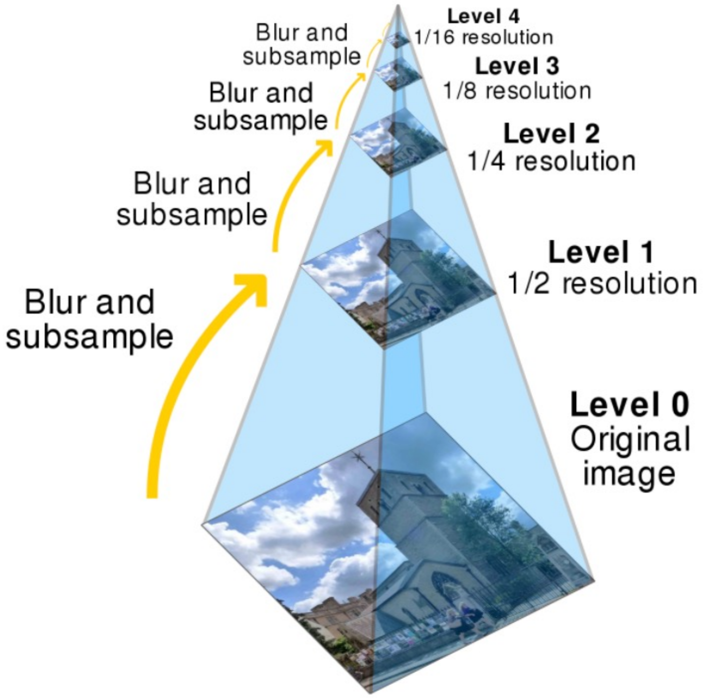
Announcement

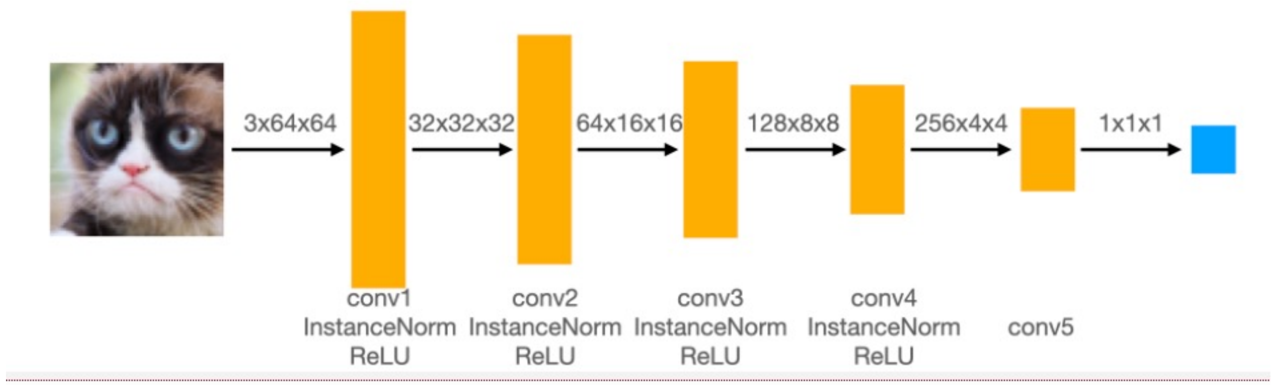
- HW1 winner: Riyaz Panjwani

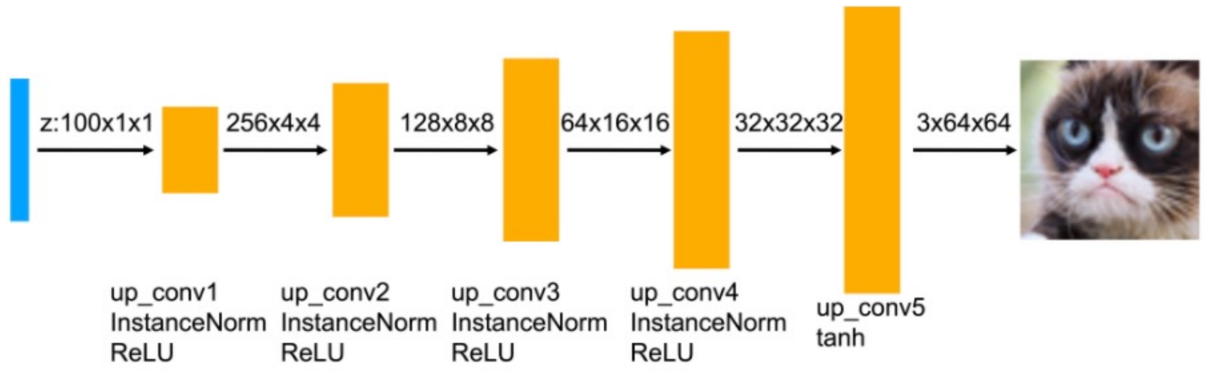
  Honorable mention: Harry Freeman



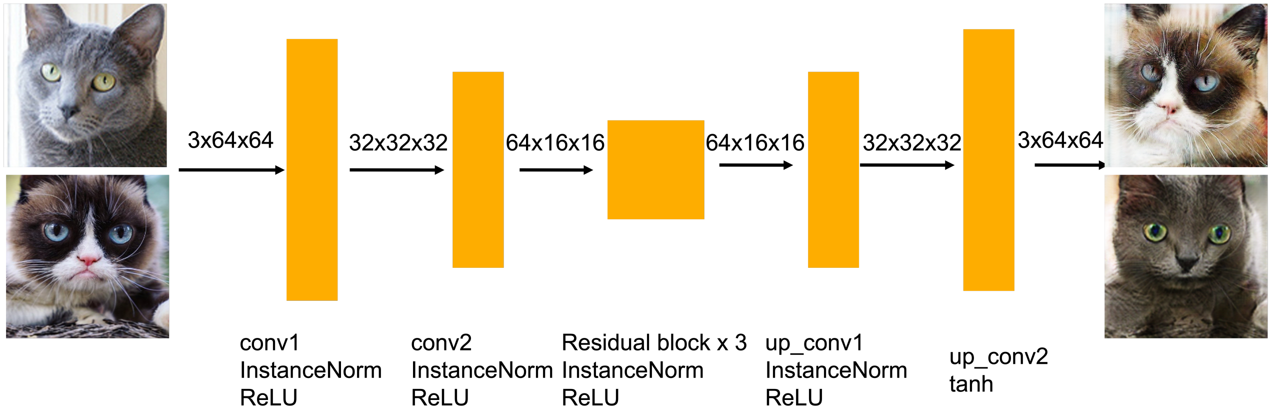
Announcement

- HW3 (due: 3/21/2022)





CycleGAN Generator



Style and Content, Texture Synthesis (part II)

Jun-Yan Zhu

16-726, Spring 2022

Many slides are from Taesung Park, Fujun Luan, etc.

photo © [Gatys et al., 2016]

Loss Functions (Image-to-Image Translation)

Style and Content

Adversarial loss

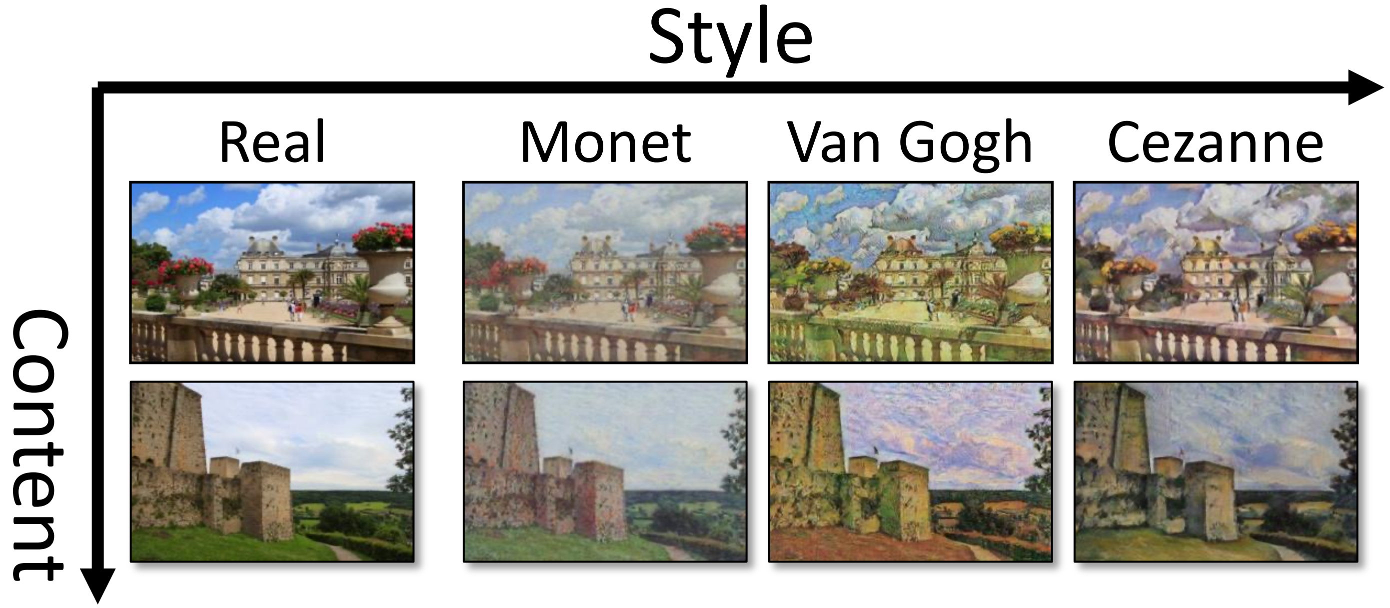
$$\mathbb{E}_x \log(1 - D_Y(G(x))) + \mathbb{E}_y \log D_Y(y)$$

$p(x) \rightarrow p(y)$ change **style**

Cycle-consistency loss

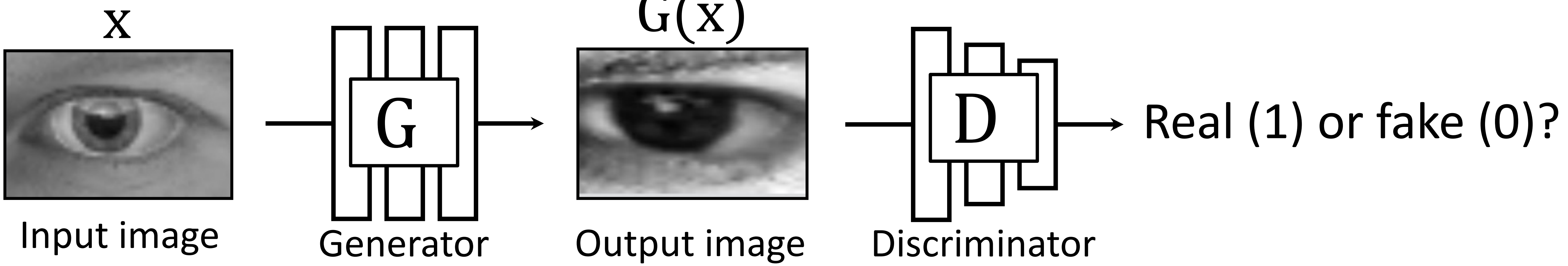
$$\mathbb{E}_x ||F(G(x)) - x||_1$$

Bidirectional: preserve **content**



Separating Style and Content
[Tenenbaum and Freeman 1996]

Style and Content

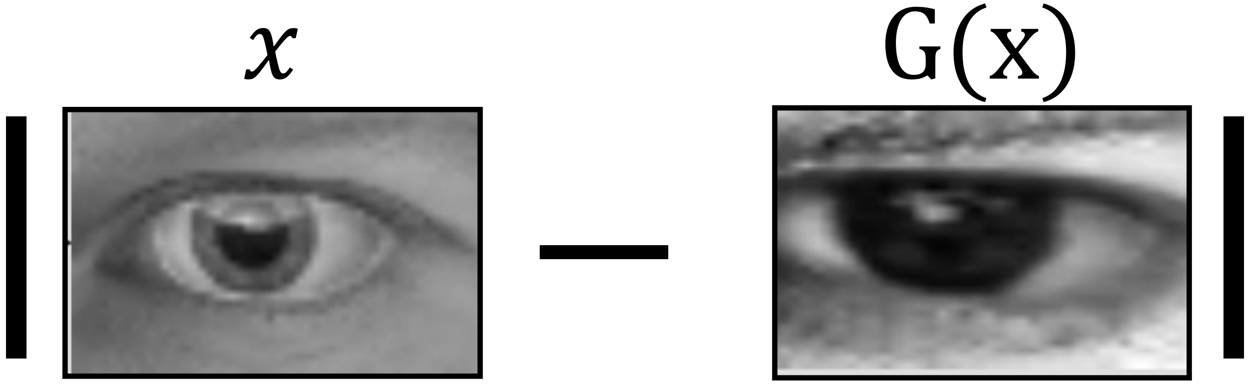


Adversarial loss (change style)

$$\mathbb{E}_x \log(1 - D_Y(G(x))) + \mathbb{E}_y \log D_Y(y)$$

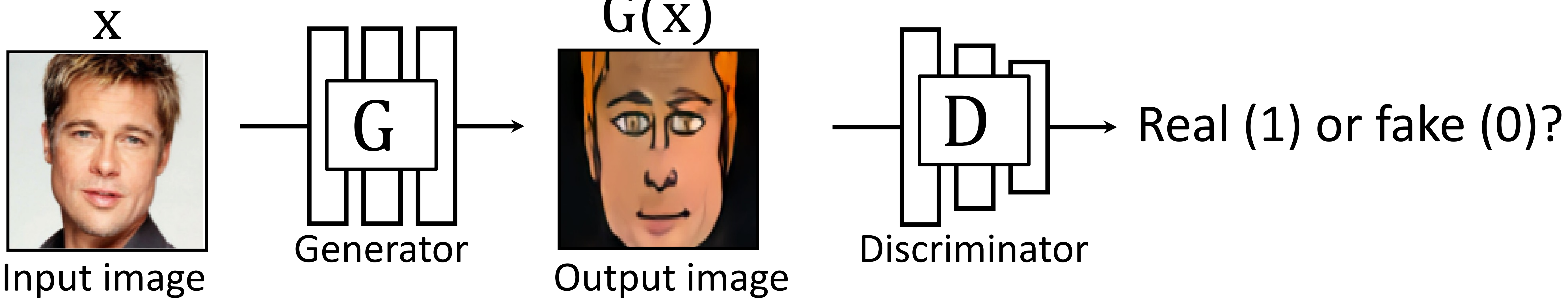
L1 loss (preserve content in pixel space)

$$\mathbb{E}_x ||G(x) - x||_1$$



SimGAN [Shrivastava et al., 2017]

Style and Content

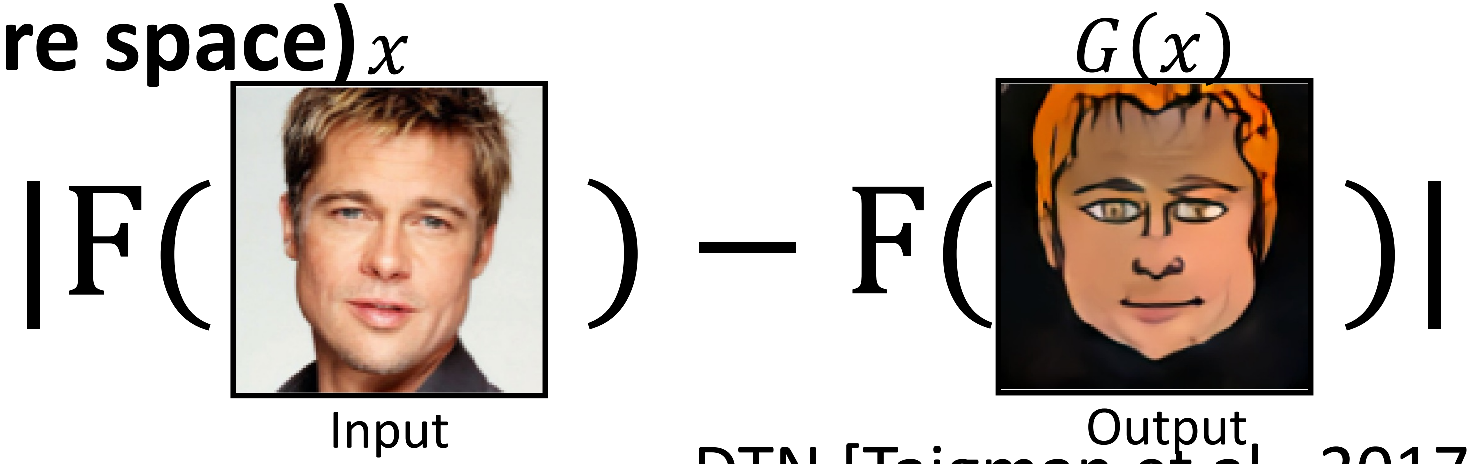


Adversarial loss (change style)

$$\mathbb{E}_x \log(1 - D_Y(G(x))) + \mathbb{E}_y \log D_Y(y)$$

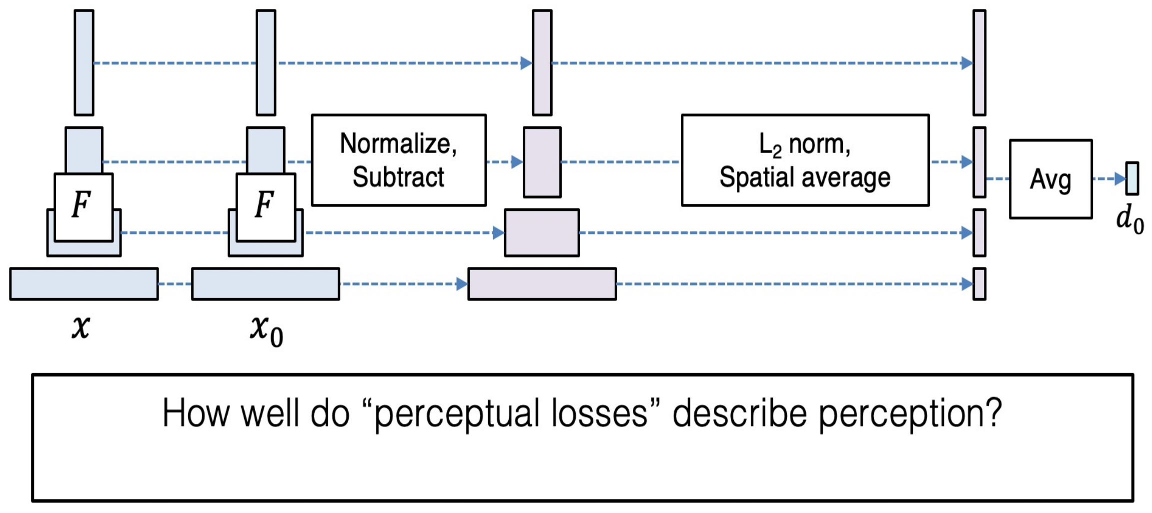
Feature loss (Preserve content in feature space)

$$\mathbb{E}_x ||F(G(x)) - F(x)||$$

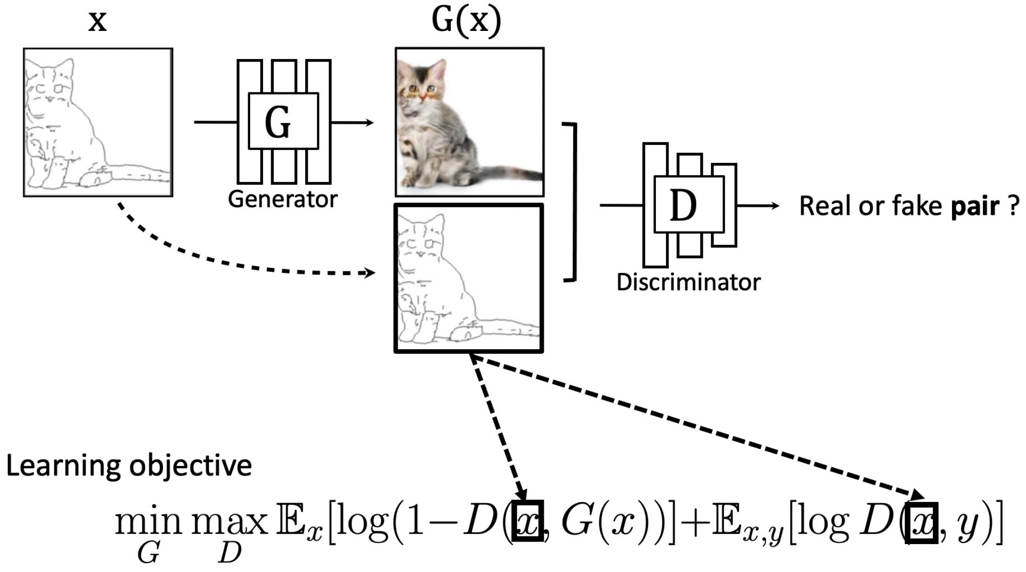


DTN [Taigman et al., 2017]

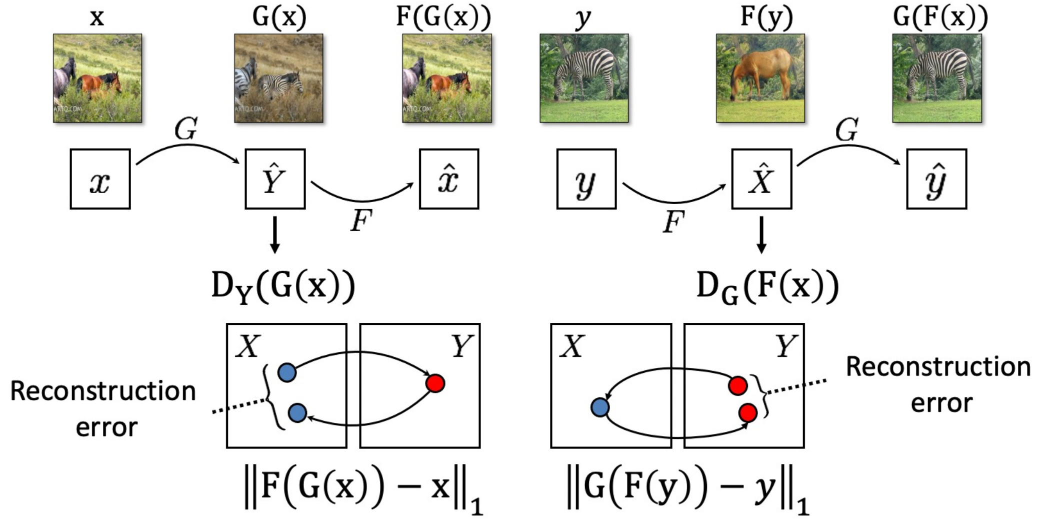
Perceptual/Feature Loss



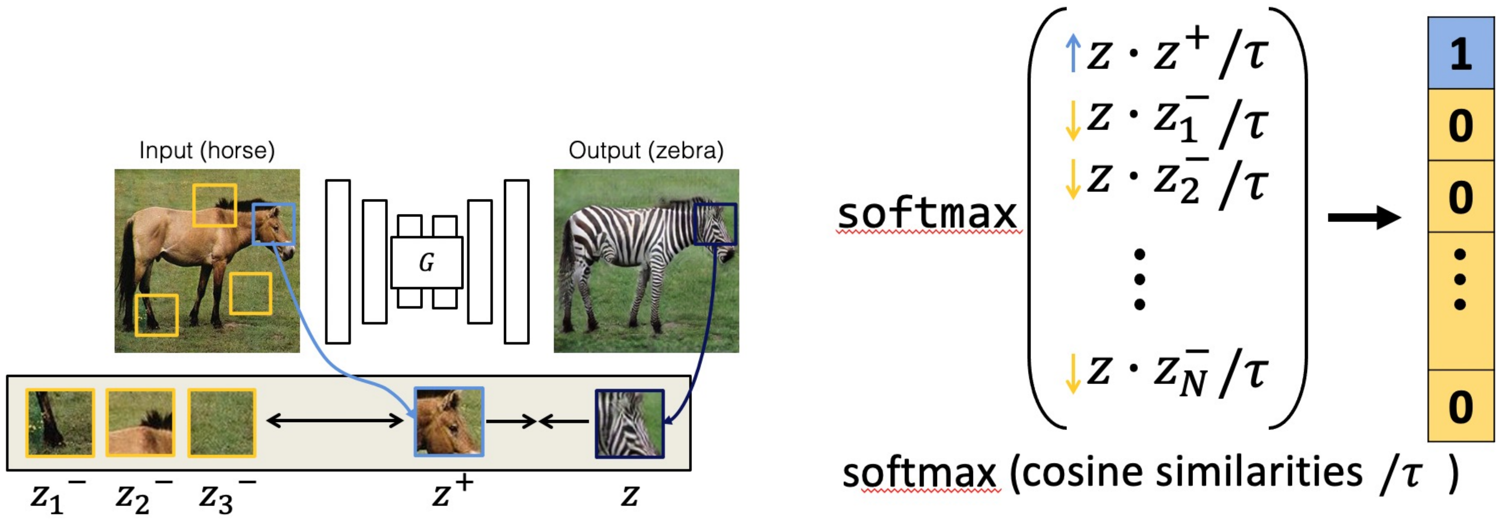
(Conditional) GAN Loss



Cycle-Consistency Loss



Patch-wise Contrastive Loss



Style and Content

- <u>Style</u>: domain-specific features

  (horse vs. zebra)

- <u>Content</u>: features shared across two domains

Loss Functions
(Neural Style Transfer)

Neural Style Transfer

content image

+

style image

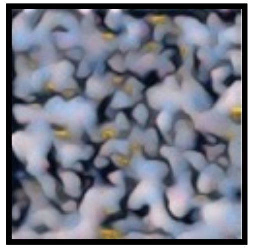
=

output result

[Gatys et al., 2016]

Style Reconstruction (Style Loss)

$\hat{y}$ y

|Gram() – Gram()|

optimized output style image

Gram = Gram Matrix of a deep network’s features (e.g., ImageNet classifier)

Style Loss

weight (j)-th layer

$$\arg\min_{\hat{y}} \sum_{j}^{M} \lambda_j ||Gram^{(j)}(\hat{y}) - Gram^{(j)}(y)||^2$$

[Gatys et al., 2016]

Computing Gram Matrix

Gram matrix:
- Cross Correlation of CNN features
- Invariant to the feature locations

$$V = [v_1, v_2, \cdots, v_n]$$

$$G_{ij} = \langle v_i, v_j \rangle \qquad G = V^\top V$$

$$Gram^{(j)}(x) = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'}.$$

h, w: pixel locations index

c: channel index

H, W: height and width of feature map

C: the number of total channels

Content Reconstruction (Perceptual Loss)

$\hat{y}$ x

|F() − F()|

optimized output content image

F is a deep network (e.g., ImageNet classifier)

Content Loss

weight (i)-th layer

$$\arg\min_{\hat{y}} \sum_{i}^{N} \lambda_i ||F^{(i)}(\hat{y}) - F^{(i)}(x)||_1$$

[Gatys et al., 2016]

Neural Style Transfer

$\hat{y}$ y

|Gram() − Gram()|

optimized output style image

$\hat{y}$ x

+ |F() − F()|

optimized output content image

$$\arg\min_{\hat{y}} \mathcal{L}_{\text{style}}(\hat{y}, y) + \lambda \mathcal{L}_{\text{content}}(\hat{y}, x)$$

[Gatys et al., 2016]


A
B
C
D

Fast Neural Style Transfer

- Optimization-based method

$$\arg\min_{\hat{y}} \mathcal{L}_{\text{style}}(\hat{y}, y) + \lambda \mathcal{L}_{\text{content}}(\hat{y}, x)$$

- Feedforward network

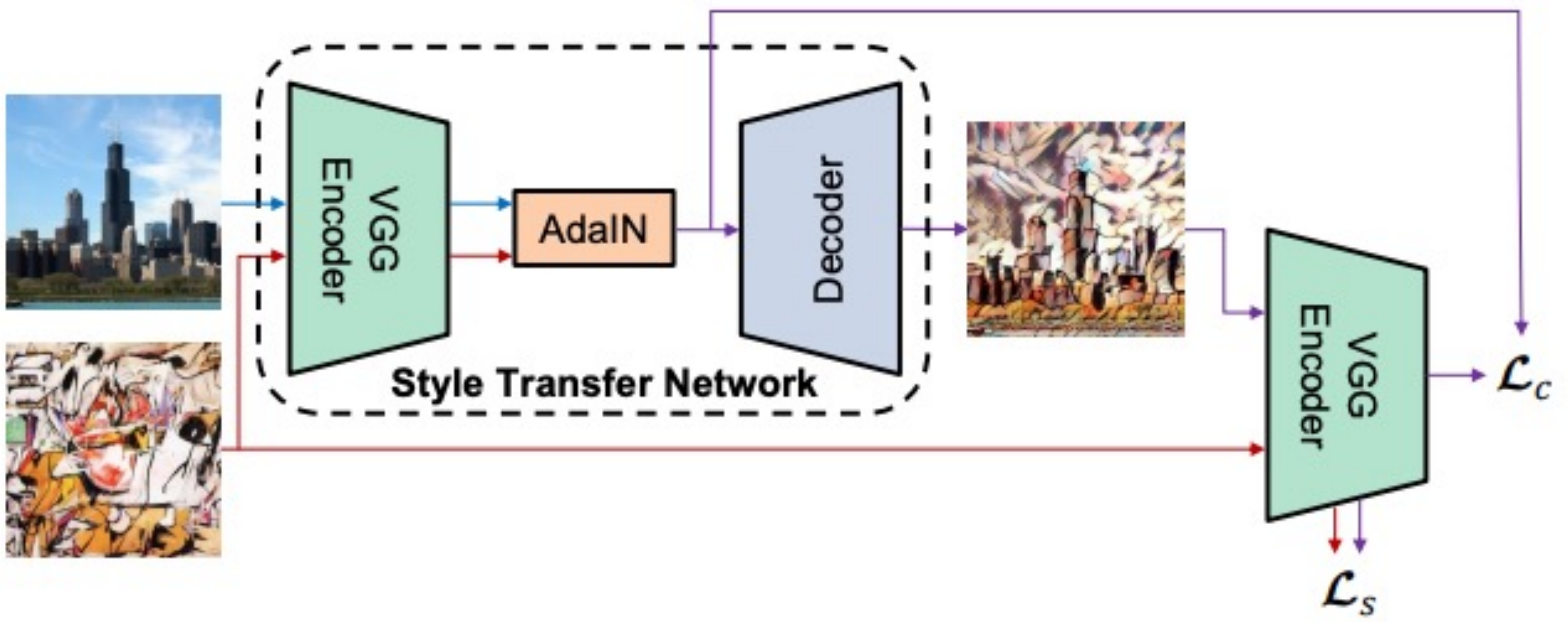
$$\arg\min_{G} \mathbb{E}_x \, \mathcal{L}_{\text{style}}(G(x), y) + \lambda \mathcal{L}_{\text{content}}(G(x), x)$$

[Johnson et al., 2016]

Arbitrary Style Transfer with AdaIN

- Feedforward network with any style

$$\arg\min_G \mathbb{E}_{x,y}\, \mathcal{L}_{\text{style}}(G(x,y),y) + \lambda \mathcal{L}_{\text{content}}(G(x,y),x)$$



[Huang et al., 2017]

Arbitrary Style Transfer with AdaIN

[Huang et al., 2017]

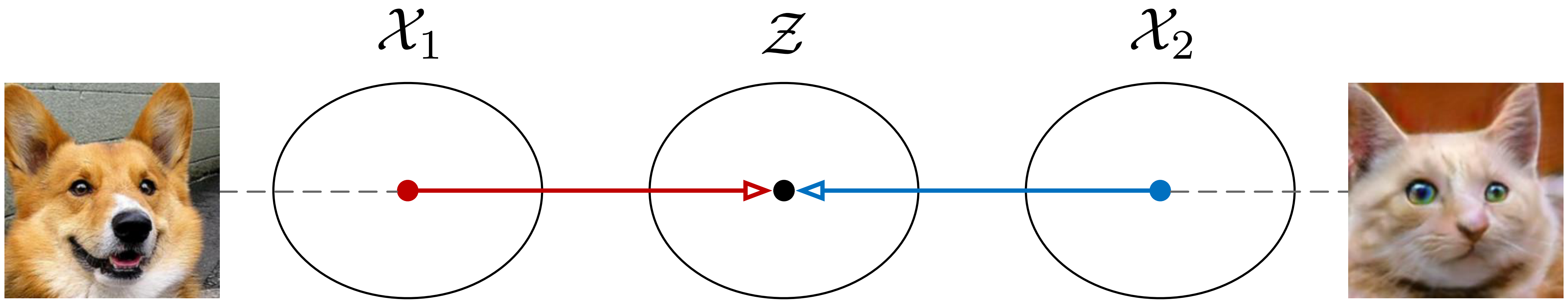
Style and Content

- <u>Style</u>: the style (color, texture, etc.) of a single painting
- <u>Content</u>: the layout and semantics of a real photo

Disentangled Latent Space

Disentangling the Latent Space

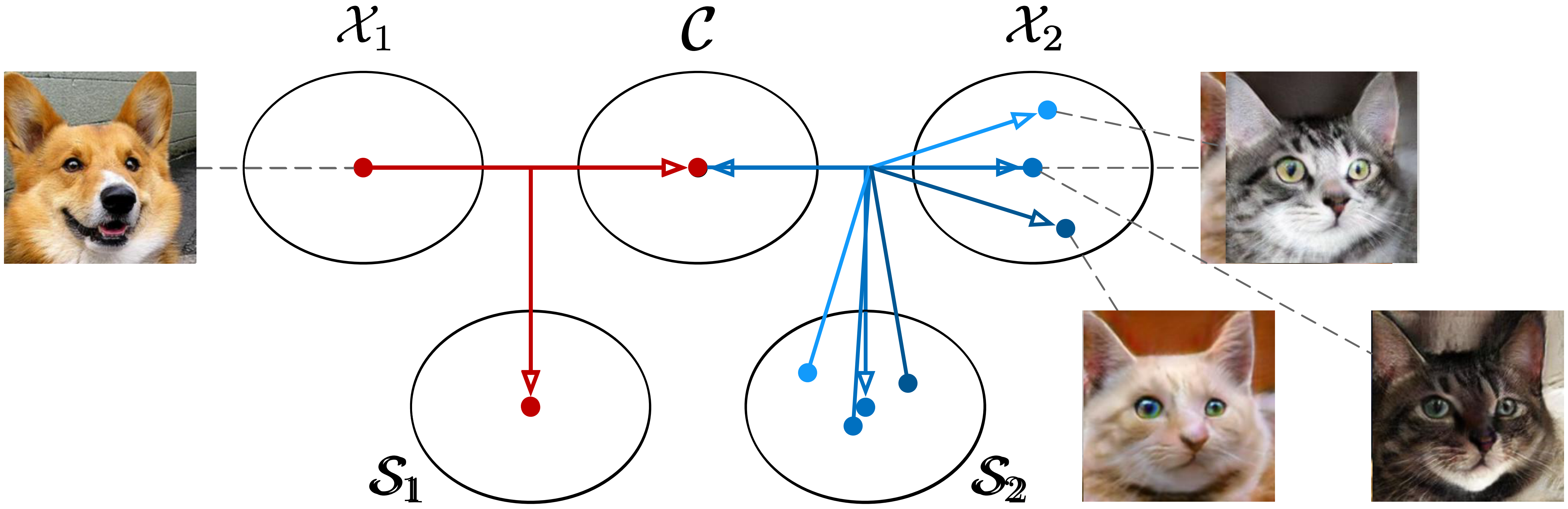
- UNIT
  - A single **shared, domain-invariant** latent space $\mathcal{Z}$



[Liu et al., 2017]

Disentangling the Latent Space

- Multimodal UNIT (MUNIT)
  - A **content** space $\mathcal{C}$ that is **shared, domain-invariant**
  - Two **style** spaces $\mathcal{S}_1, \mathcal{S}_2$ that are **unshared, domain-specific**



[Huang et al., 2018]

Style and Content

- <u>Style</u>: variations within the same domain
  (different colors, textures, etc.)
- <u>Content</u>: features shared across two domains

Church images

Are  and  from the same image?

Swapping Autoencoder [Park et al., 2020]

Answer:
No

Are  and  from the same image?

Swapping Autoencoder [Park et al., 2020]

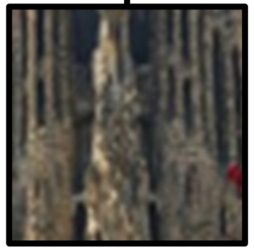
 

Answer:
Yes

Are  and  from the same image?

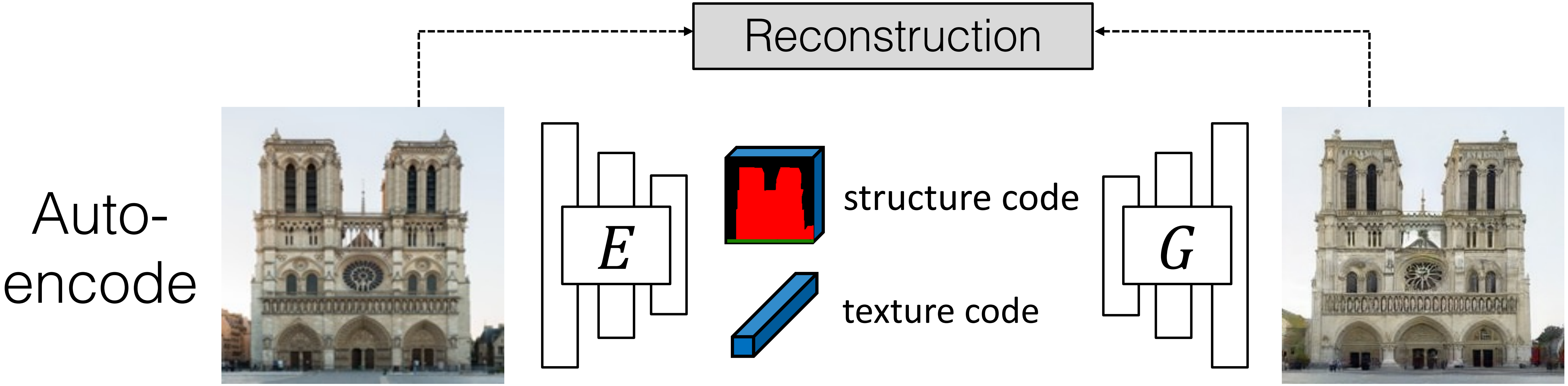
Swapping Autoencoder [Park et al., 2020]

Answer:
…?

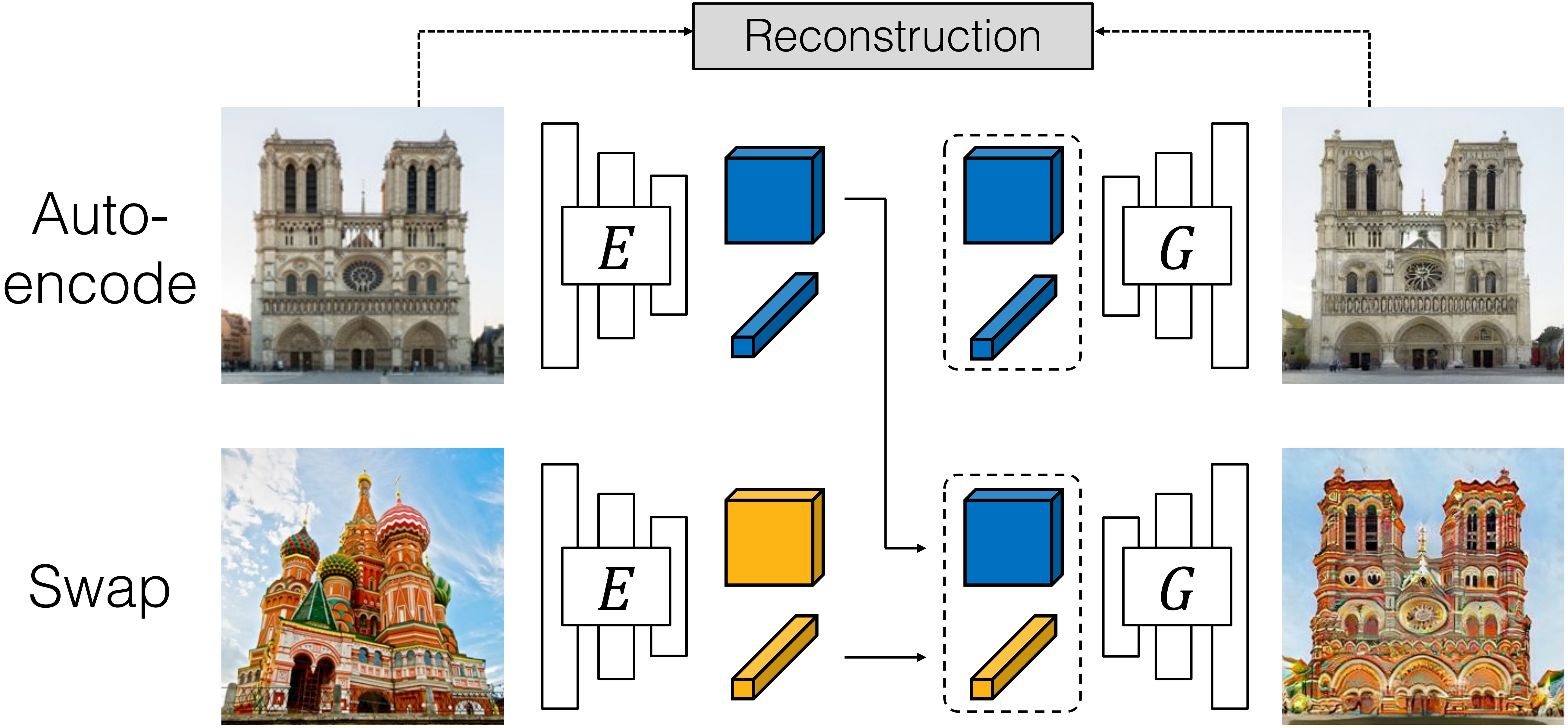
Are  and  from the same image?

Patch co-occurrence discriminator

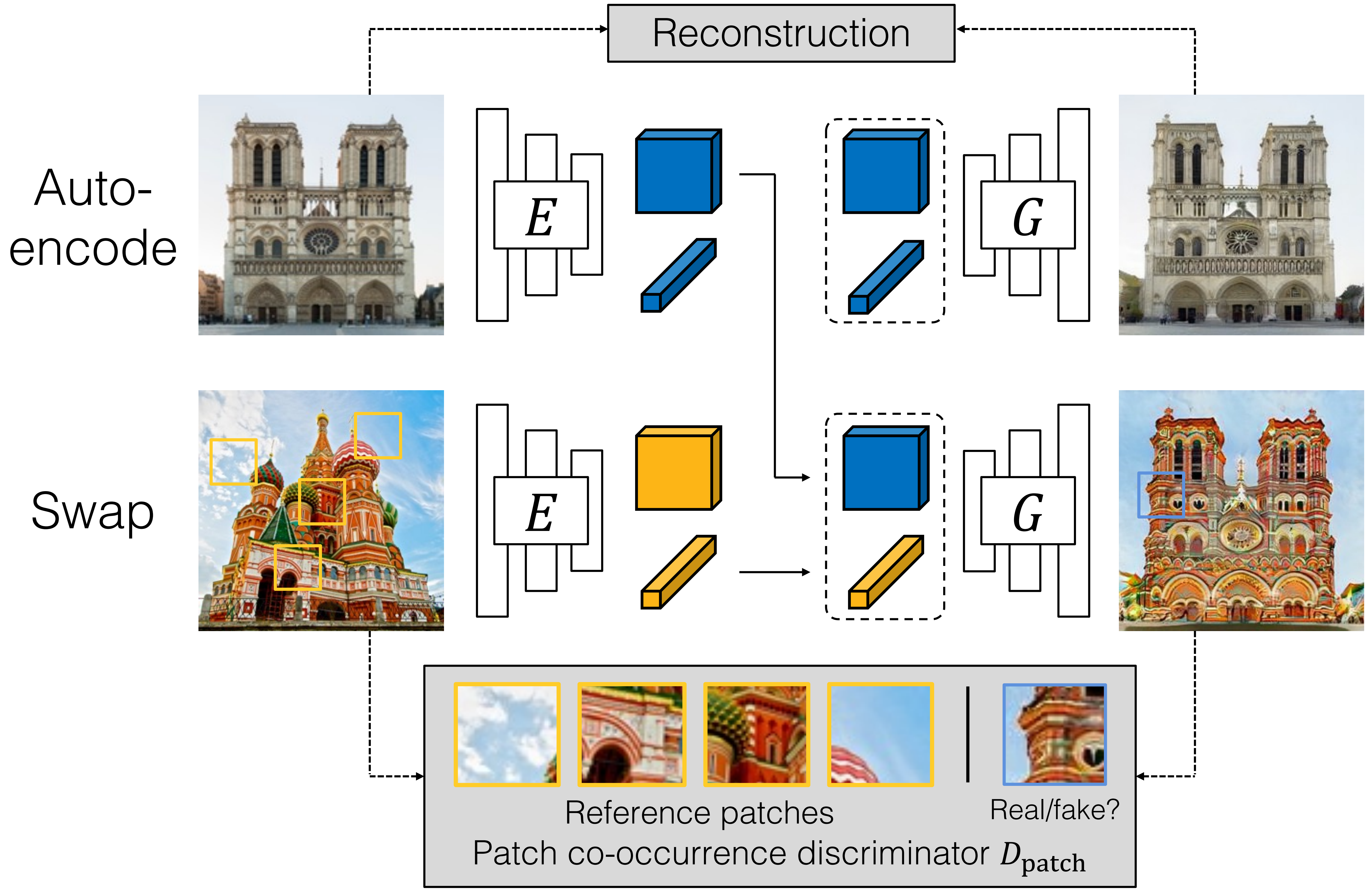
Swapping Autoencoder [Park et al., 2020]



Swapping Autoencoder [Park et al., 2020]



Swapping Autoencoder [Park et al., 2020]



Reconstruction
Auto-
encode
Swap
E
G
E
G
Reference patches
Real/fake?
Patch co-occurrence discriminator D_{patch}


texture
structure

Style and Content

- Style: variations within the same domain

  (different colors, textures, etc.)
- Content: the layout and semantics of a single photo

Landscape Mixer Demo

Photo credit: Aaron Nace

Neural Style Transfer
vs.
Image-to-Image Translation


Input
Style Image I
Style image II
Entire collection
CycleGAN
Photo → Van Gogh
Input
Style image I
Style image II
Entire collection
CycleGAN
horse → zebra

Photo Style Transfer

Deep Photo Style Transfer

(a) Reference style image (b) Input image (c) Neural Style (distortions) (d) Our result

Local color transfer? (hard to transfer texture)

[Luan et al., 2017]

Make  look like 

Make  look like 

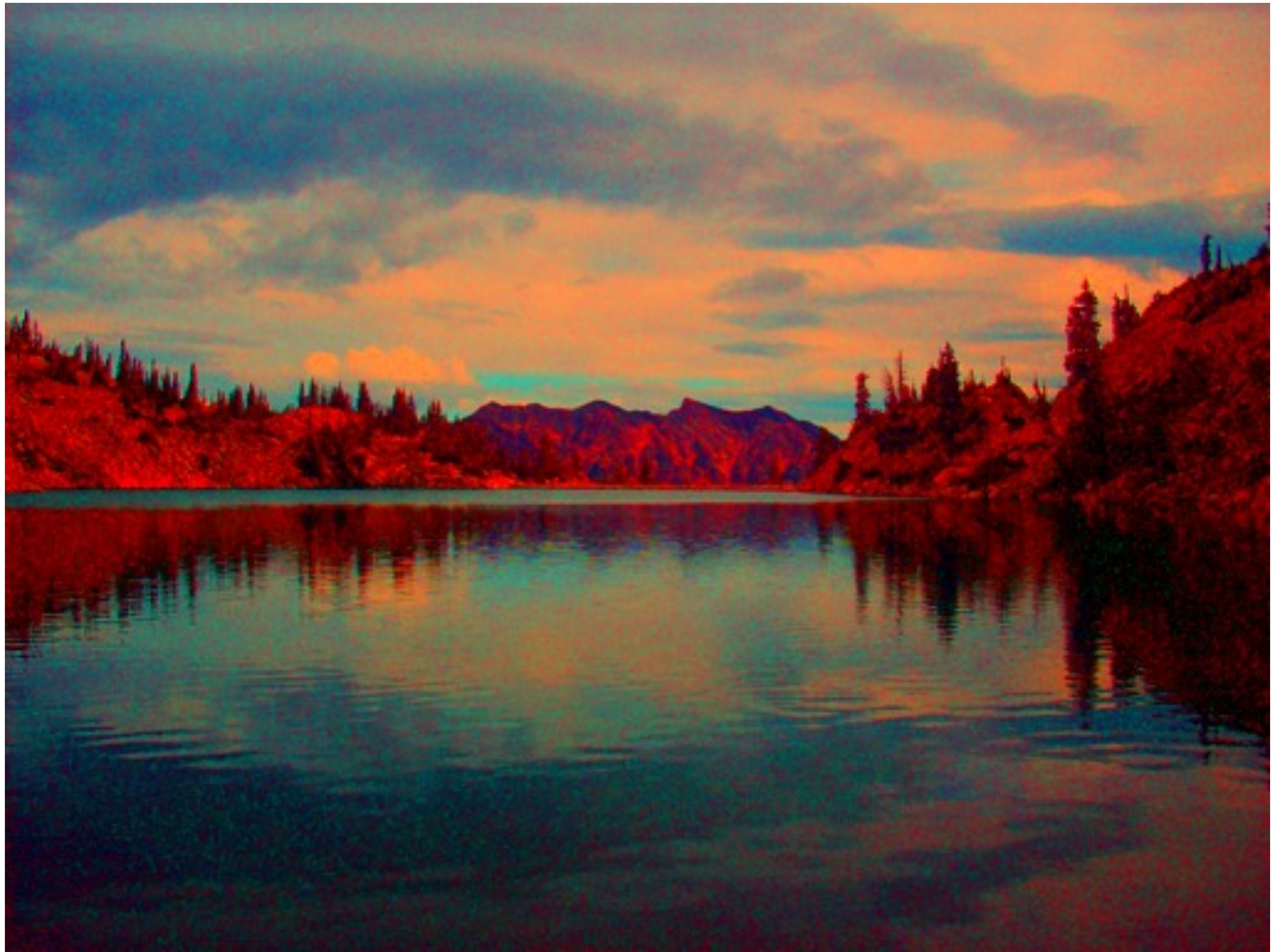
Histogram Matching

Make  look like 

Reinhard et al. [2001]

Make  look like 

Pitie et al.
[2005]



Make  look like 

Photoshop Match Color

Make  look like 

Gatys et al. [2016]

Make  look like 

Our method 

Target

Target

Motivation

The neural style algorithm…

- Works well for **Paintings**!

- What about **Photos**?

Motivation

- So we tried it on photos:

Style

Input

Result

Fixing Distortion

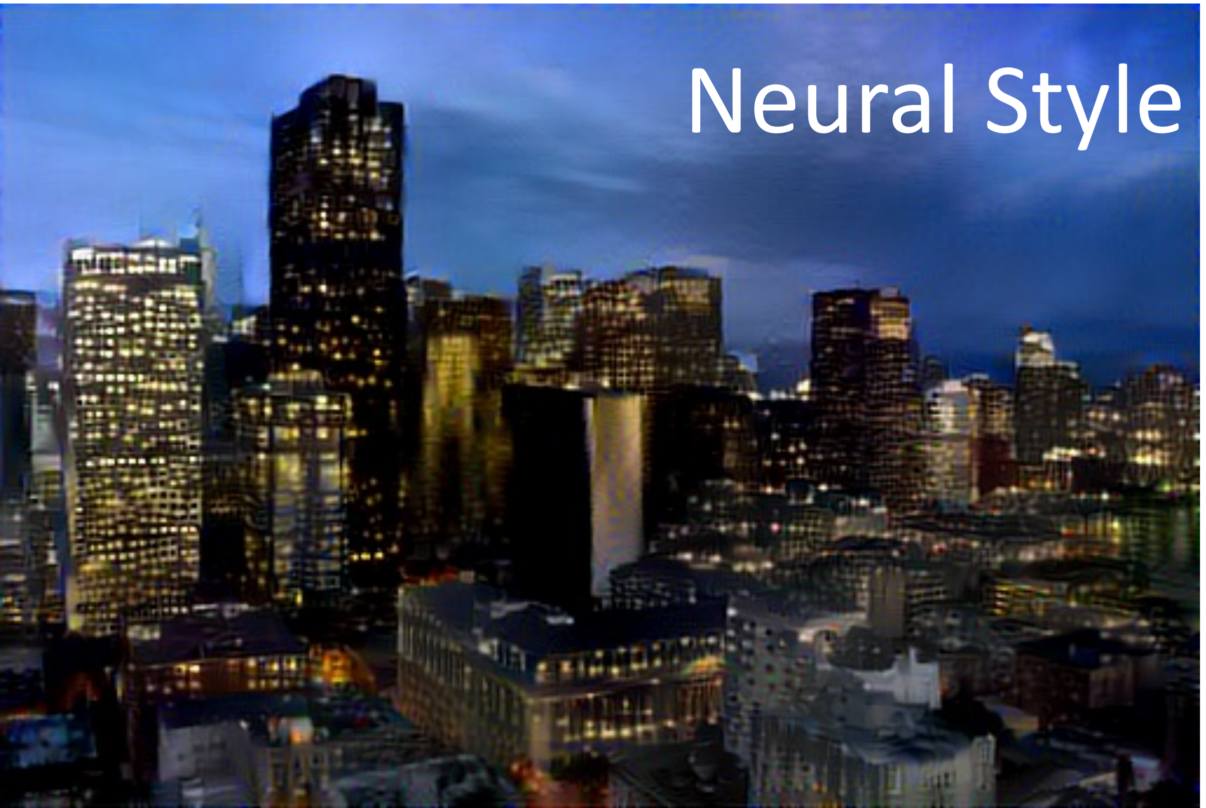
Local affine color transform for each patch [Levin et al. 2006]

$$\begin{pmatrix} r_{out} \\ g_{out} \\ b_{out} \end{pmatrix} = A_{3\times3} \begin{pmatrix} r_{in} \\ g_{in} \\ b_{in} \end{pmatrix} + B_{3\times1}$$

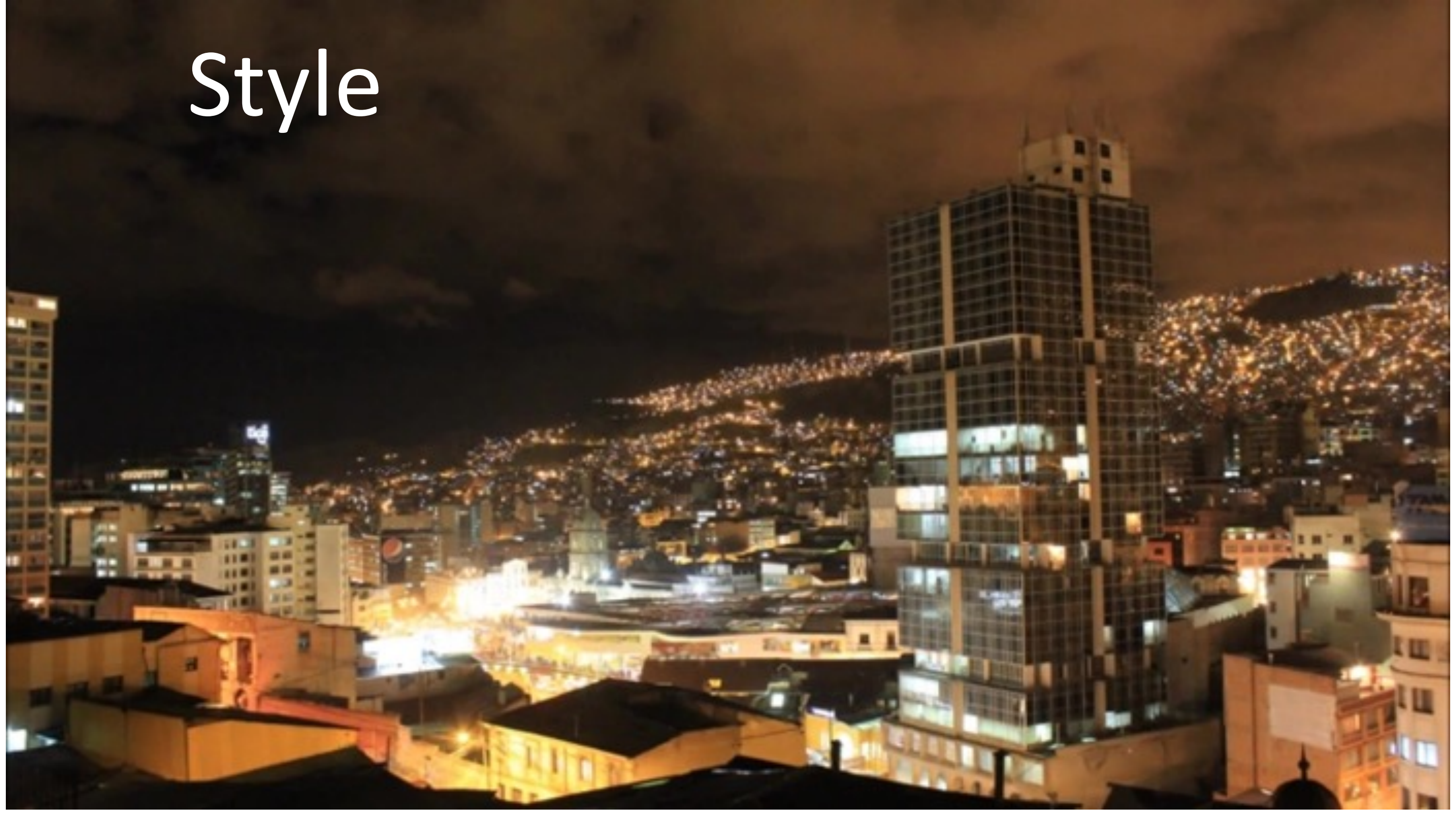
See more technical details on Wednesday's paper presentation


Input


Style


Neural Style

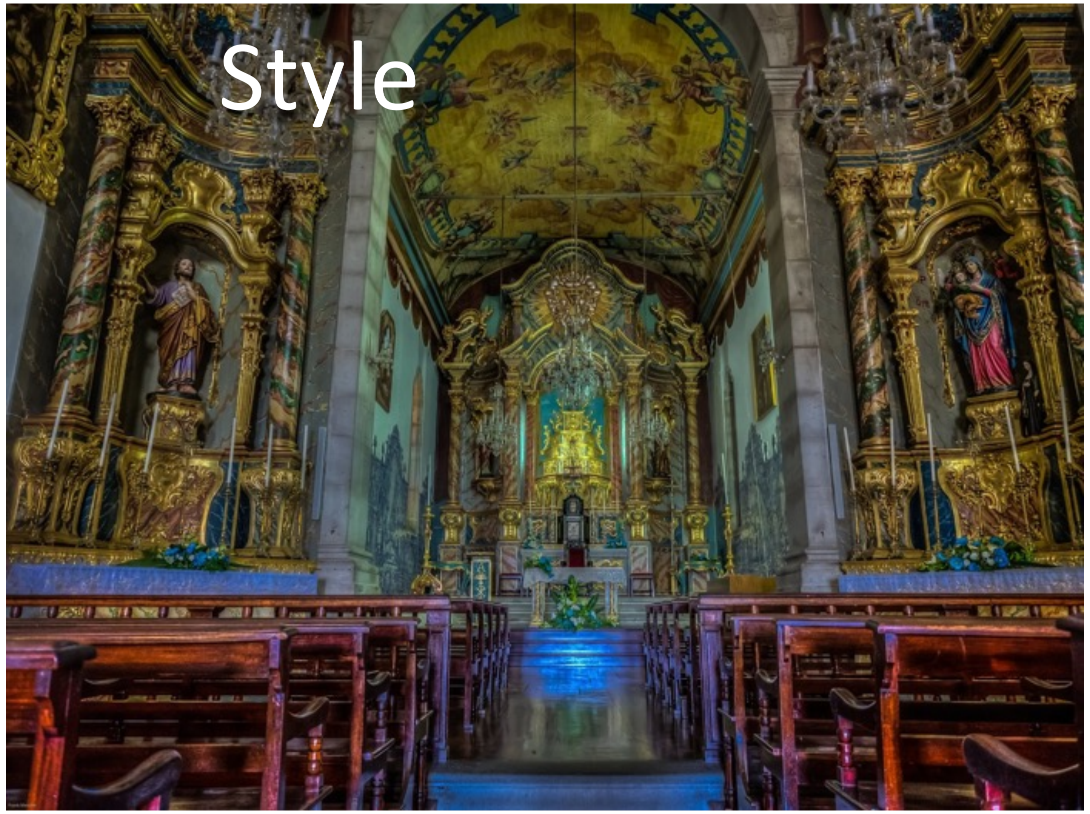

Ours


Input

Style

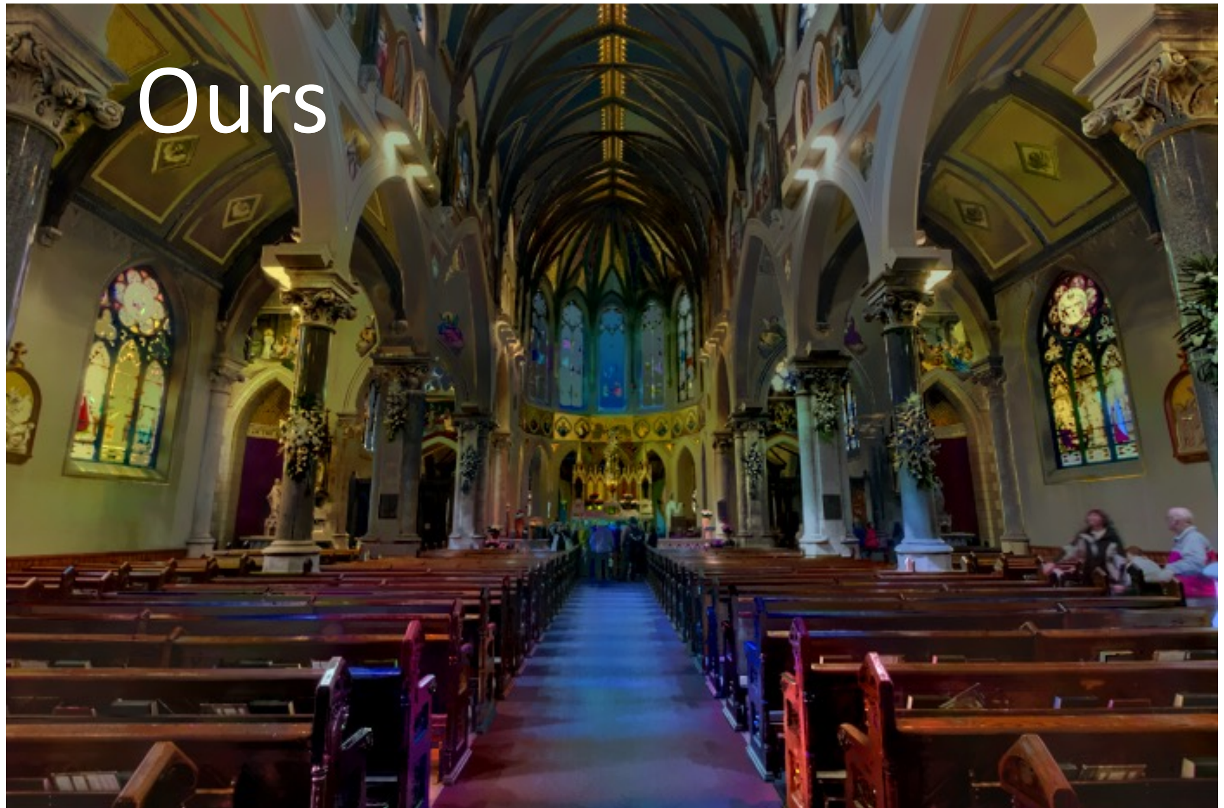
Neural Style

Ours


Input


Style


Neural Style

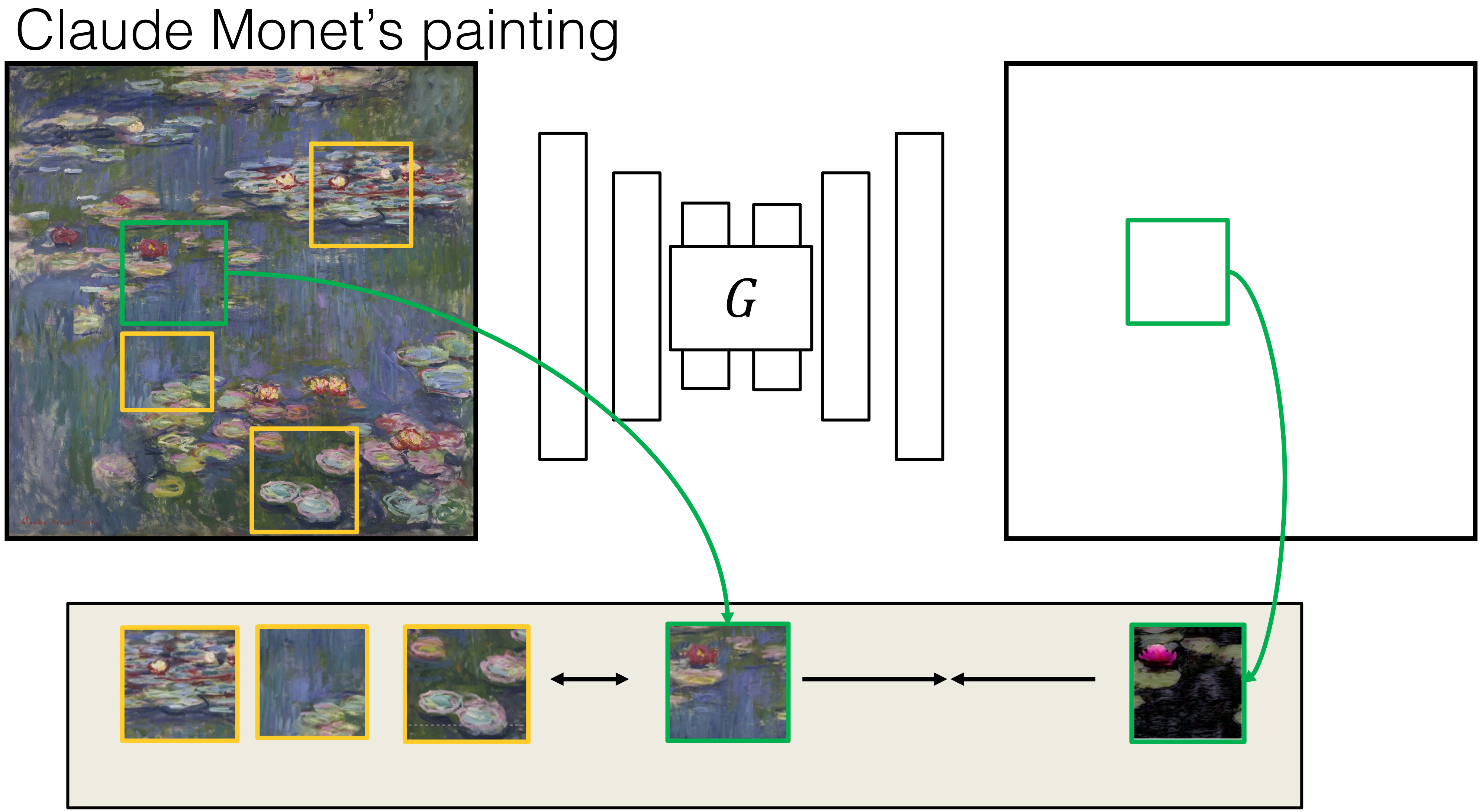

Ours

Single Image Translation

Domain = {patches of a single image}

Single Image Translation

[Park et al., 2020]

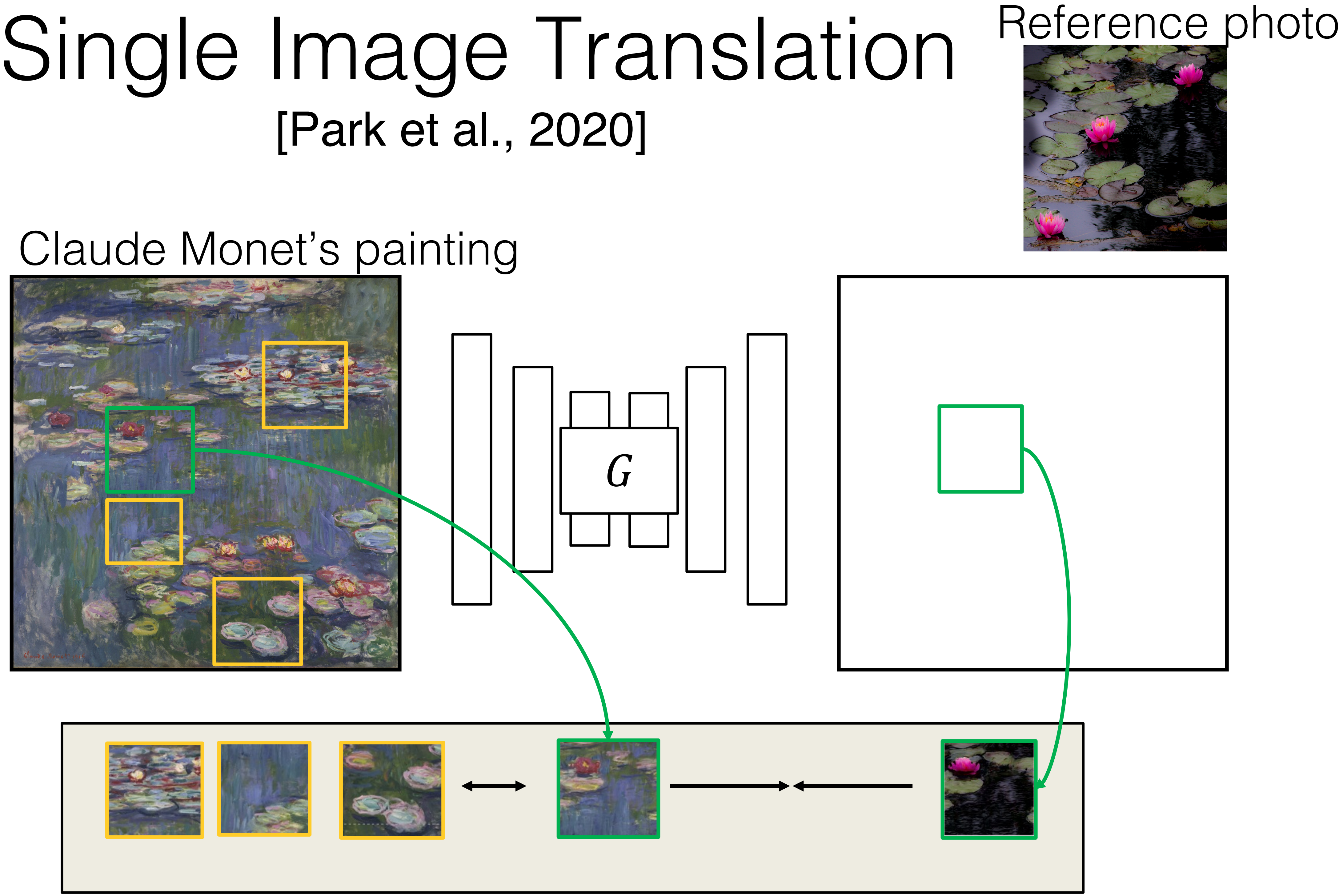


Internal contrastive loss is well-suited for single image translation.

Also see InGAN (Shocher et al., ICCV'19), SinGAN (Shaham et al., ICCV'19)

Single Image Translation

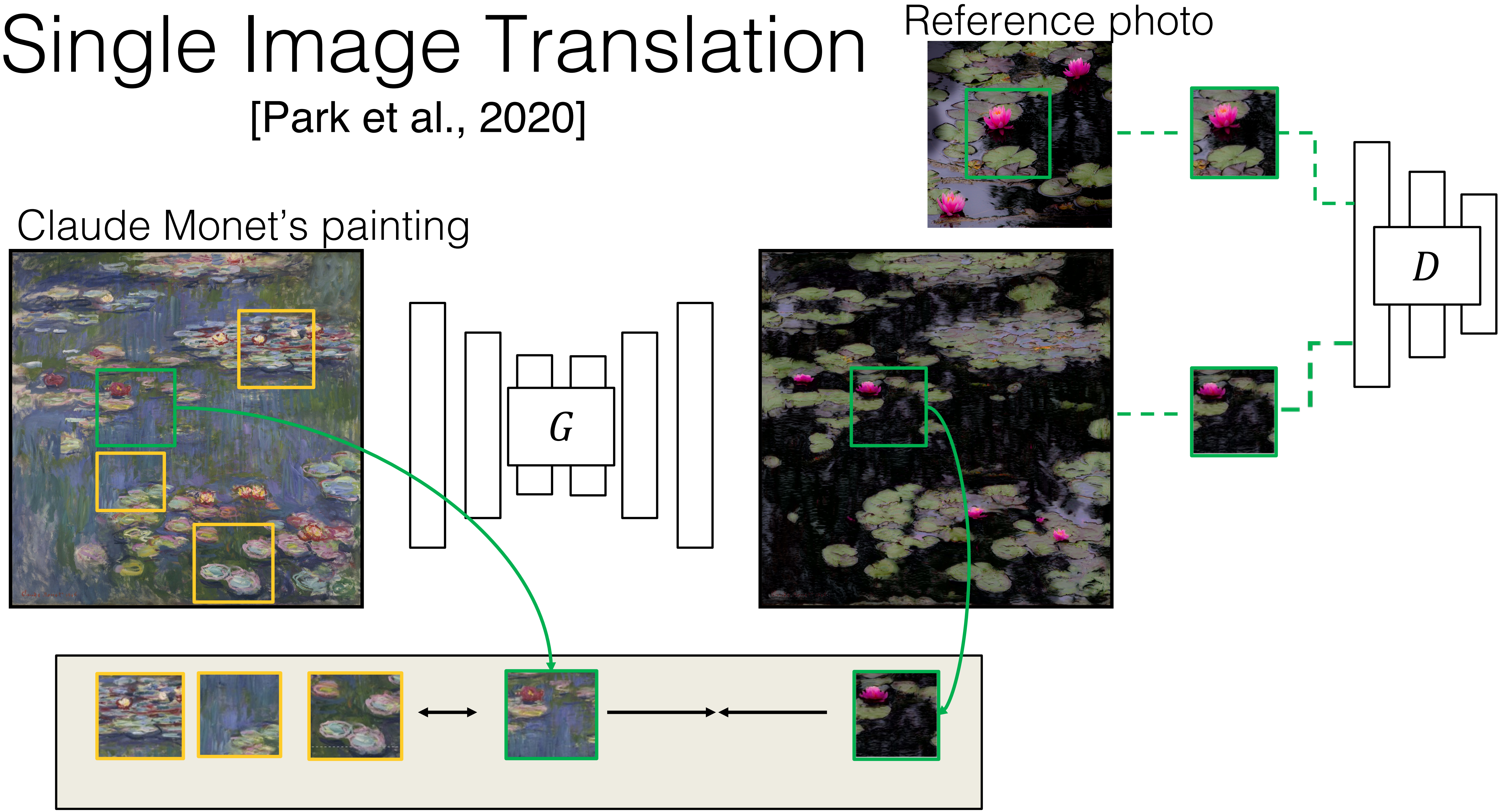
[Park et al., 2020]



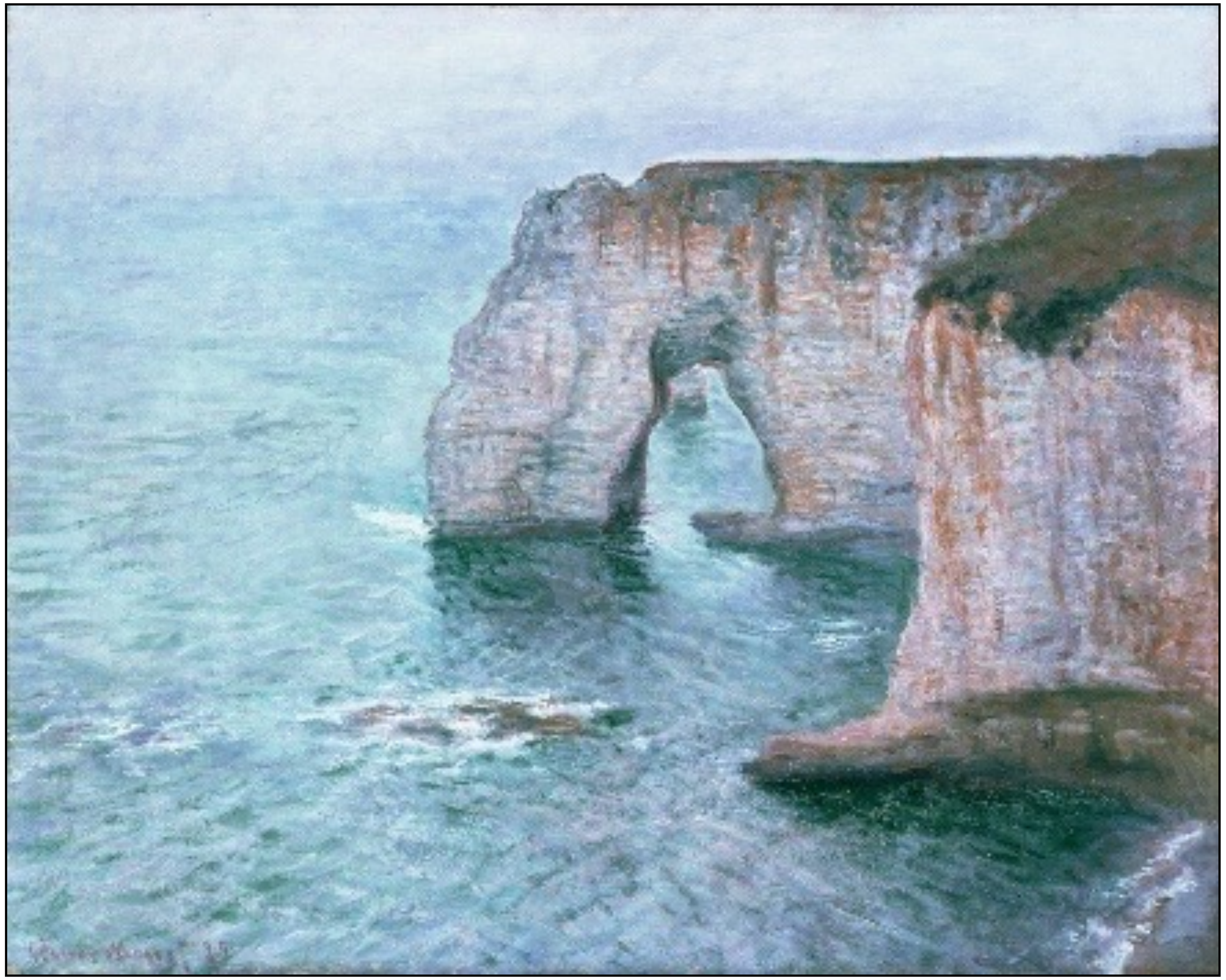
Internal contrastive loss is well-suited for single image translation.
Also see InGAN (Shocher et al., ICCV'19), SinGAN (Shaham et al., ICCV'19)

Single Image Translation

[Park et al., 2020]



Internal contrastive loss is well-suited for single image translation.
Also see InGAN (Shocher et al., ICCV'19), SinGAN (Shaham et al., ICCV'19)



Painting

Reference

Painting

Reference

Painting

Gatys et al. CVPR'16

Reference

Painting

STROTSS (Kolkin et al., CVPR'19)

Deep Image Analogy's extension

Reference

Painting

WCT^2 (Yoo et al., ICCV'19)

Photo style transfer's extension

Reference

Painting

Our translation result

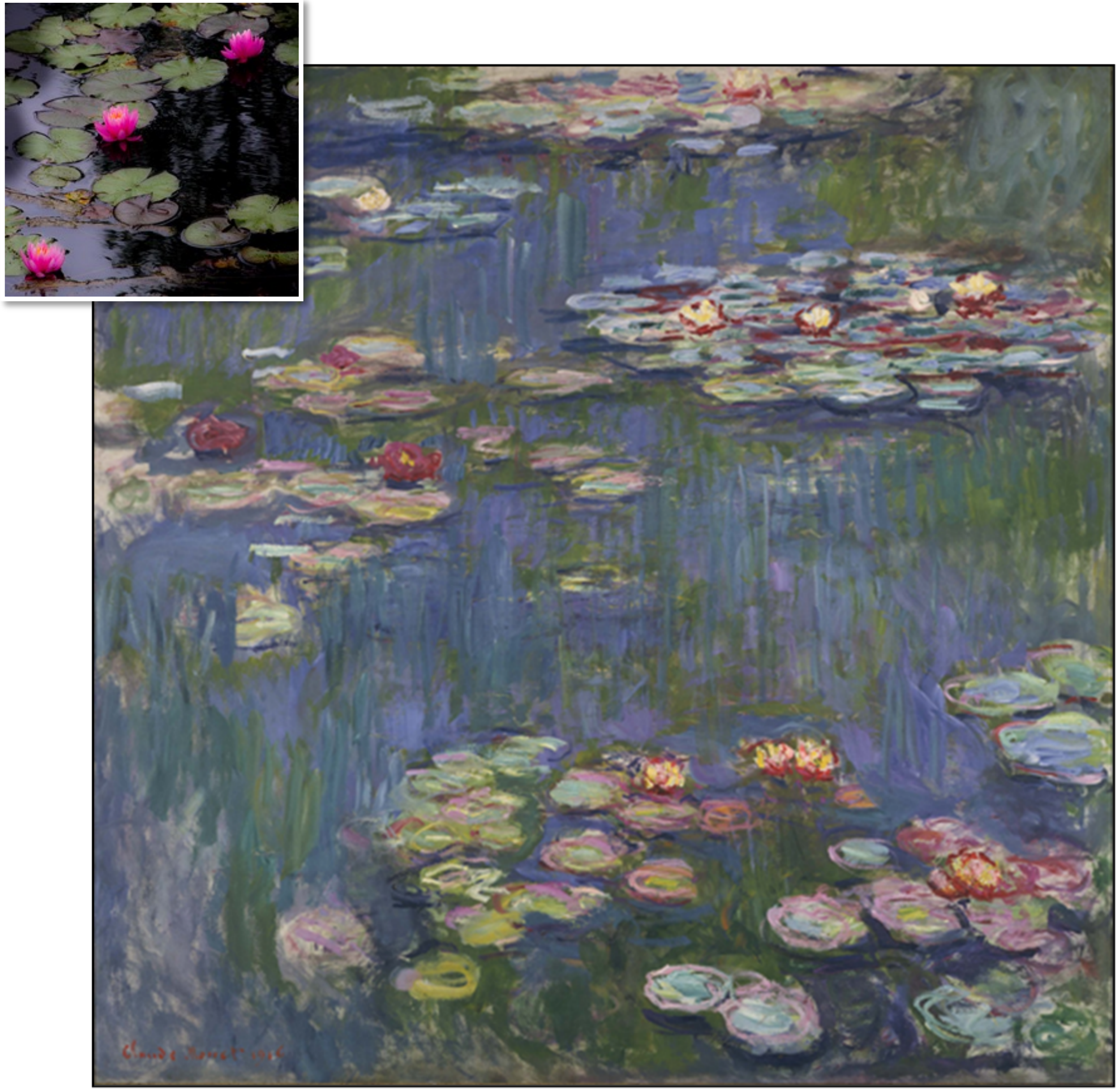
Reference

Painting

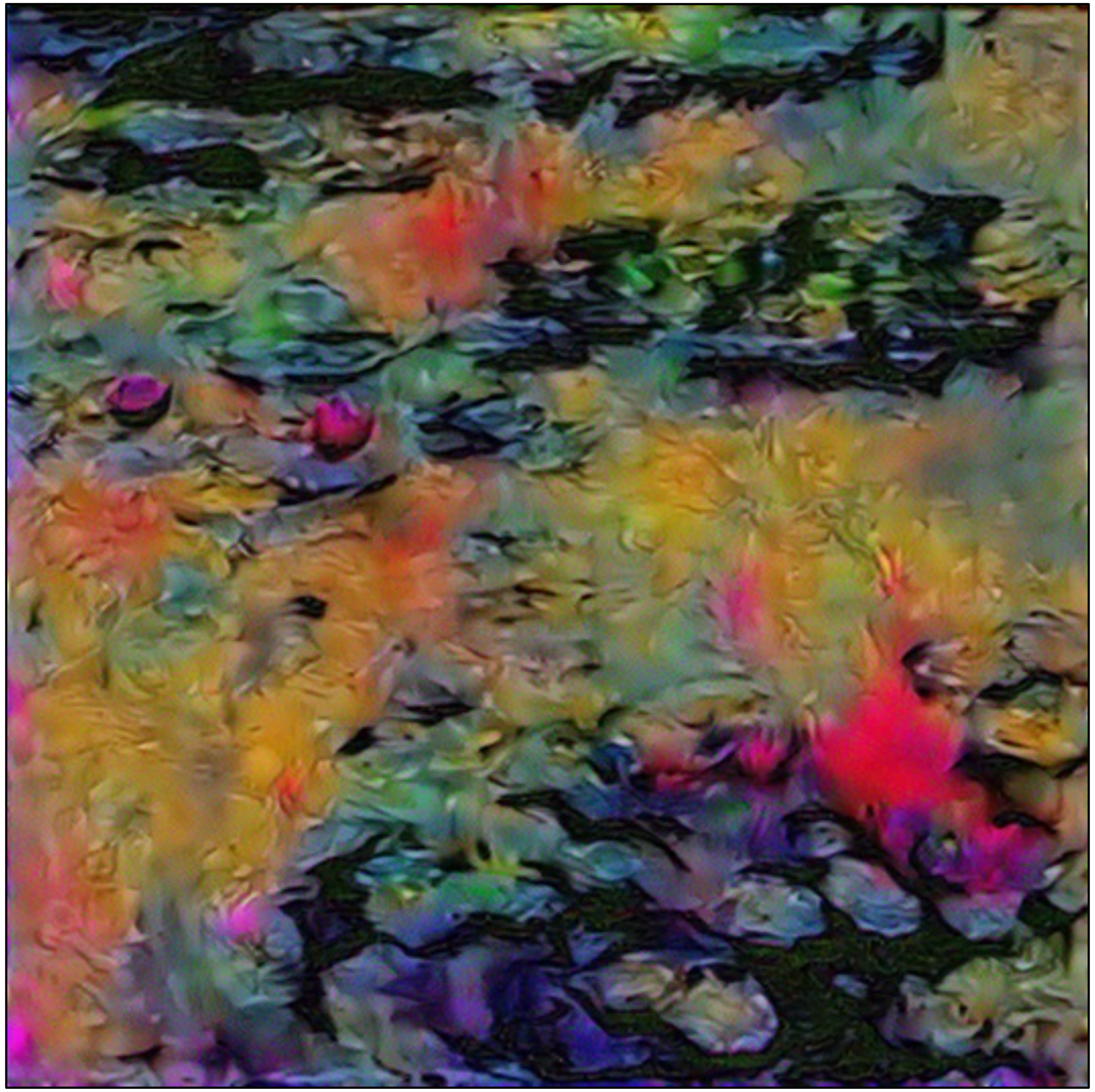
CycleGAN

Painting

Reference

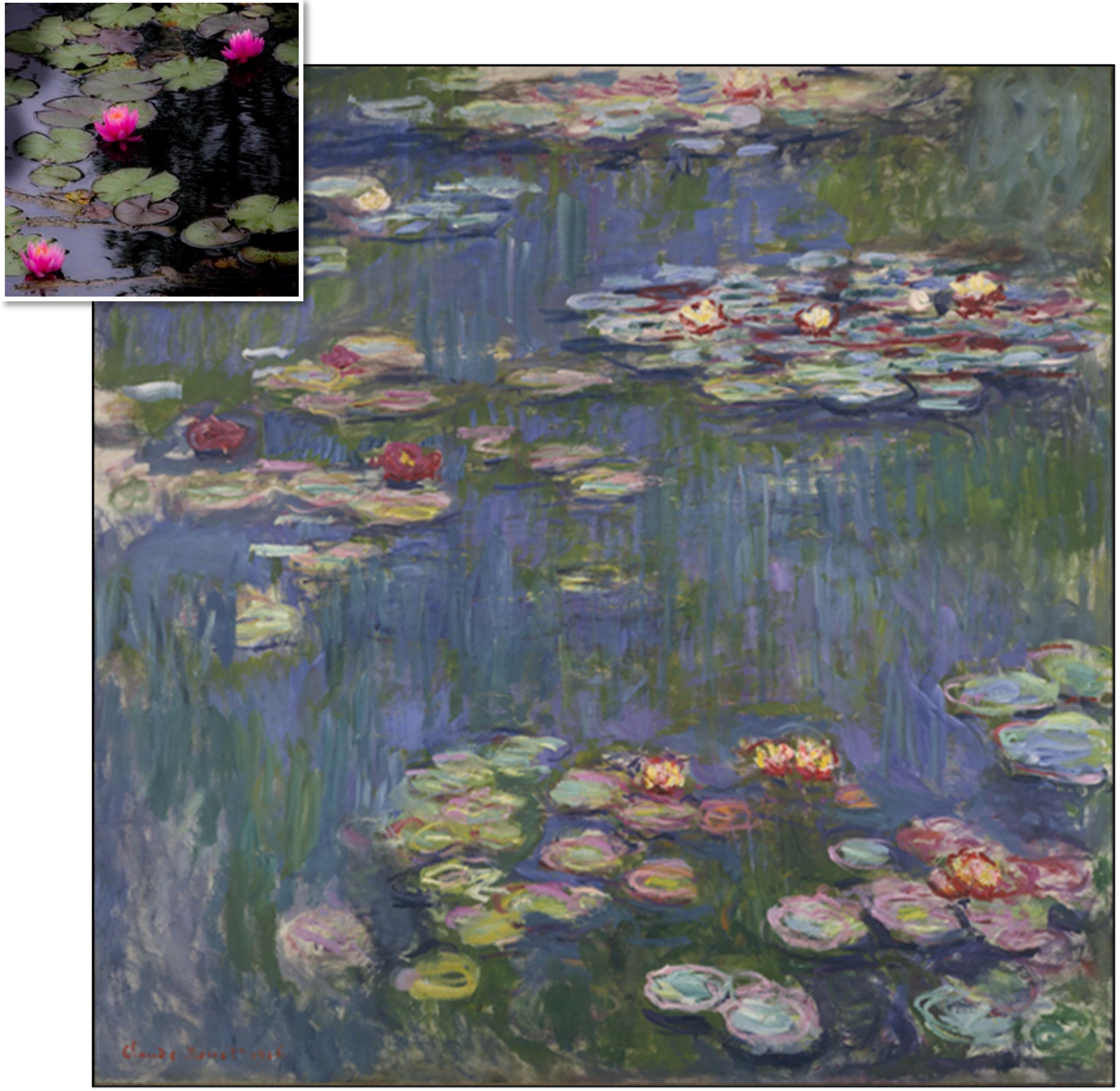


Painting

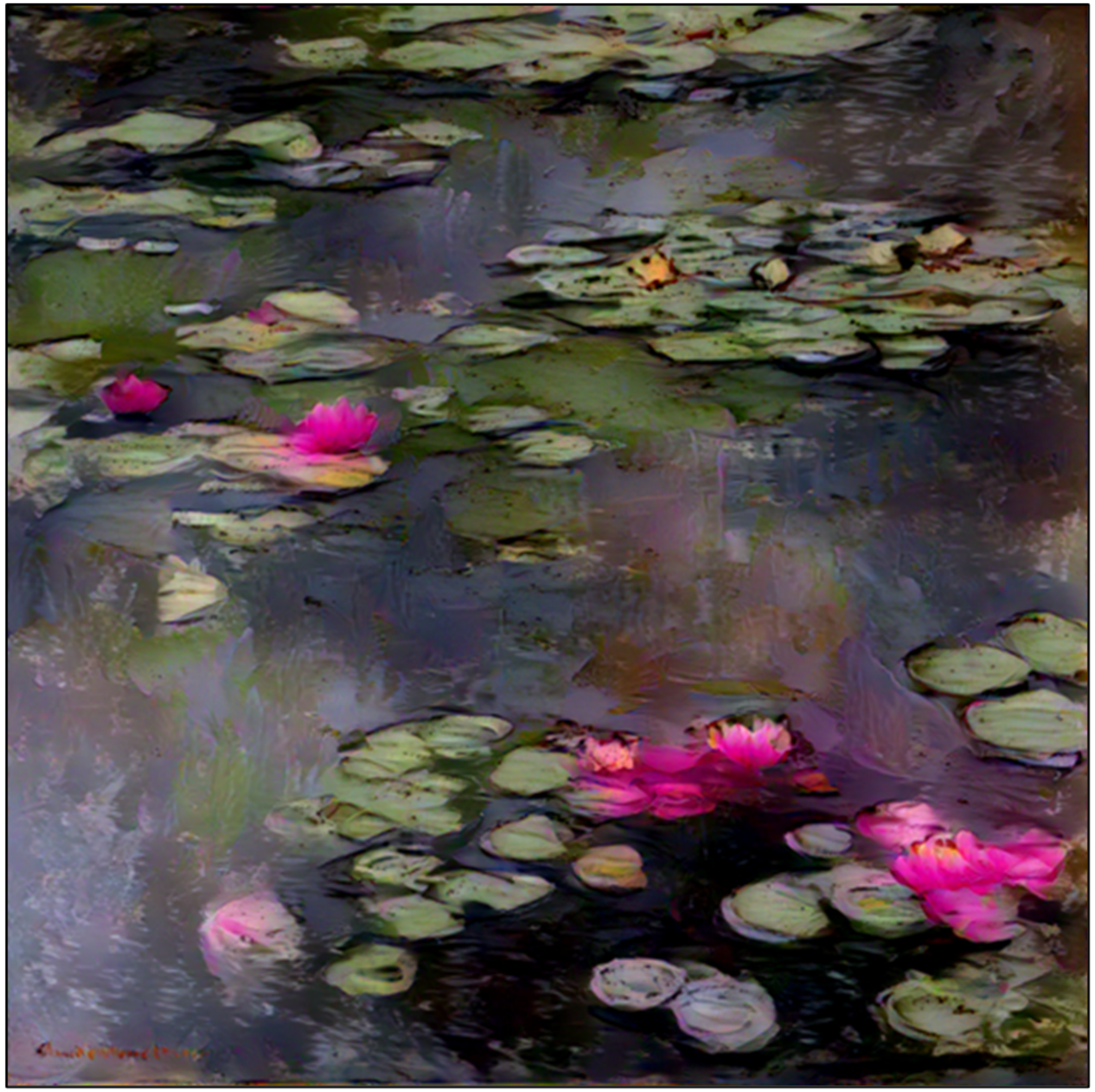


Gatys et al. CVPR’16

Reference

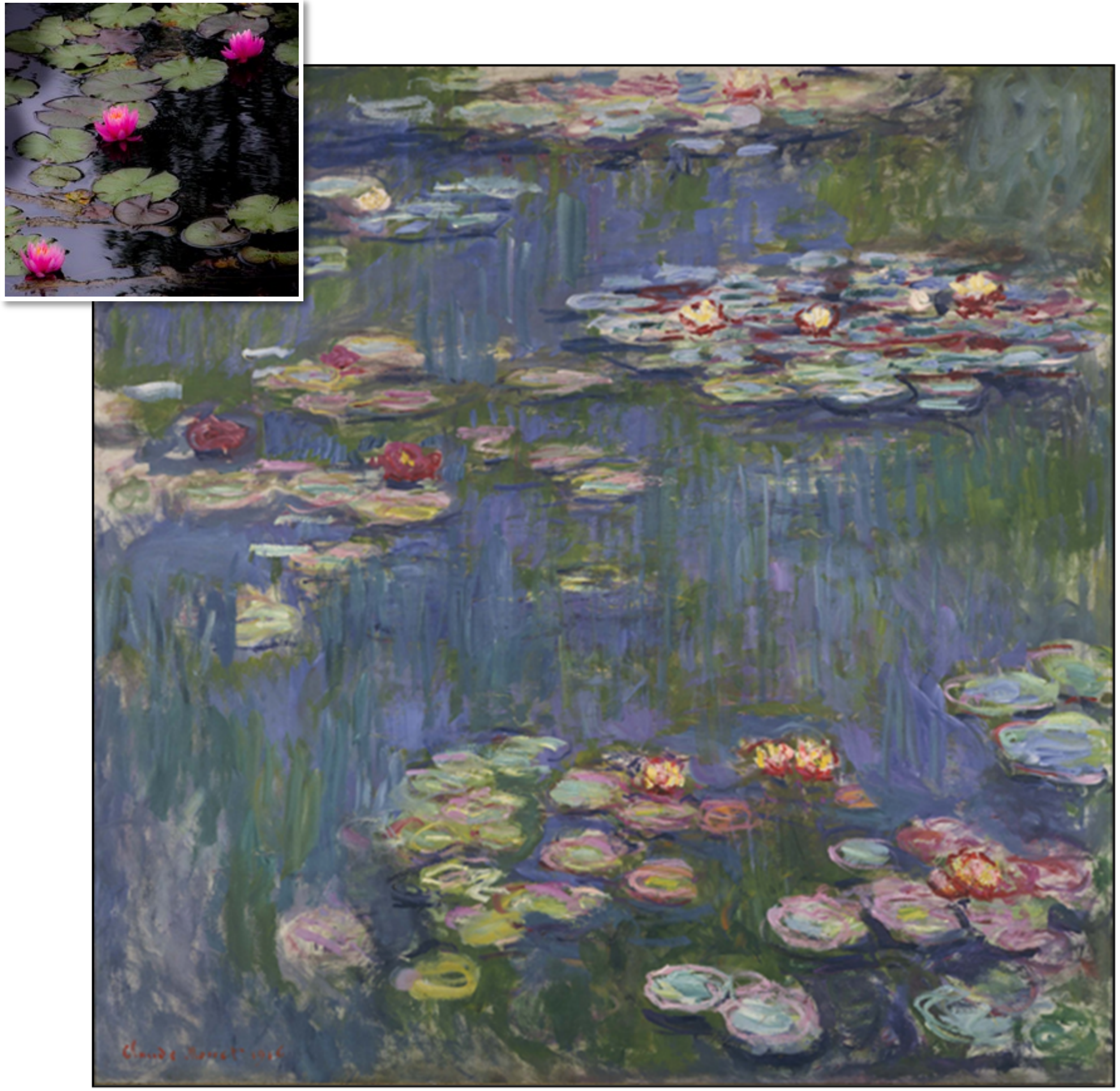


Painting

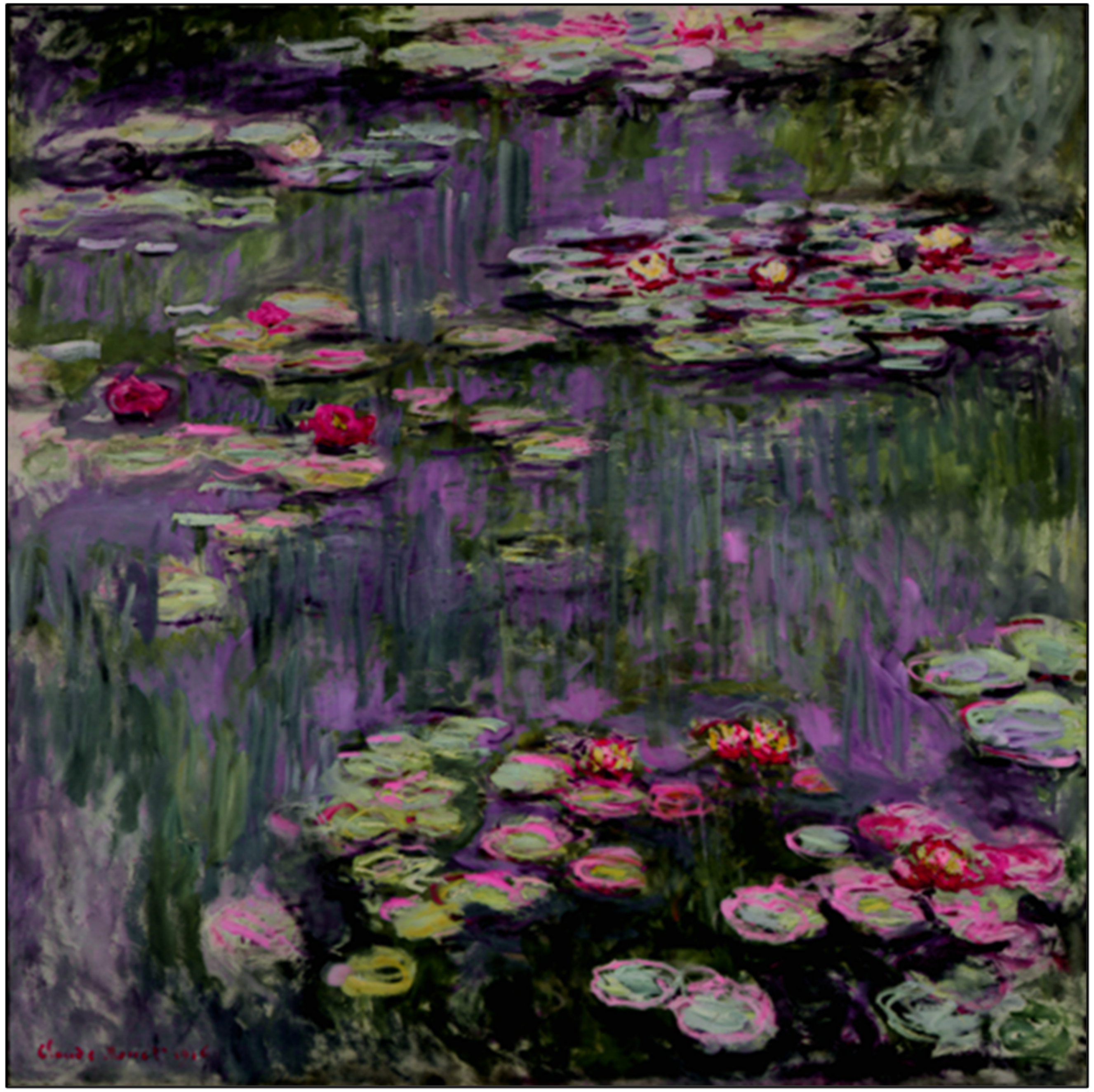


STROTSS (Kolkin et al., CVPR'19)

Reference

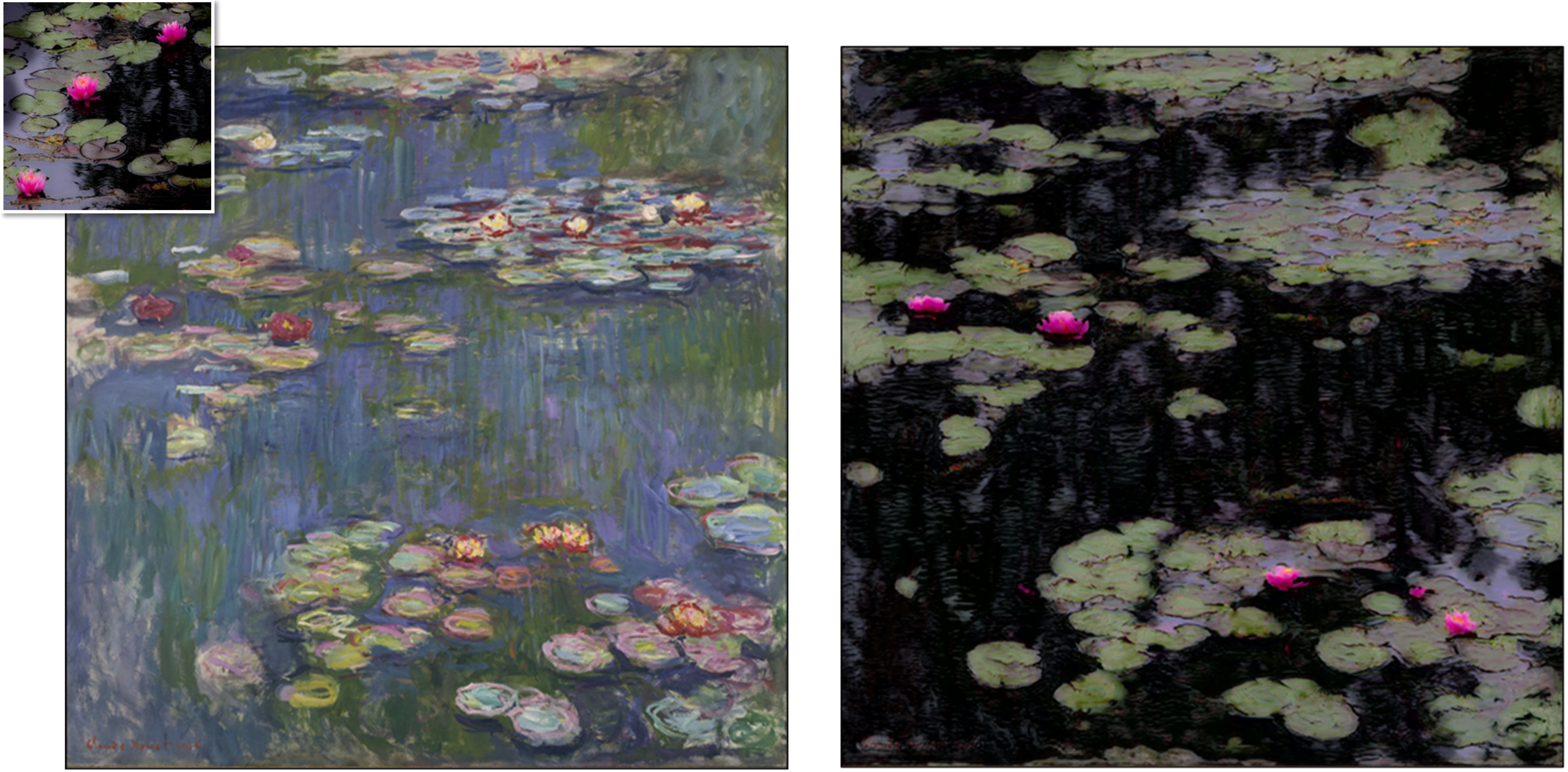


Painting



WCT^2 (Yoo et al., ICCV'19)

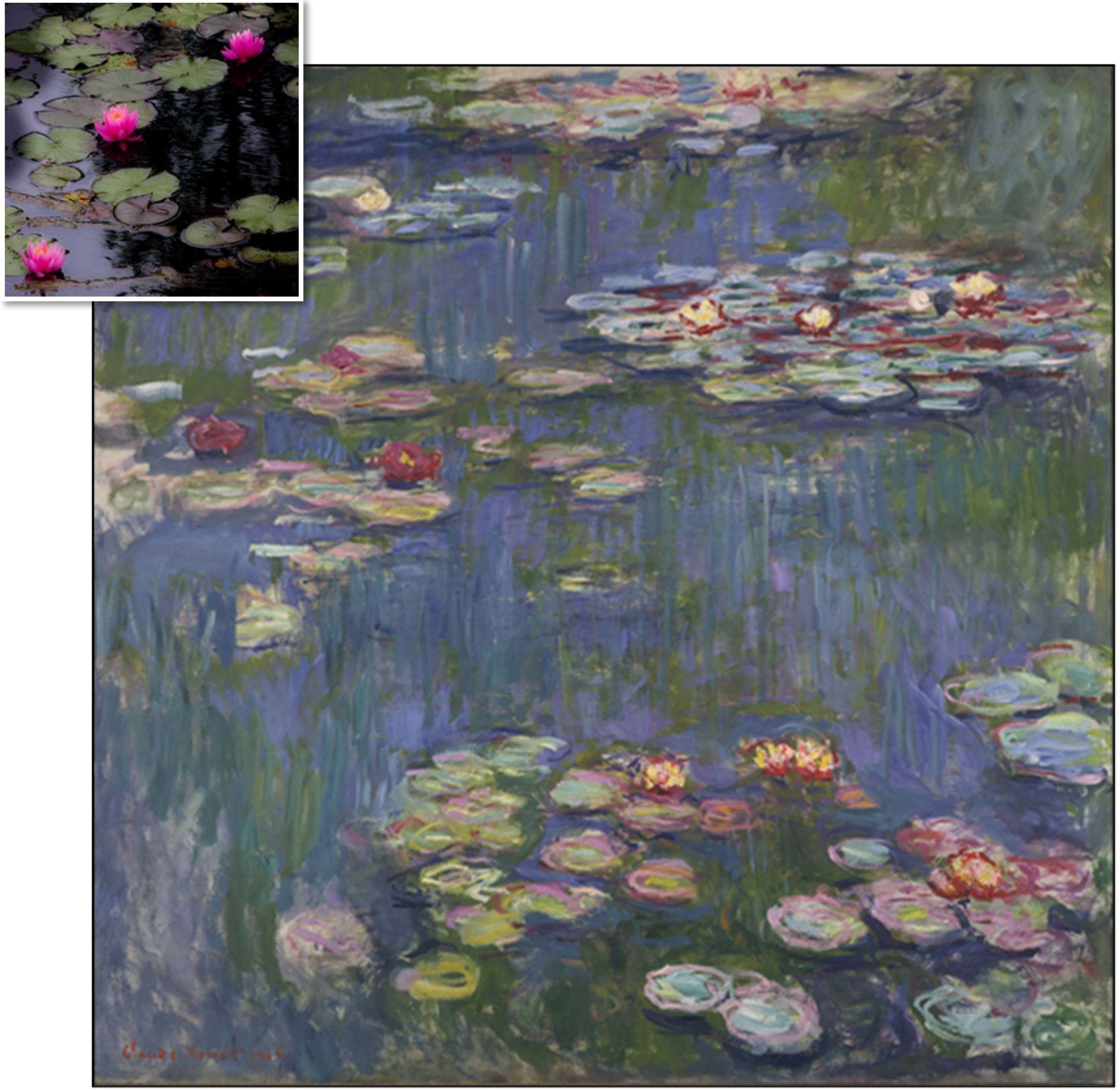
Reference



Painting

Ours

Reference

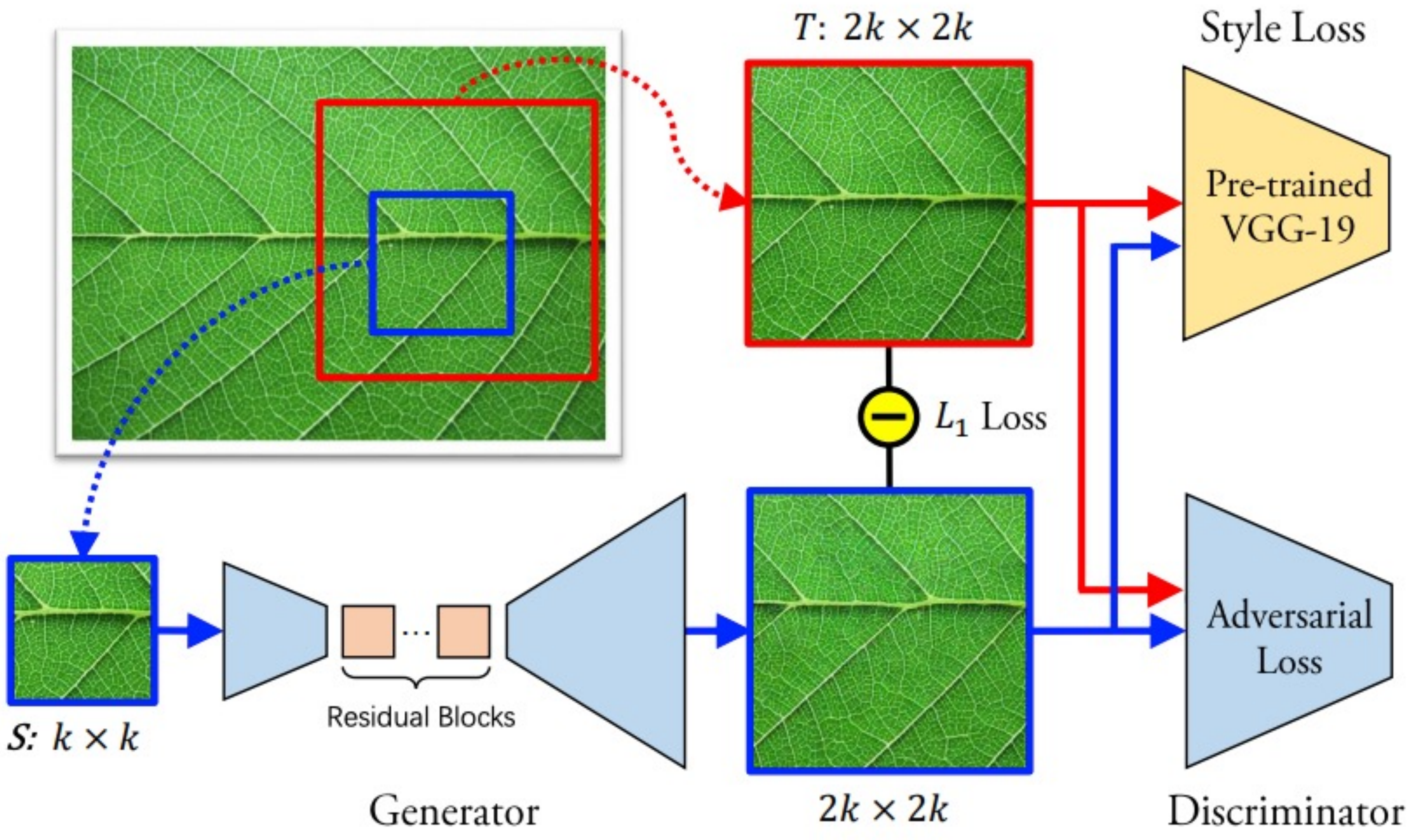


Painting

CycleGAN

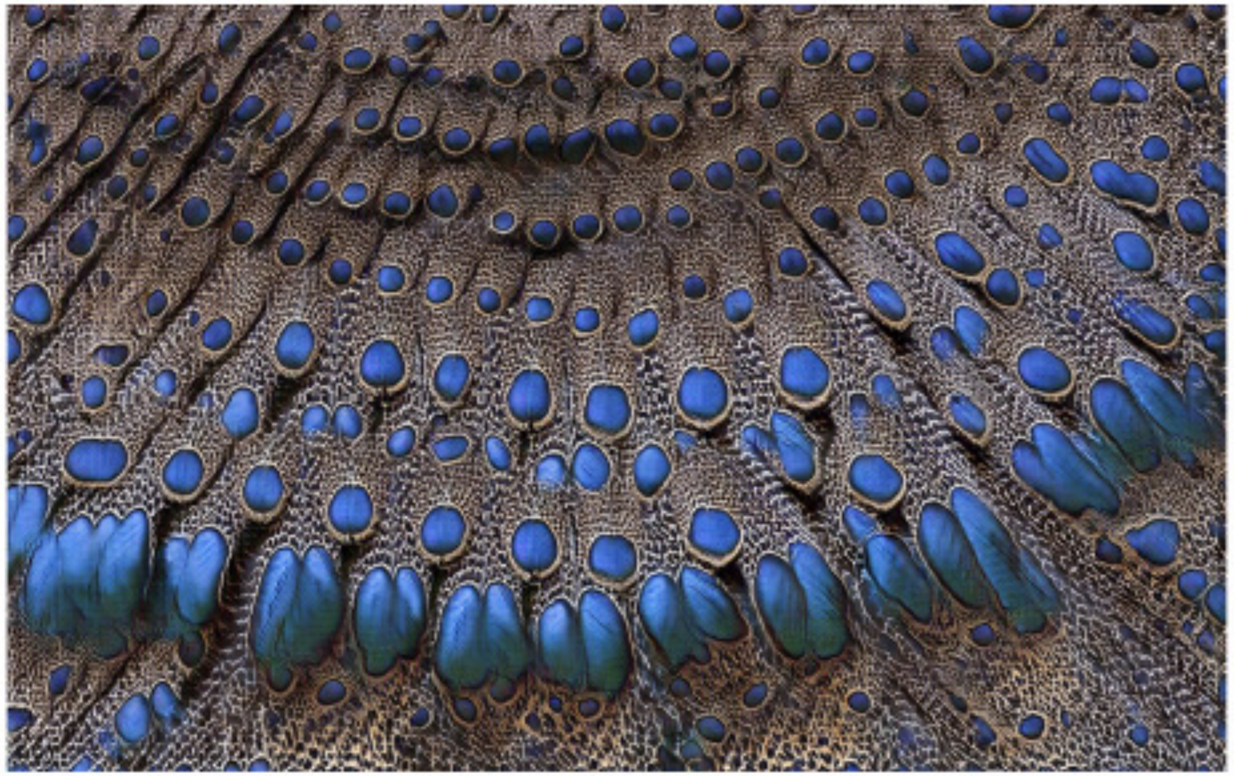
Image-to-Image Translation for Texture Synthesis

Texture Synthesis by Conditional GANs



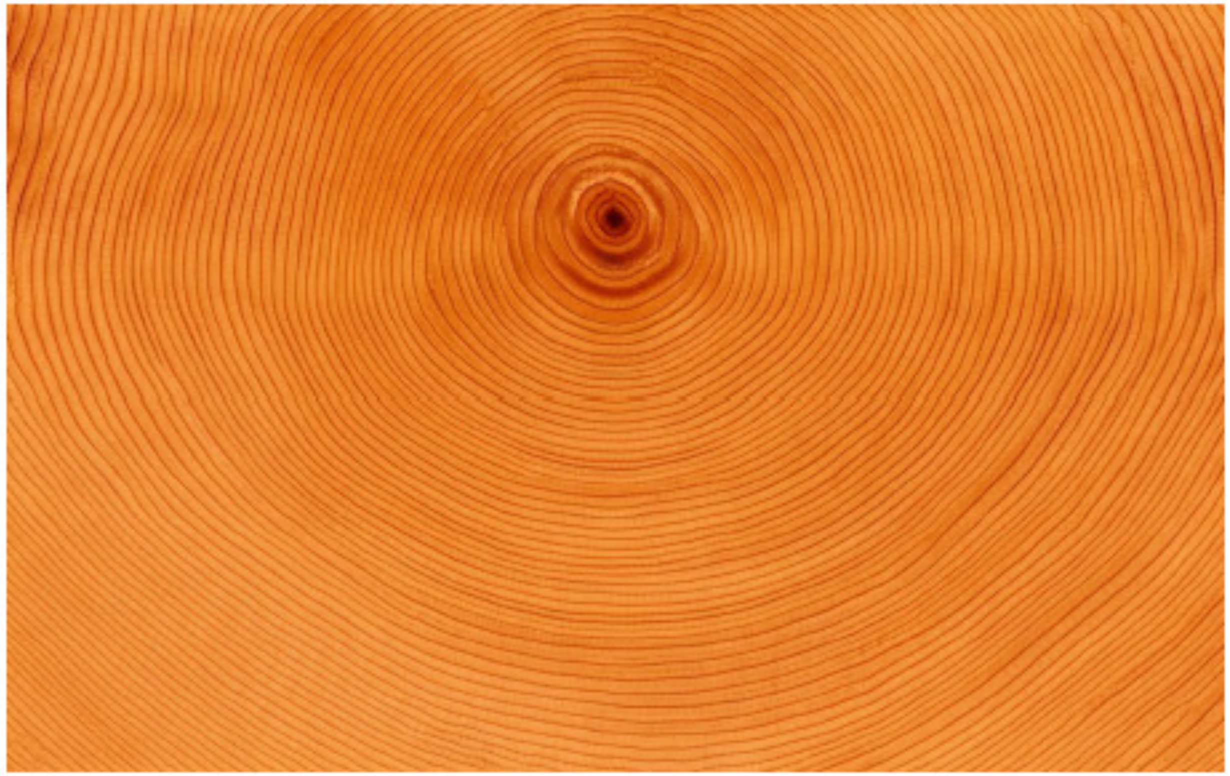
Non-stationary Texture Synthesis by Adversarial Expansion. Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, Hui Huang. SIGGRAPH 2018.

Texture Synthesis by Conditional GANs



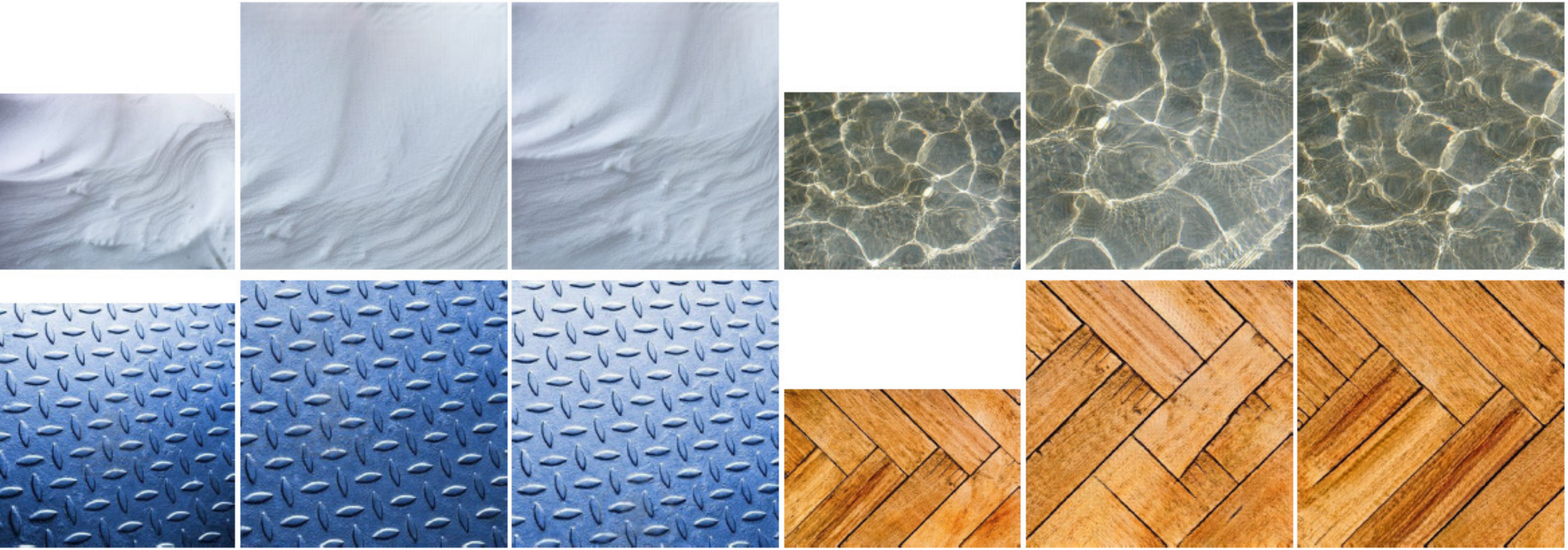
Output

Input

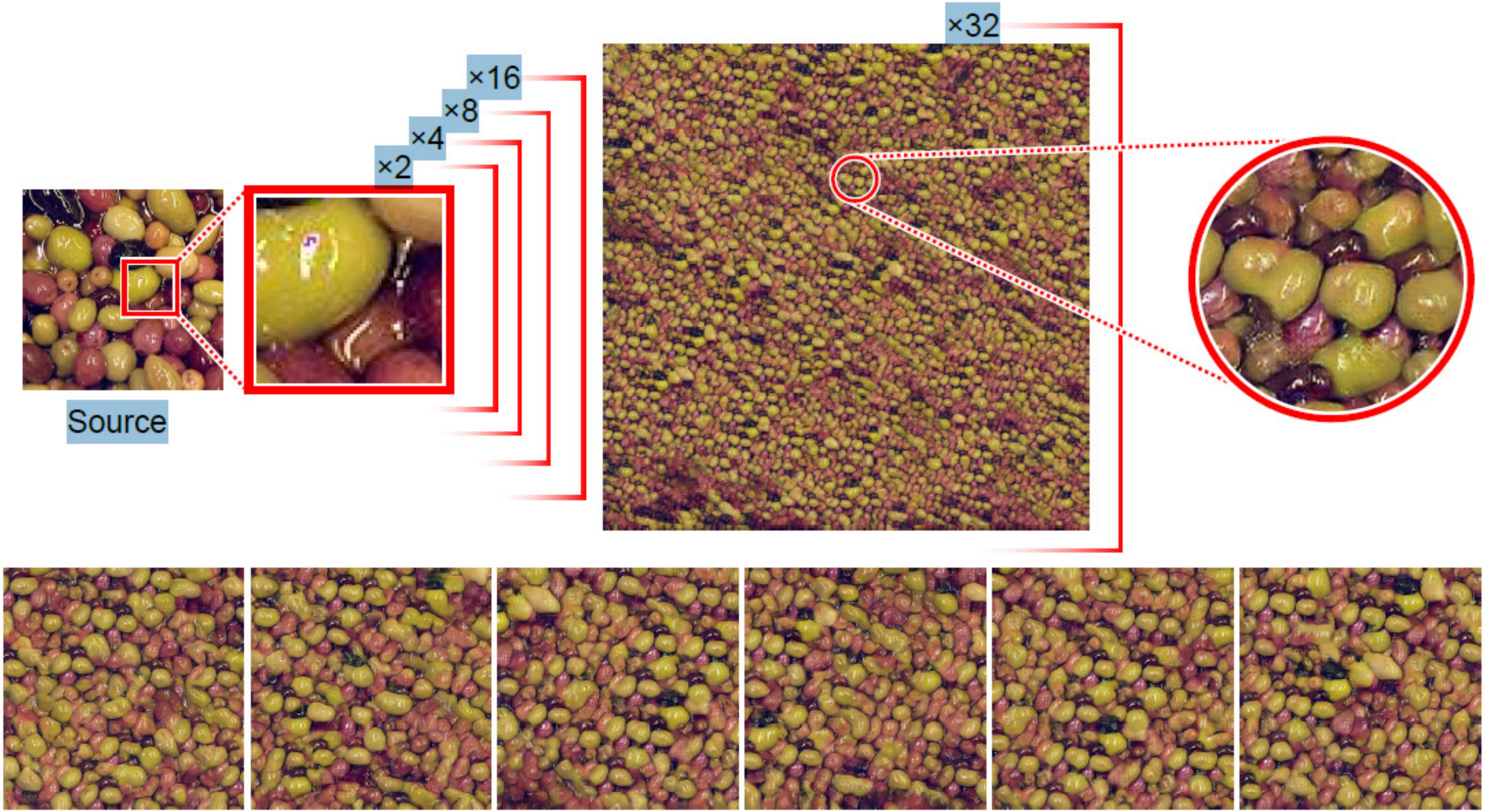


Output

Texture Synthesis by Conditional GANs



Input | Output 1 | Output 2 | Input | Output 1 | Output 2



Random crops of the large result

Style Transfer vs. Image-to-Image Translation

- Data (how to define Style)
  - A single image? A collection of images?
- Applications
  - Photo -> Painting (Neural Style Transfer, Image-to-Image Translation)
  - Photo -> Photo (Image-to-Image Translation, Photo Style Transfer (Color))
  - Painting -> Photo (Image-to-Image Translation, Deep Image Analogy)
- Algorithms:
  - Patch-based method (i.e., correspondence between output and input)
  - Optimization-based method
  - Feed-forward network
- Loss functions
  - Style Loss: GAN loss, Gram matrix loss
  - Content Loss: Perceptual Loss (L2 reconstruction loss), identity loss, conditional GAN Loss, Cycle-consistency loss, Contrastive Loss (InfoNCE)

Style Transfer + Poisson Blending

Motivation: Image compositing on paintings

+

=

Poisson blending

Ours

Deep Painterly Harmonization
[Luan et al., 2018]

Intuition 3: Two-pass framework

- Two-pass harmonzation is more robust than one-pass version

Inputs

Pass 1: Coarse color

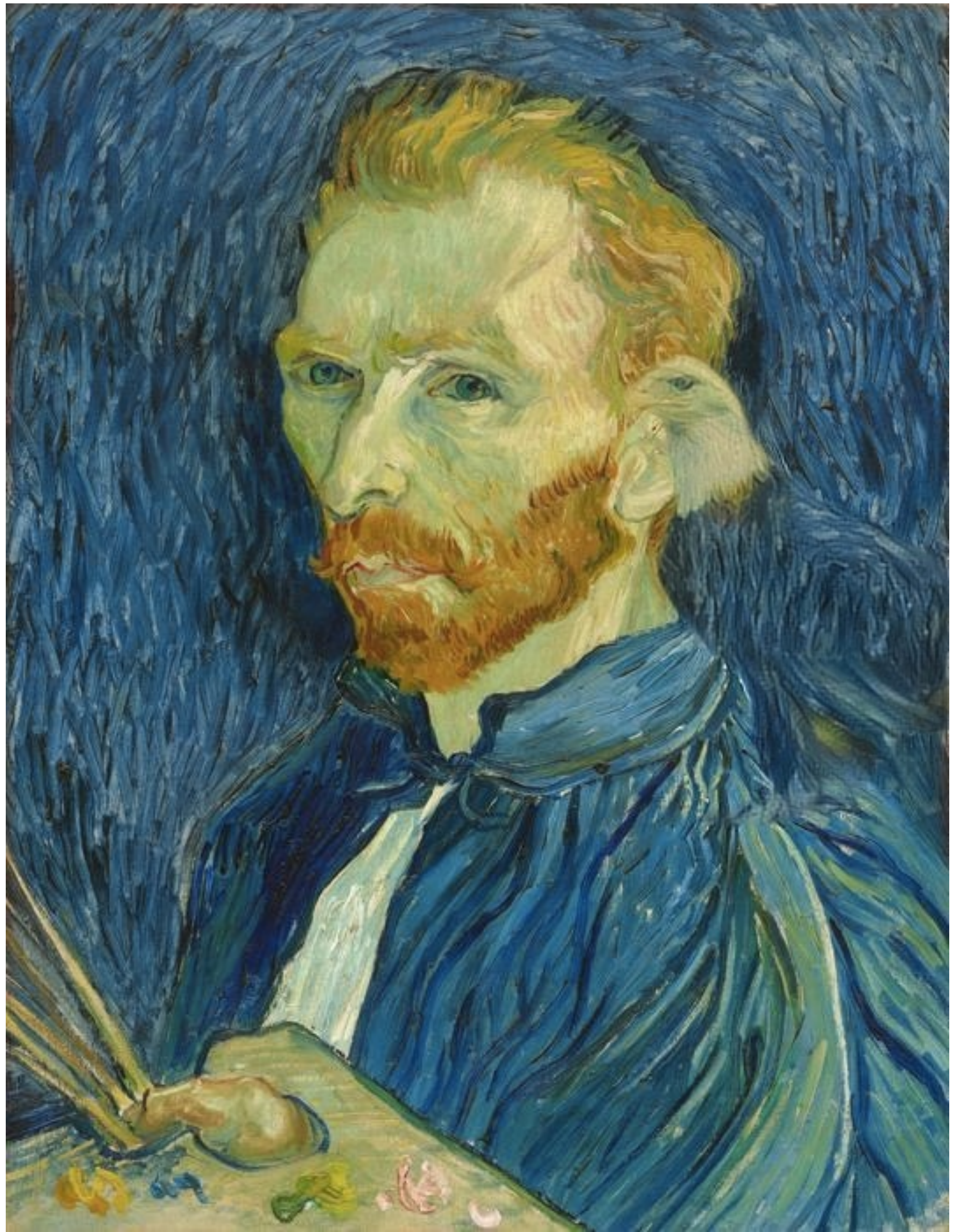
Pass 2: Fine texture

More results

Deep Painterly Harmonization [Luan et al., 2018]

More results

Deep Painterly Harmonization [Luan et al., 2018]



More results

Deep Painterly Harmonization [Luan et al., 2018]