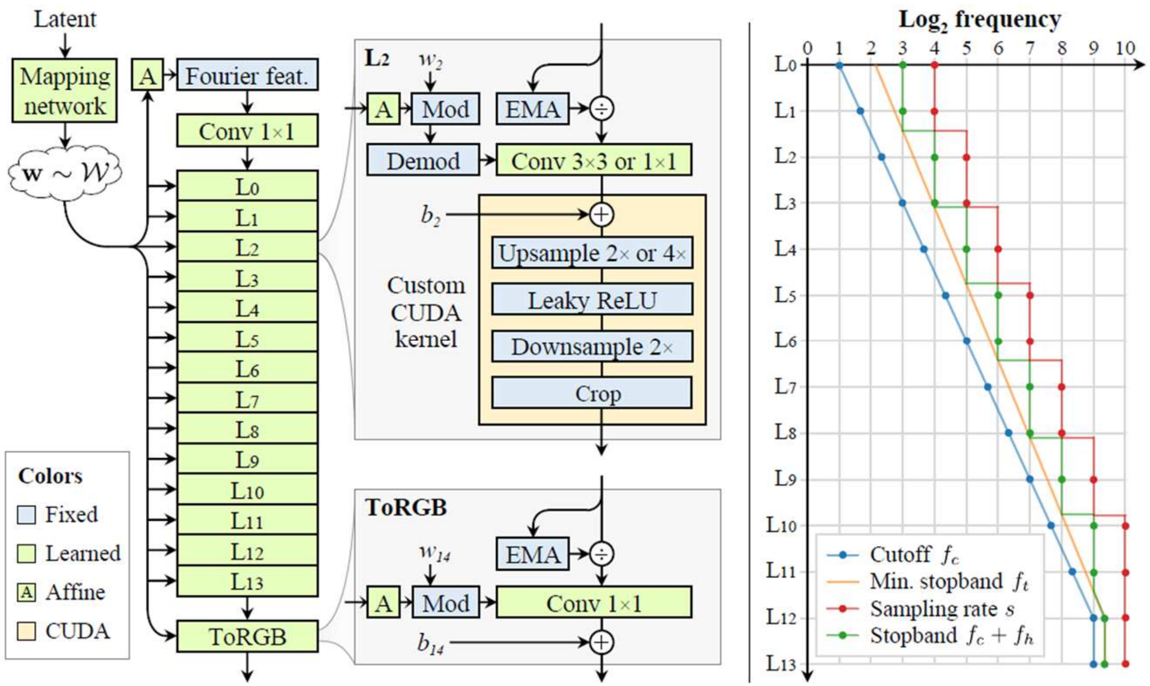


Figure 4c illustrates an example progression of filter parameters in a 14-layer generator with two critically sampled full-resolution layers at the end.

The cutoff frequency grows geometrically from $f_c = 2$ in the first layer to $f_c = s_N = 2$ in the first critically sampled layer. We choose the minimum acceptable stopband frequency to start at $f_{t,0} = 2^{2.1}$, and it grows geometrically but slower than the cutoff frequency. In our tests, the stopband target at the last layer is $f_t = f_c \cdot 2^{0.3}$, but the progression is halted at the first critically sampled layer. Next, we set the sampling rate s for each layer so that it accommodates frequencies up tof_t, rounding up to the next power of two without exceeding the output resolution. Finally, to maximize the attenuation of aliasing frequencies, we set the transition band half-width to $f_h = \max\left(\frac{s}{2}, f_t\right) - f_c$, i.e., making it as wide as possible within the limits of the sampling rate, but at least wide enough to reach f_t. The resulting improvement depends on how much slack is left between f_t and $s = 2$; as an extreme example, the first layer stopband attenuation improves from 42 dB to 480 dB using this scheme.

3.2 Step-by-step redesign motivated by continuous interpretation

➢ **Rotation equivariance (config R) Permalink**

- network를 rotation equivariant 하게 변형하고자 할 때에는 2가지를 변경함
  - ✓ 3x3 conv를 1x1 conv 로 변경. 대신 feature map의 수를 2배 증가
  - ✓ sinc-based downsampling filter를 radially symmetric jinc-based filter로 변경
- 학습과정에서 trainable parameter가 56% 줄어드는 효과
- FID는 비슷하며 EQ-R은 약간 향상됨

	Configuration	FID↓	EQ-T↑	EQ-R↑
A	StyleGAN2	5.14	–	–
B	+ Fourier features	4.79	16.23	10.81
C	+ No noise inputs	4.54	15.81	10.84
D	+ Simplified generator	5.21	19.47	10.41
E	+ Boundaries & upsampling	6.02	24.62	10.97
F	+ Filtered nonlinearities	6.35	30.60	10.81
G	+ Non-critical sampling	4.78	43.90	10.84
H	+ Transformed Fourier features	4.64	45.20	10.61
T	+ Flexible layers (StyleGAN3-T)	4.62	63.01	13.12
R	+ Rotation equiv. (StyleGAN3-R)	**4.50**	**66.65**	**40.48**

➢ An additional stabilization trick in this configuration.

- Early on in the training, we blur all images the discriminator sees using a Gaussian filter. We start with σ=10 pixels, which we ramp to zero over the first 200k images. This prevents the discriminator from focusing too heavily on high frequencies early on. Without this trick, config R is prone to early collapses because the generator sometimes learns to produce high frequencies with a small delay, trivializing the discriminator's task.

4. Results

Dataset	Config	FID↓	EQ-T↑	EQ-R↑
FFHQ-U 70000 img, 1024^2 Train from scratch	StyleGAN2	3.79	15.89	10.79
	StyleGAN3-T (ours)	3.67	61.69	13.95
	StyleGAN3-R (ours)	**3.66**	**64.78**	**47.64**
FFHQ 70000 img, 1024^2 Train from scratch	StyleGAN2	**2.70**	13.58	10.22
	StyleGAN3-T (ours)	2.79	61.21	13.82
	StyleGAN3-R (ours)	3.07	**64.76**	**46.62**
METFACES-U 1336 img, 1024^2 ADA, from FFHQ-U	StyleGAN2	18.98	18.77	13.19
	StyleGAN3-T (ours)	**18.75**	64.11	16.63
	StyleGAN3-R (ours)	**18.75**	**66.34**	**48.57**
METFACES 1336 img, 1024^2 ADA, from FFHQ	StyleGAN2	15.22	16.39	12.89
	StyleGAN3-T (ours)	**15.11**	**65.23**	16.82
	StyleGAN3-R (ours)	15.33	64.86	**46.81**
AFHQV2 15803 img, 512^2 ADA, from scratch	StyleGAN2	4.62	13.83	11.50
	StyleGAN3-T (ours)	**4.04**	60.15	13.51
	StyleGAN3-R (ours)	4.40	**64.89**	**40.34**
BEACHES 20155 img, 512^2 ADA, from scratch	StyleGAN2	5.03	15.73	12.69
	StyleGAN3-T (ours)	**4.32**	59.33	15.88
	StyleGAN3-R (ours)	4.57	**63.66**	**37.42**

Ablation	Translation eq.		+ Rotation eq.		
	FID↓	EQ-T↑	FID↓	EQ-T↑	EQ-R↑
* Main configuration	4.62	63.01	**4.50**	66.65	40.48
With mixing reg.	**4.60**	63.48	4.67	63.59	40.90
With noise inputs	4.96	24.46	5.79	26.71	26.80
Without flexible layers	4.64	45.20	4.65	44.74	22.52
Fixed Fourier features	5.93	64.57	6.48	66.20	41.77
With path length reg.	5.00	**68.36**	5.98	**71.64**	**42.18**
0.5× capacity	7.43	63.14	6.52	63.08	39.89
* 1.0× capacity	4.62	63.01	4.50	66.65	40.48
2.0× capacity	**3.80**	**66.61**	**4.18**	**70.06**	**42.51**
* Kaiser filter, $n = 6$	**4.62**	**63.01**	4.50	**66.65**	**40.48**
Lanczos filter, $a = 2$	4.69	51.93	**4.44**	57.70	25.25
Gaussian filter, $\sigma = 0.4$	5.91	56.89	5.73	59.53	39.43

G-CNN comparison	FID↓	EQ-T↑	EQ-R↑	Params	Time
* StyleGAN3-T (ours)	4.62	63.01	13.12	23.3M	**1.00×**
+ $p4$ symmetry [16]	4.69	61.90	17.07	21.8M	2.48×
* StyleGAN3-R (ours)	**4.50**	**66.65**	**40.48**	15.8M	1.37×

Figure 5: **Left:** Results for six datasets. We use adaptive discriminator augmentation (ADA) [32] for the smaller datasets. "StyleGAN2" corresponds to our baseline config B with Fourier features. **Right:** Ablations and comparisons for FFHQ-U (unaligned FFHQ) at 256^2. * indicates our default choices.

4. Results

➢ Internal Representations

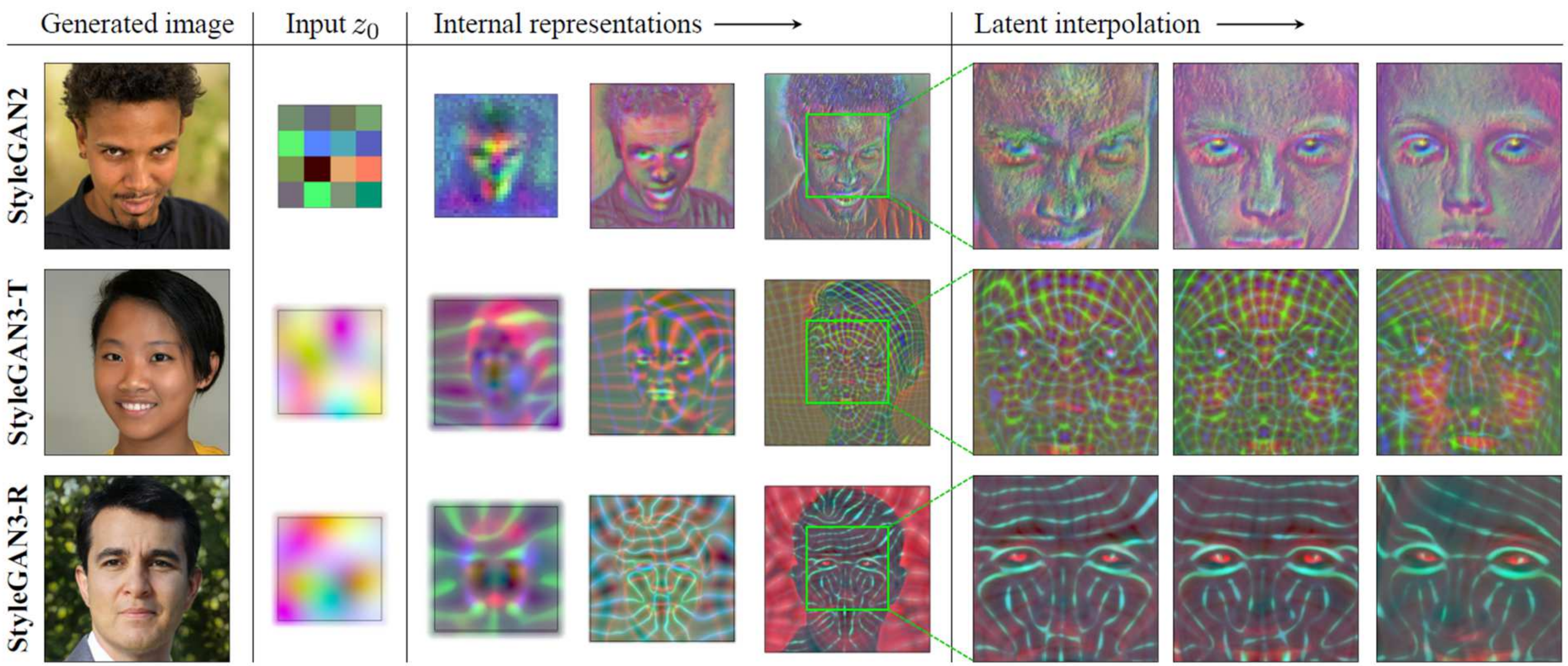


Figure 6: Example internal representations (3 feature maps as RGB) in StyleGAN2 and our generators.

- While in StyleGAN2 all feature maps seem to encode signal magnitudes, in our networks some of the maps take a different role and encode phase information instead. Clearly this is something that is needed when the network synthesizes detail on the surfaces; it needs to invent a coordinate system.
- In StyleGAN3-R, the emergent positional encoding patterns appear to be somewhat more well-defined.
- We believe that the existence of a coordinate system that allows precise localization on the surfaces of objects will prove useful in various applications, including advanced image and video editing.