Image Editing with Optimization (part I)

Jun-Yan Zhu

16-726, Spring 2025

Image Editing with Optimization

input photo

Image Editing Program

User Input

result

$$\arg\min_{\hat{y}} \mathcal{L}_{\text{background\_boundary}}(\hat{y}, y) + \lambda \mathcal{L}_{\text{source\_gradient}}(\hat{y}, x)$$

result background result object

[Tao et al. 2014]

Image Editing with Optimization

input photo

Image Editing Program

User Input

result

Desired output:

- stay close to the input.
- satisfy user's constraint.

[Tao et al. 2014]

Image Editing with Optimization

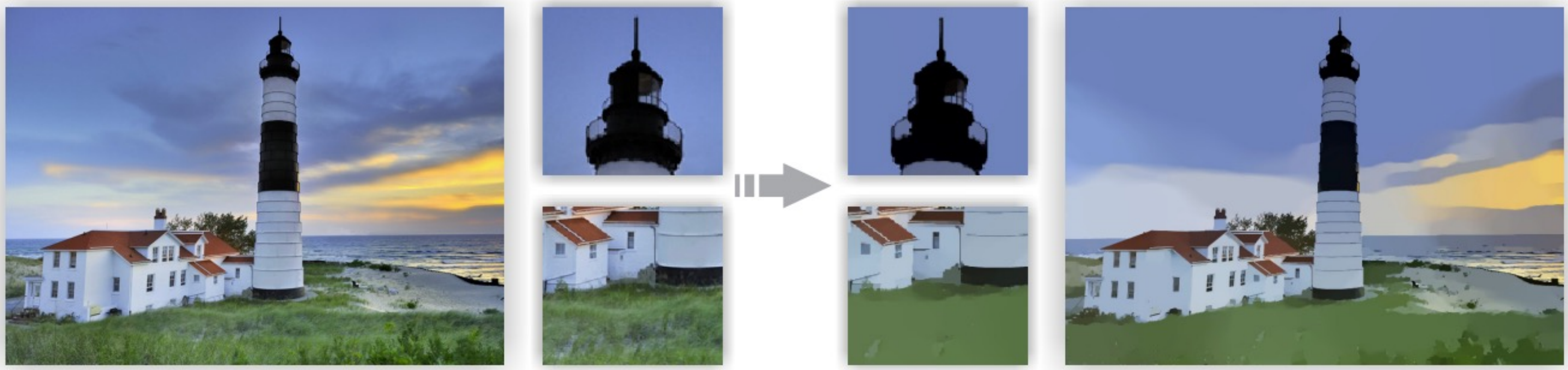
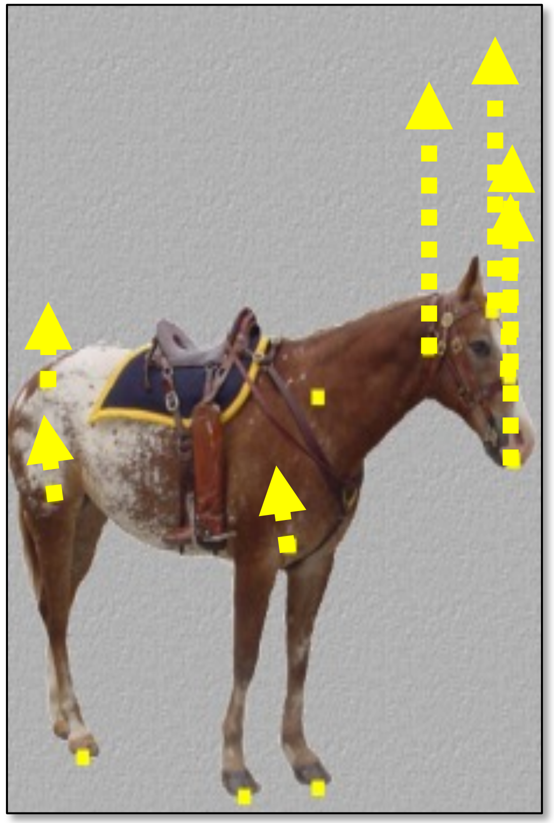


Image Smoothing via L0 Gradient Minimization [Xu et al., SIGGRAPH Asia 2011]

$$\arg\min_{\hat{y}}\{||\hat{y} - x|| + \lambda C(\hat{y})\}$$

output input L0 norm on image gradients
(the total number of nonzero elements)

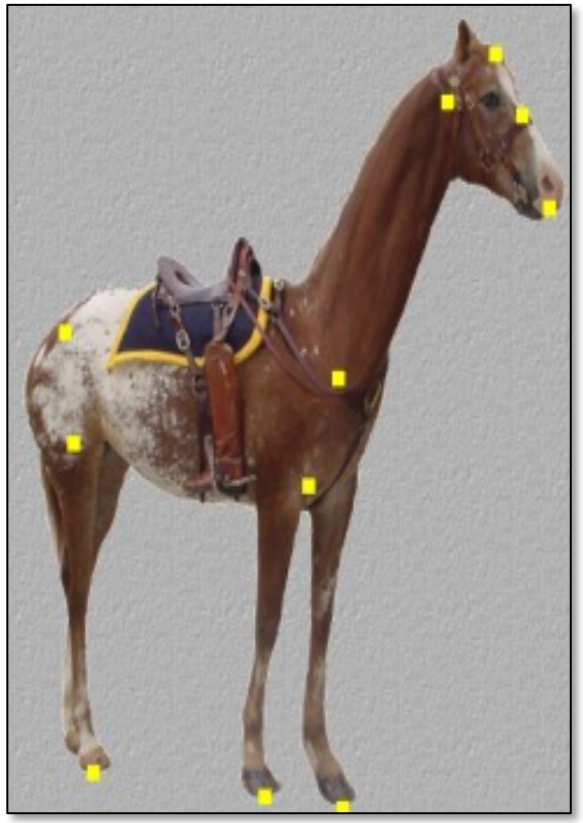
Image Editing with Optimization



input photo

Image Editing Program

User Input



result

Moving least squares + transformation parameters.

Desired output:

- stay close to the input.
- satisfy user’s constraint.

[Schaefer et al. 2006]

So far so good

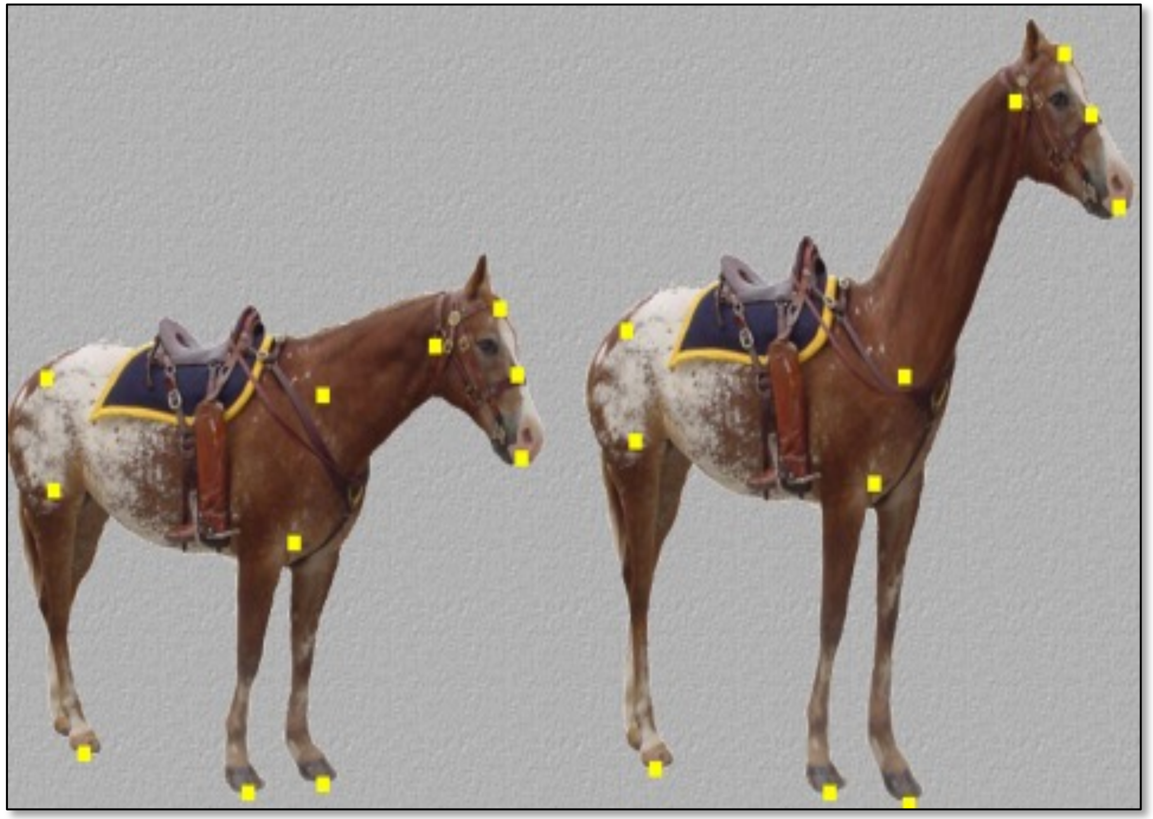


Image Warping

Image Composition

Things can get really bad

Image Warping

Image Composition

The lack of "safety wheels"

Adding the “safety wheels”

Input Photo

Image Editing Program

User Input

Output Result

Natural Image Manifold

A desired output:

- stay close to the input.
- satisfy user’s constraint.
- Lie on the natural image manifold

Learning Natural Image Manifold

- Deep generative models: $G(z) : z \to x$
  - Generative Adversarial Network (**GAN**)
    (e.g., DCGAN, StyleGAN2, BigGAN)
  - Variational Auto-Encoder (**VAE**)
    (e.g., VQ-VAE2)
  - Flow-based models (e.g., RealNVP, Glow)…
  - Diffusion models (e.g., DDPM, DDIM)

  …

Changing Variables

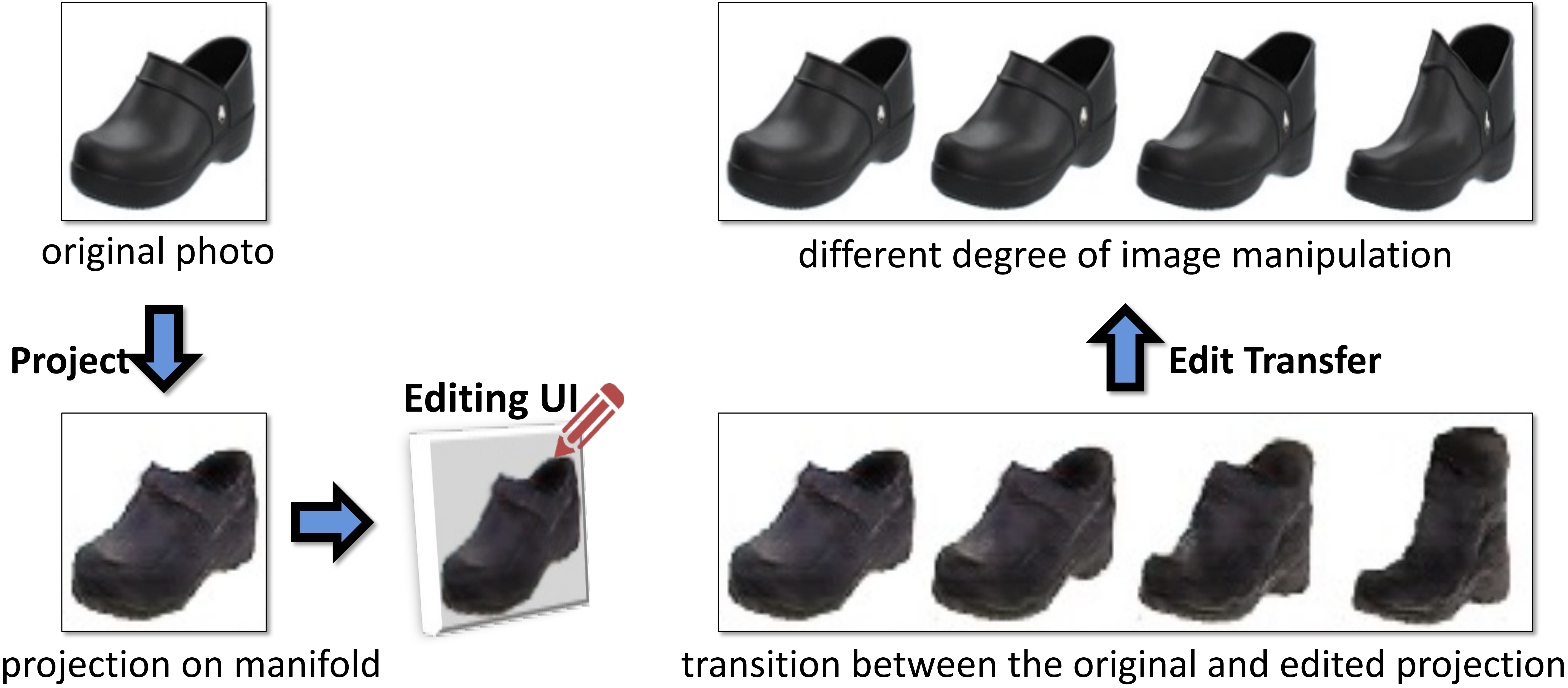
- Traditional method: Optimizing the image

$$\hat{y}^* = \arg\min_{\hat{y}} \mathcal{L}(\underset{\text{input}}{x}, \overset{\text{user constraint}}{y}, \underset{\text{output}}{\hat{y}})$$

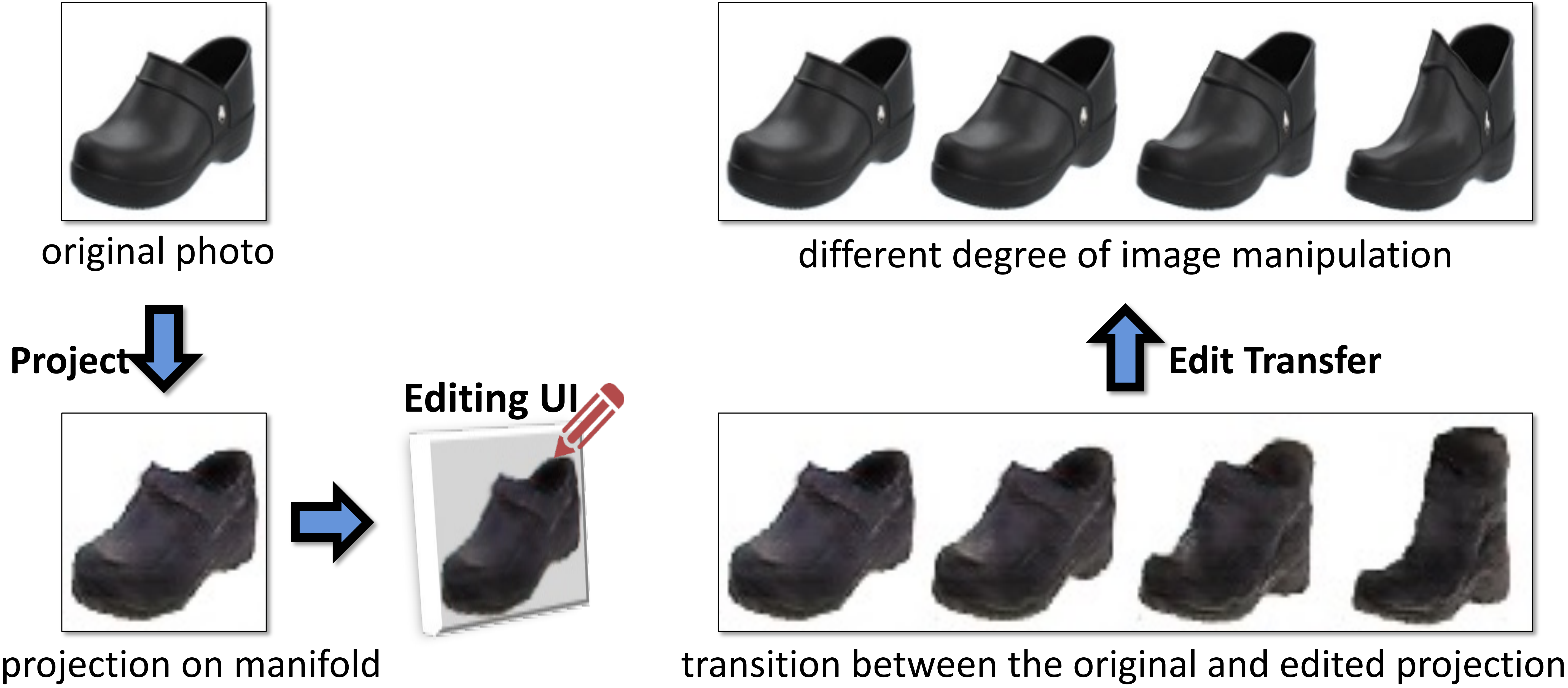
- New method: Optimizing the latent code

$$z^* = \arg\min_{z} \mathcal{L}(\underset{\text{input}}{x}, \overset{\text{user constraint}}{y}, \underset{\text{Generator}}{G}(\underset{\text{Latent code}}{z}))$$

Projecting and Editing an Image



Projecting and Editing an Image



Projecting an Image into GAN Manifold

Input: real image x

Output: latent vector z

Optimization

$$z^* = \arg\min_z \mathcal{L}(G(z), x)$$

Reconstruction loss

Generative model

0.196 0.238 0.332

Projecting an Image into GAN Manifold

Input: real image x

Output: latent vector z

Optimization

$$z^* = \arg\min_z \mathcal{L}(G(z), x)$$

Inverting Network $z = E(x)$

$$E = \arg\min_E \mathbb{E}_x \underbrace{\mathcal{L}(G(E(x)), x)}_{\text{Auto-encoder with a fixed decoder}}$$

Projecting an Image into GAN Manifold

Input: real image x

Output: latent vector z

Optimization

$$z^* = \arg\min_z \mathcal{L}(G(z), x)$$

Inverting Network $z = E(x)$

$$E = \arg\min_E \mathbb{E}_x \mathcal{L}(G(E(x)), x)$$

Hybrid Method

Use the network as initialization for the optimization problem

Manipulating the Latent Code

Manipulating the Latent Code

constraint violation loss L_g

user guidance image

Objective:

$$z^* = \arg\min_{z \in \mathbb{Z}} \Big\{ \underbrace{\sum_g (\mathcal{L}_g(G(z), v_g)}_{\text{data term}} + \underbrace{\lambda_s \cdot \|z - z_0\|_2^2}_{\substack{\text{manifold}\\\text{smoothness}}} \Big\}.$$

Guidance v_g

$G(z)$

z_0

Post-Processing

original photo

Project

projection on manifold

Editing UI

different degree of image manipulation

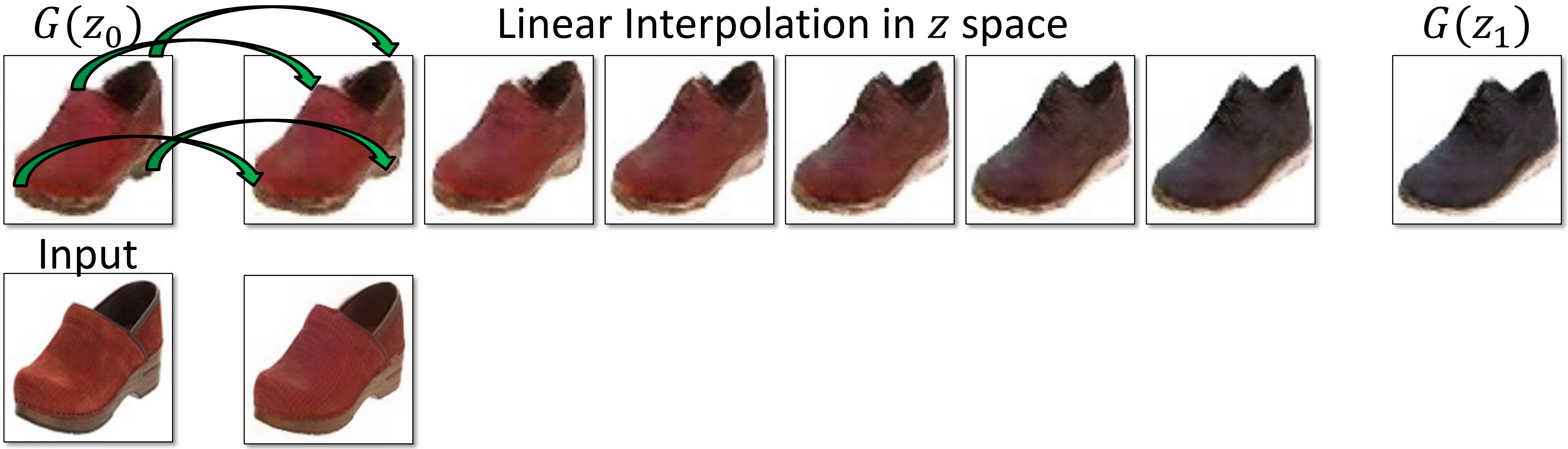
Edit Transfer

transition between the original and edited projection

Edit Transfer

Motion (u, v)+ Color ($A_{3\times 4}$): estimate per-pixel geometric and color variation

$$\iint \underbrace{\|I(x,y,t) - \boxed{A} \cdot I(x+\boxed{u}, y+\boxed{v}, t+1)\|^2}_{\text{data term}} + \underbrace{\sigma_s(\|\nabla u\|^2 + \|\nabla v\|^2)}_{\text{spatial reg}} + \underbrace{\sigma_c \|\nabla A\|^2}_{\text{color reg}} dxdy$$



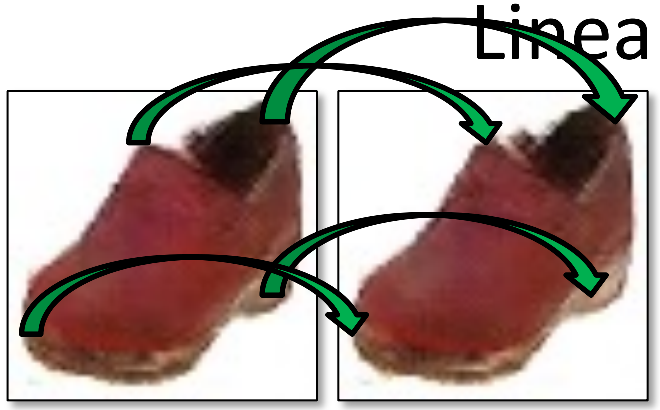
Edit Transfer

Motion (u, v)+ Color ($A_{3\times4}$): estimate per-pixel geometric and color variation

$$\iint \underbrace{\|I(x,y,t) - \boxed{A}\cdot I(x+\boxed{u}, y+\boxed{v}, t+1)\|^2}_{\text{data term}} + \underbrace{\sigma_s(\|\nabla u\|^2 + \|\nabla v\|^2)}_{\text{spatial reg}} + \underbrace{\sigma_c\|\nabla A\|^2}_{\text{color reg}} dxdy$$

$G(z_0)$

Linear Interpolation in z space



$G(z_1)$

Input

Edit Transfer

Motion (u, v)+ Color ($A_{3\times4}$): estimate per-pixel geometric and color variation

$$\iint \underbrace{\|I(x,y,t)-\boxed{A}\cdot I(x+\boxed{u},y+\boxed{v},t+1)\|^2}_{\text{data term}}+\underbrace{\sigma_s(\|\nabla u\|^2+\|\nabla v\|^2)}_{\text{spatial reg}}+\underbrace{\sigma_c\|\nabla A\|^2}_{\text{color reg}}dxdy$$

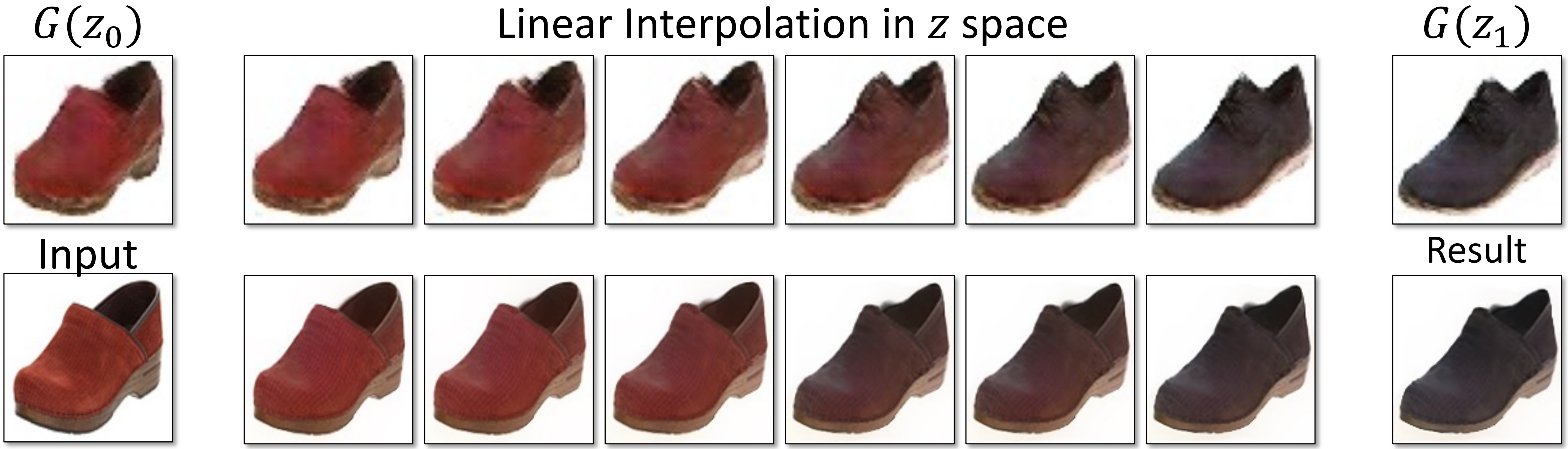


Image Manipulation Demo

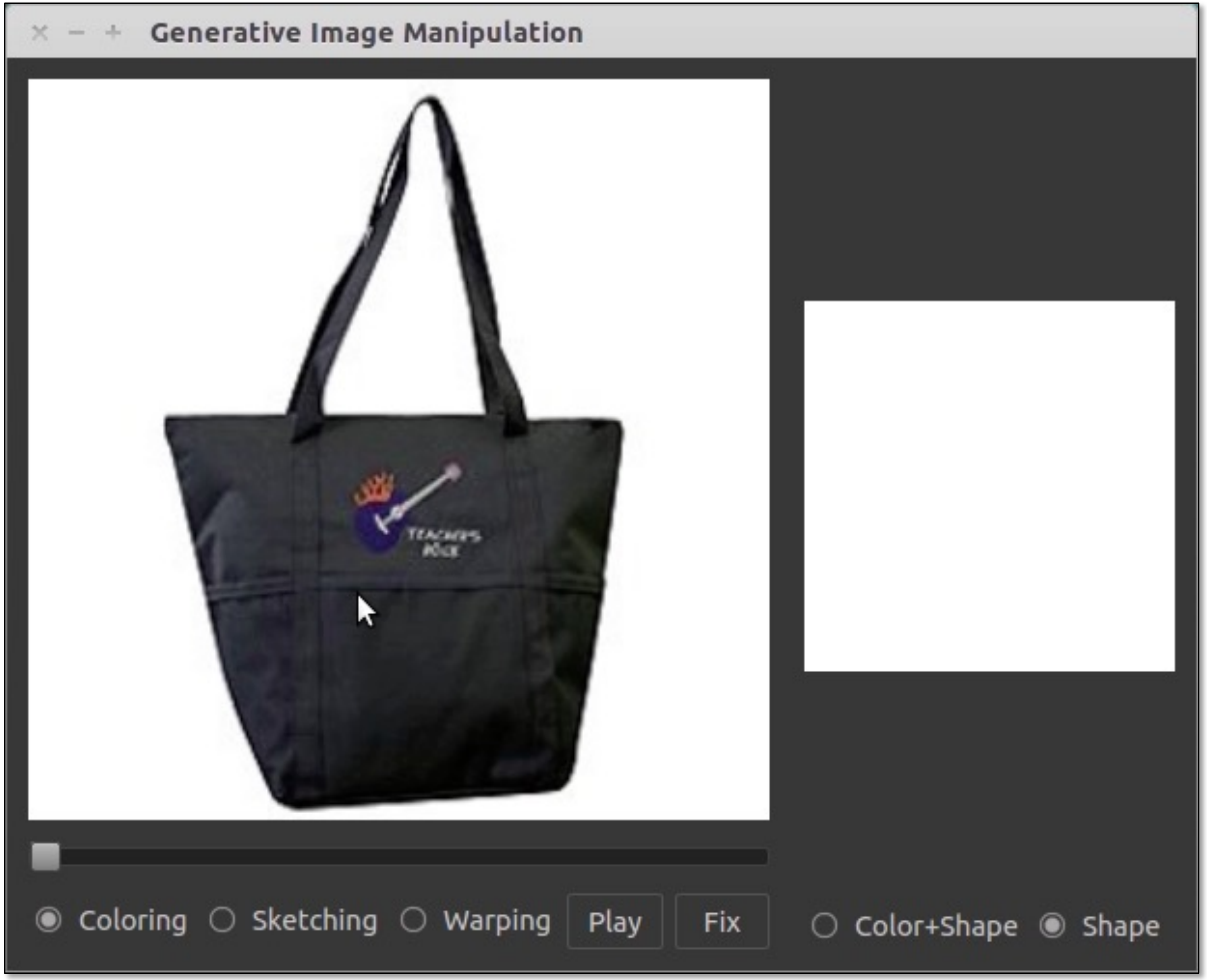
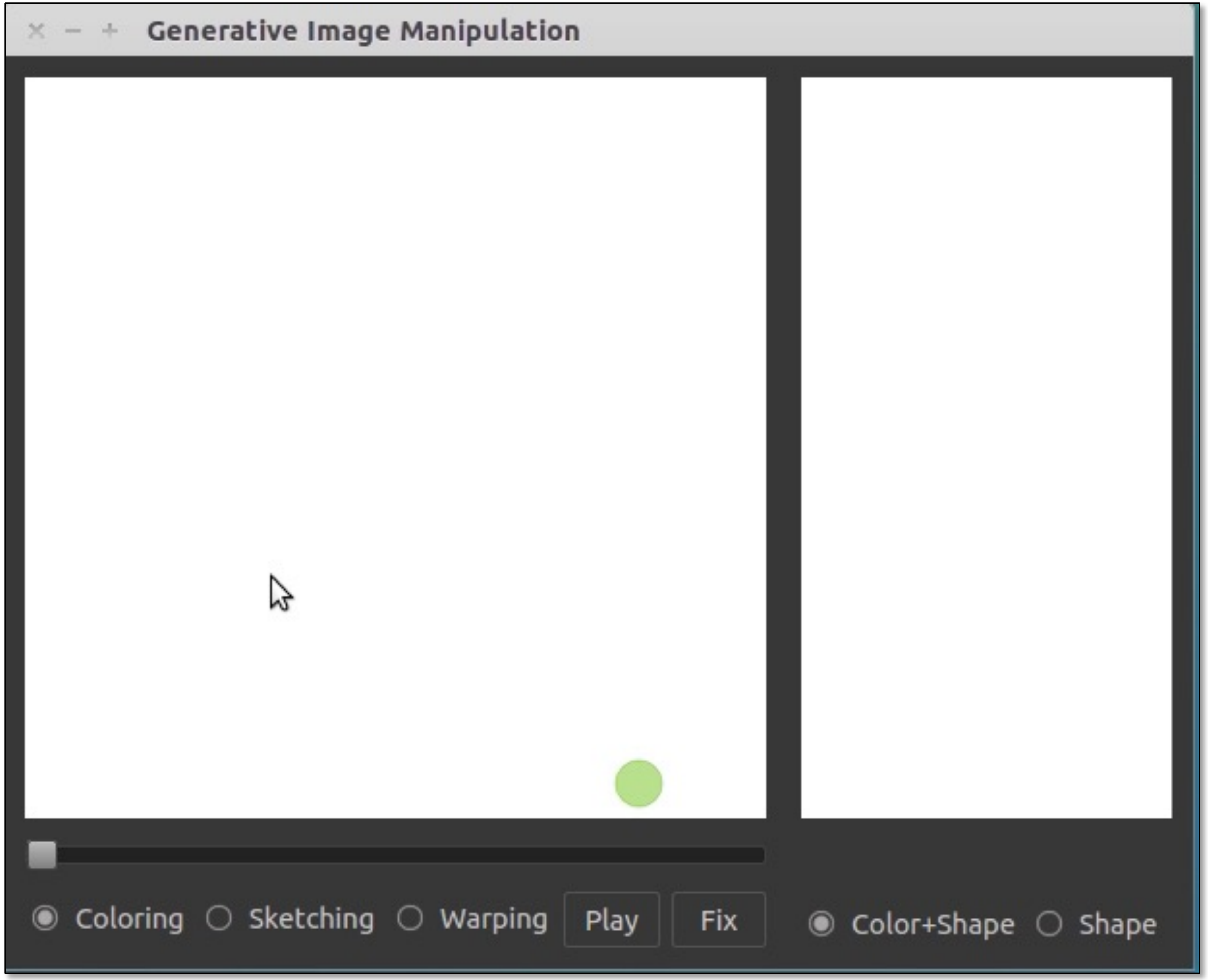
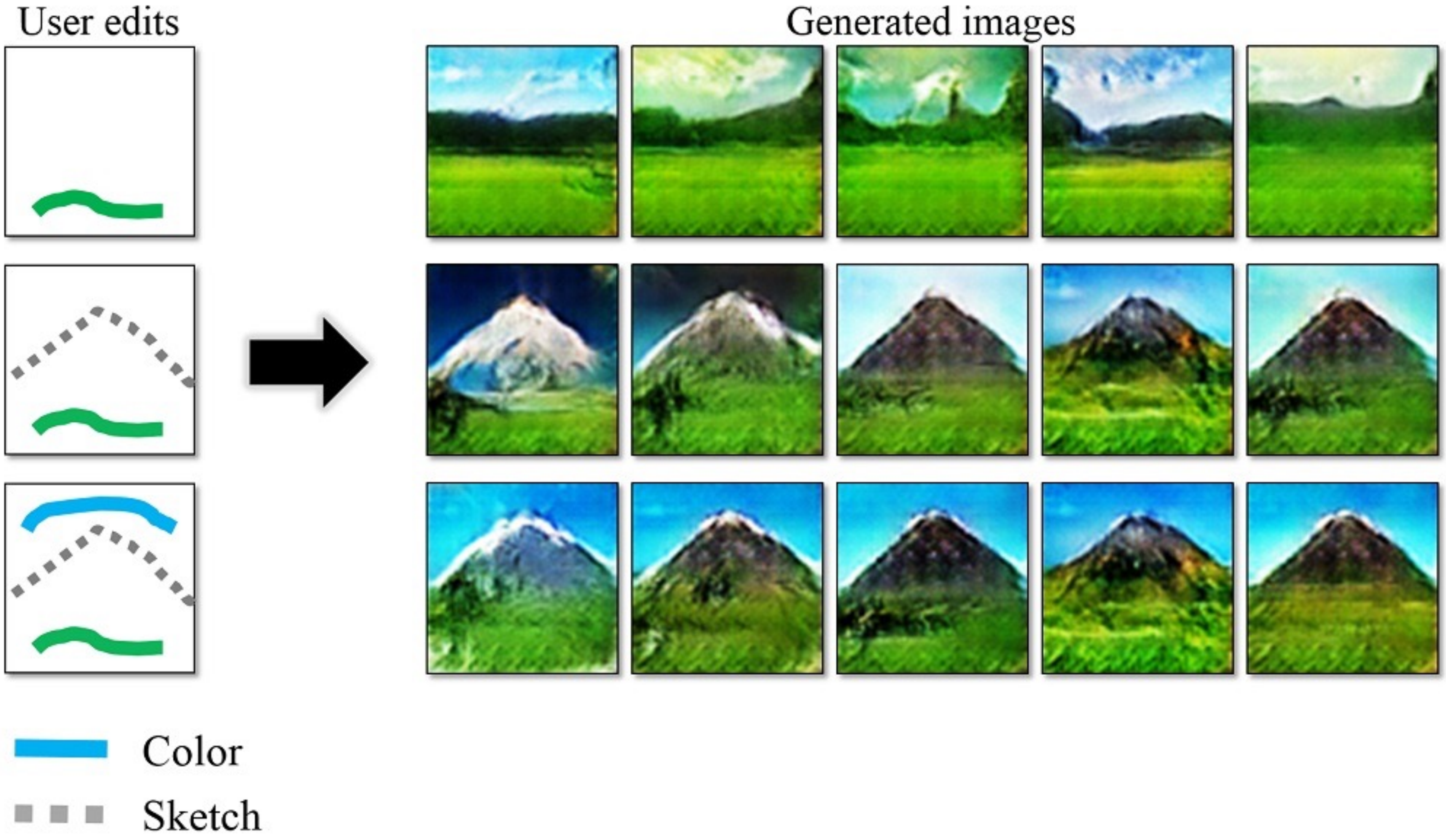


Image Manipulation Demo

Interactive Image Generation





iGAN [Zhu et al. 2016]. Also see Neural Photo Editor [Brock et al. 2017]

Changing Variables

- Traditional method: Optimizing the image

$$\hat{y}^* = \arg\min_{\hat{y}} \mathcal{L}(\underbrace{x}_{\text{input}}, \overset{\text{user constraint}}{y}, \underbrace{\hat{y}}_{\text{result}})$$

- New method: Optimizing the latent code

$$z^* = \arg\min_{z} \mathcal{L}(\underbrace{x}_{\text{input}}, \overset{\text{user constraint}}{y}, \underbrace{G}_{\text{Generator}}(\underbrace{z}_{\text{Latent code}}))$$

Projecting and Editing an Image

original photo

Project

projection on manifold

Editing UI

transition between the original and edited projection

Post-processing

different degree of image manipulation

Image Editing with GANs

- Step 1: Image Projection/Reconstruction

$$z_0 = \arg\min_z \mathcal{L}(G(z), x)$$

- Step 2: Manipulating the latent code

$$z_1 = z_0 + \Delta z$$

- Step 3: Generate the edited result

$$G(z_1)$$

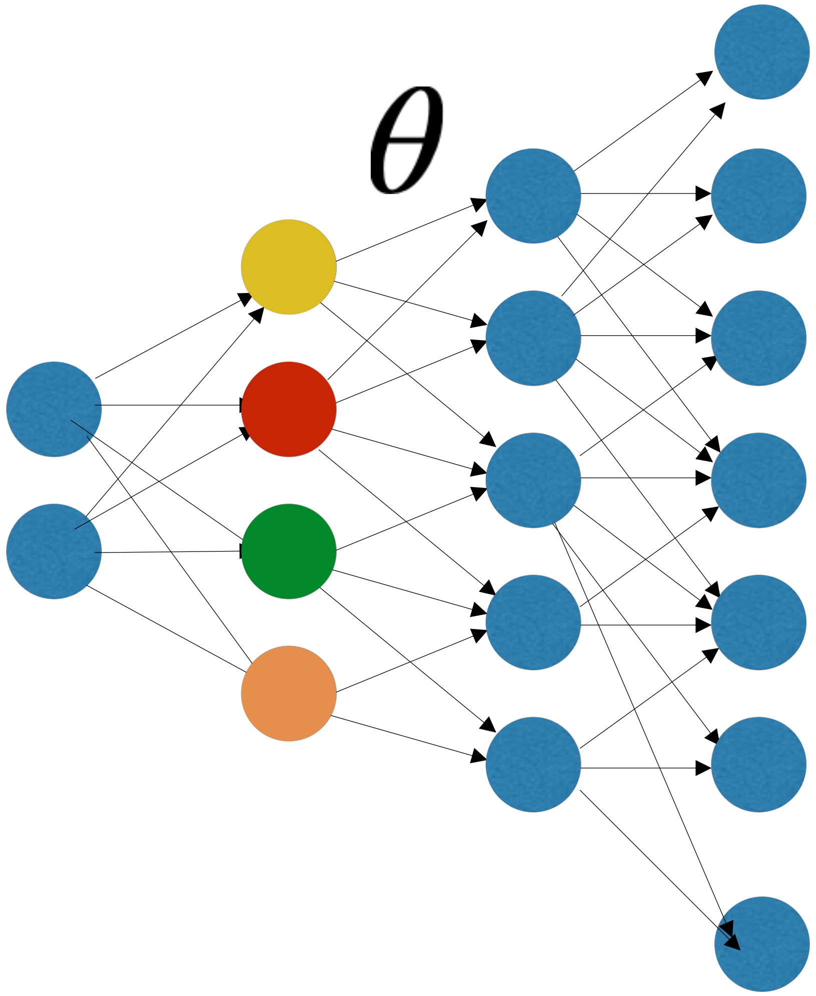
Image Projection with GANs

Image Reconstruction (high-res images, Big Models)

Original image x

Image Reconstruction (high-res images, Big Models)

Random vector z ?

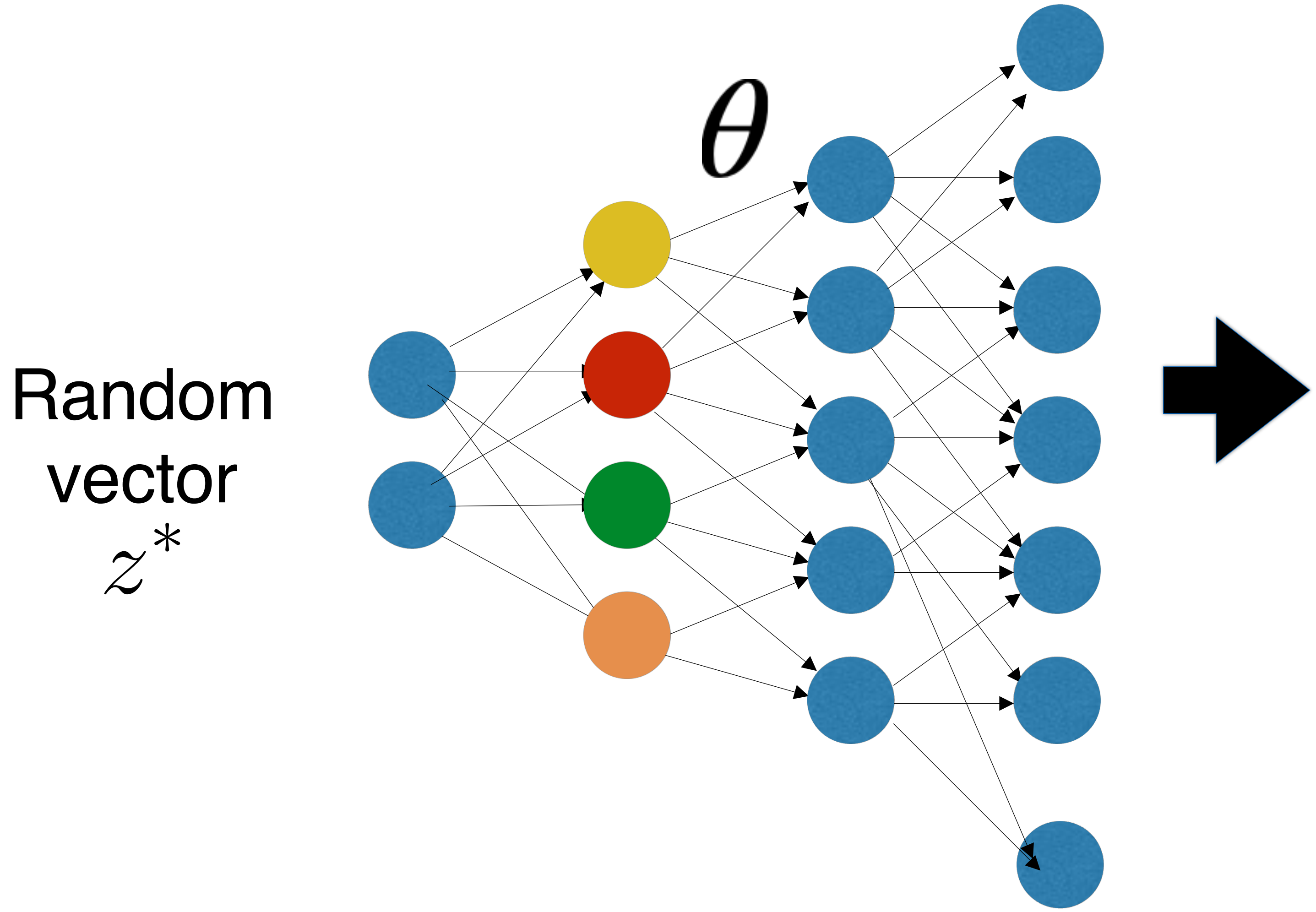


Original image x

$$z^* = \arg\min_z \mathcal{L}(G(z;\theta), x)$$

iGAN [Zhu et al. 2016]

Image Reconstruction (high-res images, Big Models)



Reconstructed image $G(z^*; \theta)$

$$z^* = \arg\min_z \mathcal{L}(G(z;\theta), x)$$

iGAN [Zhu, Krähenbühl, Shechtman, Efros]

Find the Differences…

Original image

GAN reconstructed image

Find the Differences...

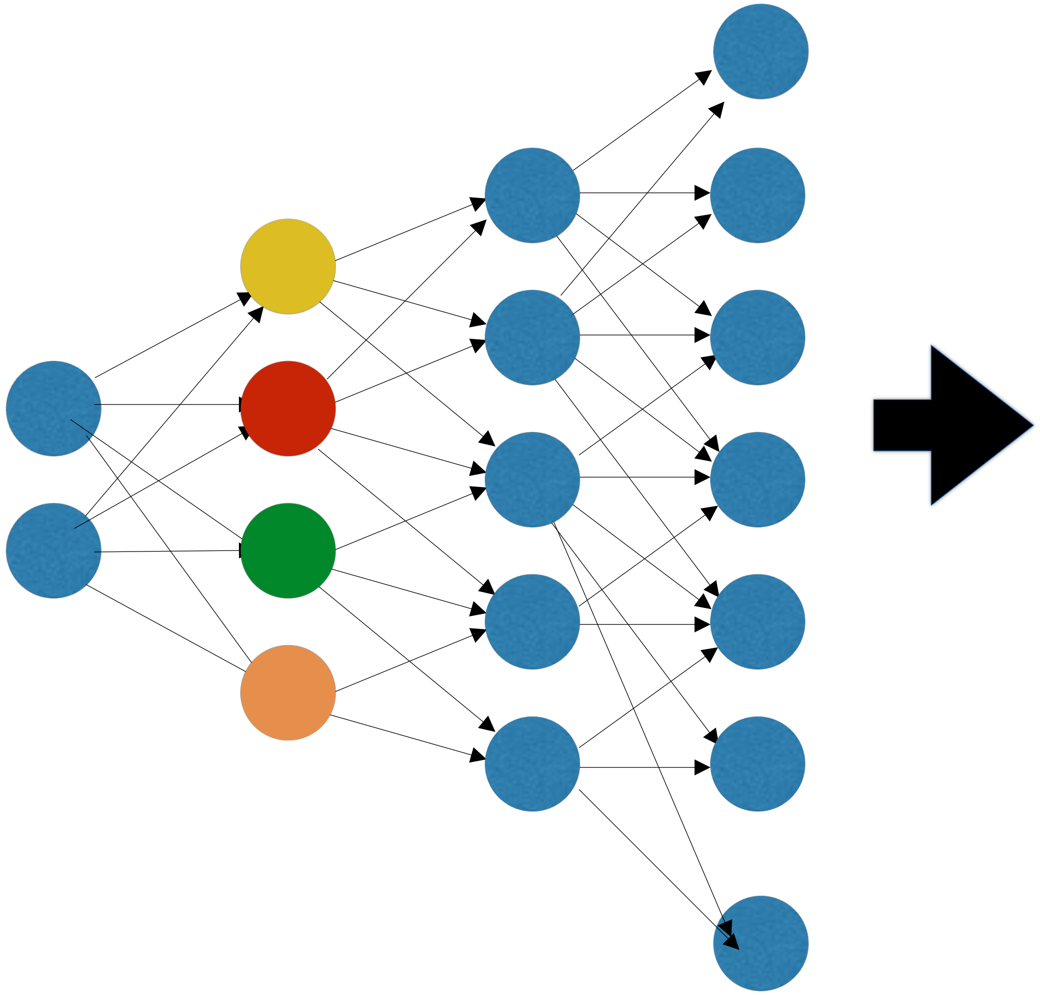
Original image

GAN reconstructed image

Reconstructing a Real Photo

Original image

Random vector z^*



Reconstructed image $G(z^*; \theta)$

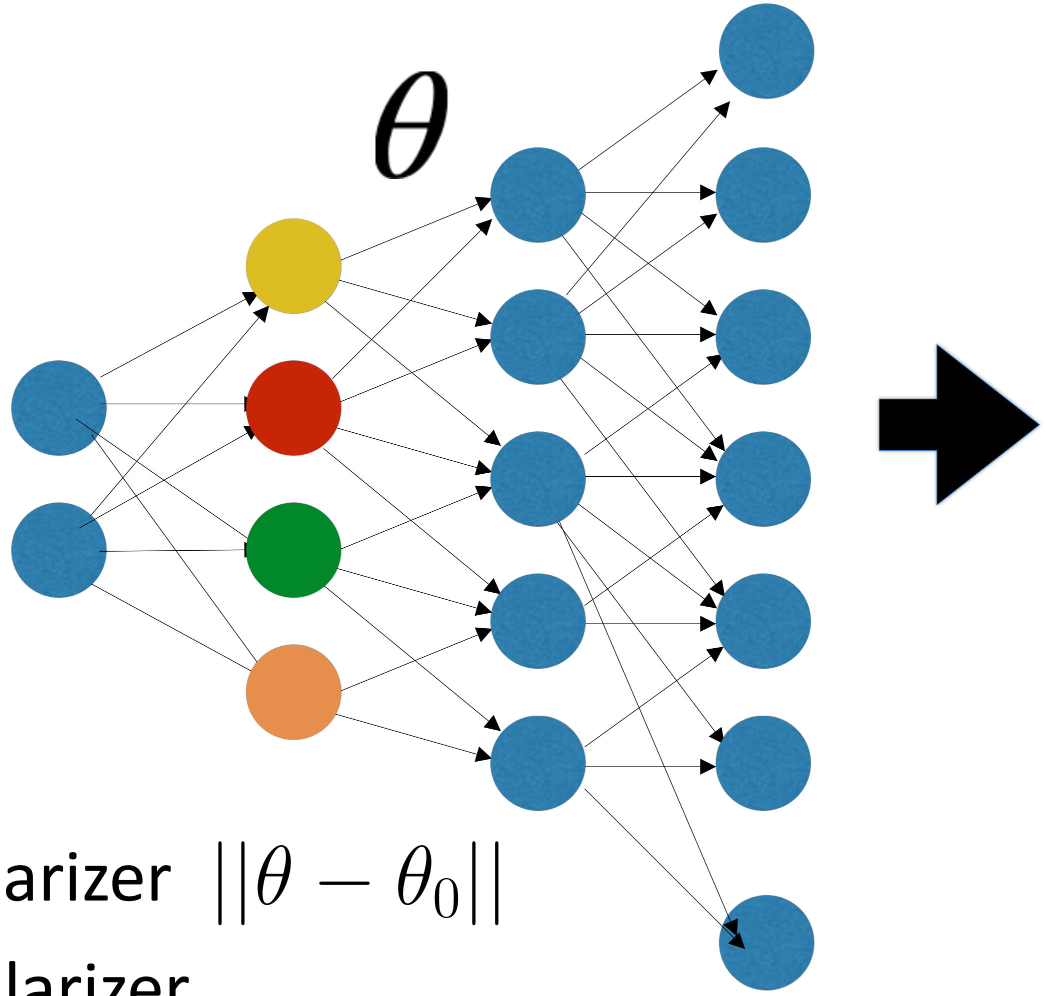
$$z^* = \arg\min_z \mathcal{L}(G(z;\theta), x)$$

iGAN [Zhu et al. 2016]

Original image

Reconstructing a Real Photo

Random vector
z^*



Reconstructed image $G(z^*; \theta)$

Weight space regularizer $||\theta - \theta_0||$

Feature space regularizer

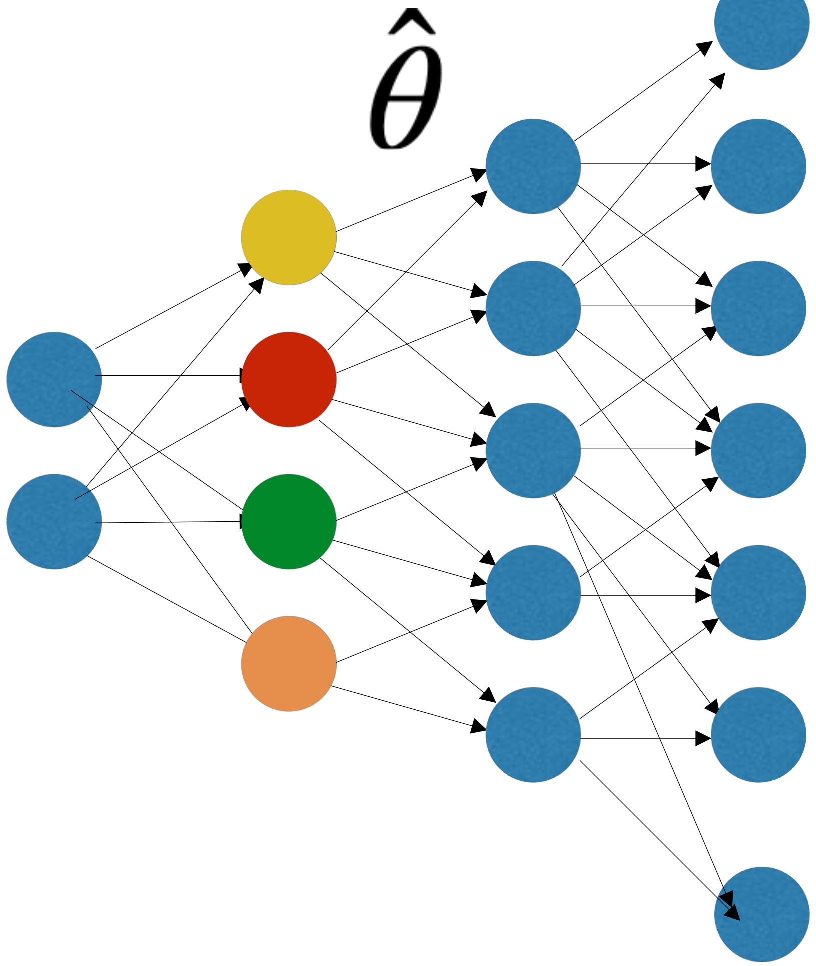
$$z^*, \theta^* = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x)$$

⟵ Regularizer

Reconstructing a Real Photo

Original image

Random vector z^*



Reconstructed image $G(z^*; \theta^*)$

$$z^*, \theta^* = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x) + R(\theta)$$

⟵ Regularizer

Reconstructing a Real Photo

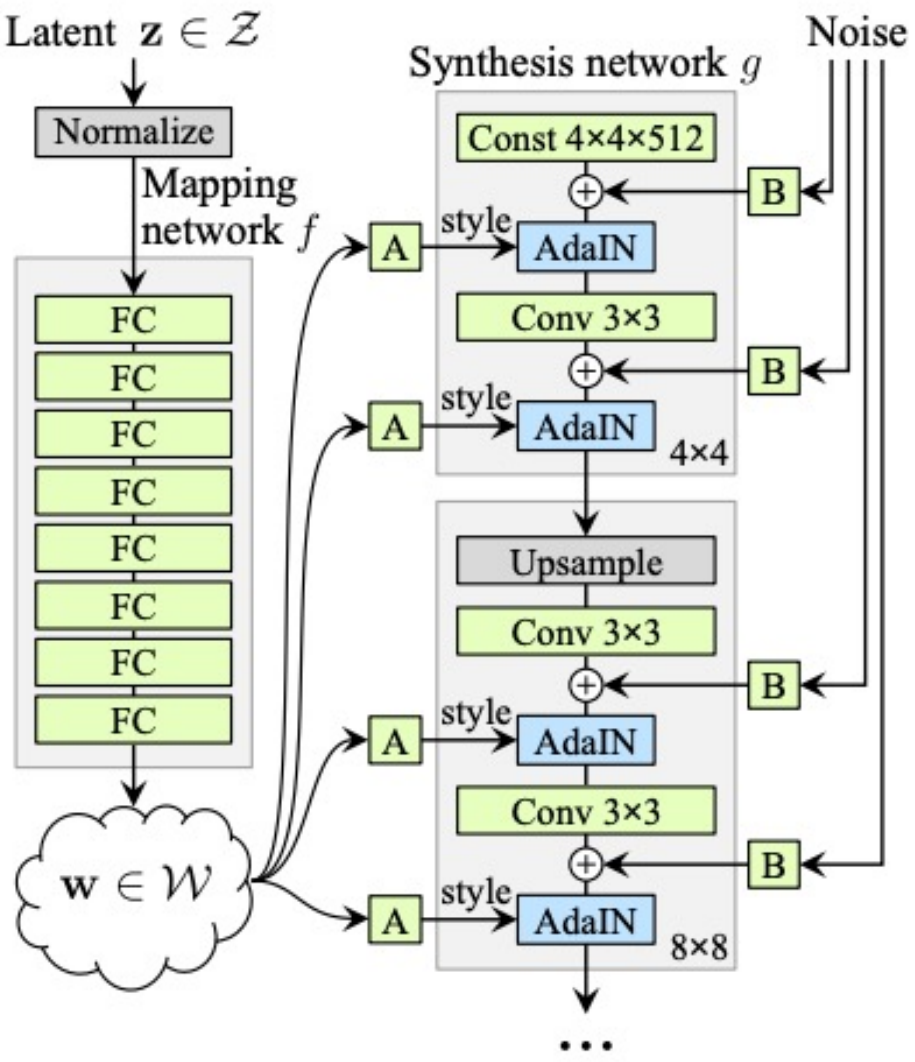
Original image

With z^*

With z^* and θ^*

Semantic Photo Manipulation [Bau, Strobelt, Peebles, Wulff, Zhou, Zhu, Torralba, SIGGRAPH 2019]
Inspired by Deep Image Prior [Ulyanov et al.] and Deep Internal learning [Shocher et al.]

Using Different Layers



Optimizing the latent code

$$z^* = \arg\min_z \mathcal{L}(G(z), x)$$

Optimizing the style code

$$w^* = \arg\min_w \mathcal{L}(g(w), x)$$

Optimizing the extended style code

$$w_+^* = \arg\min_{w+} \mathcal{L}(g(w_+), x)$$

Image2StyleGAN [Abdal et al., 2019], StyleGAN2 [Karras et al., 2019]

Using Different Layers: w space

StyleGAN — generated images

StyleGAN2 — generated images

StyleGAN2 [Karras et al., 2019]

Using Different Layers: w space

StyleGAN2 — real images

StyleGAN2 [Karras et al., 2019]

Using Different Layers: w+ space

All the results are reconstructed using Face Model

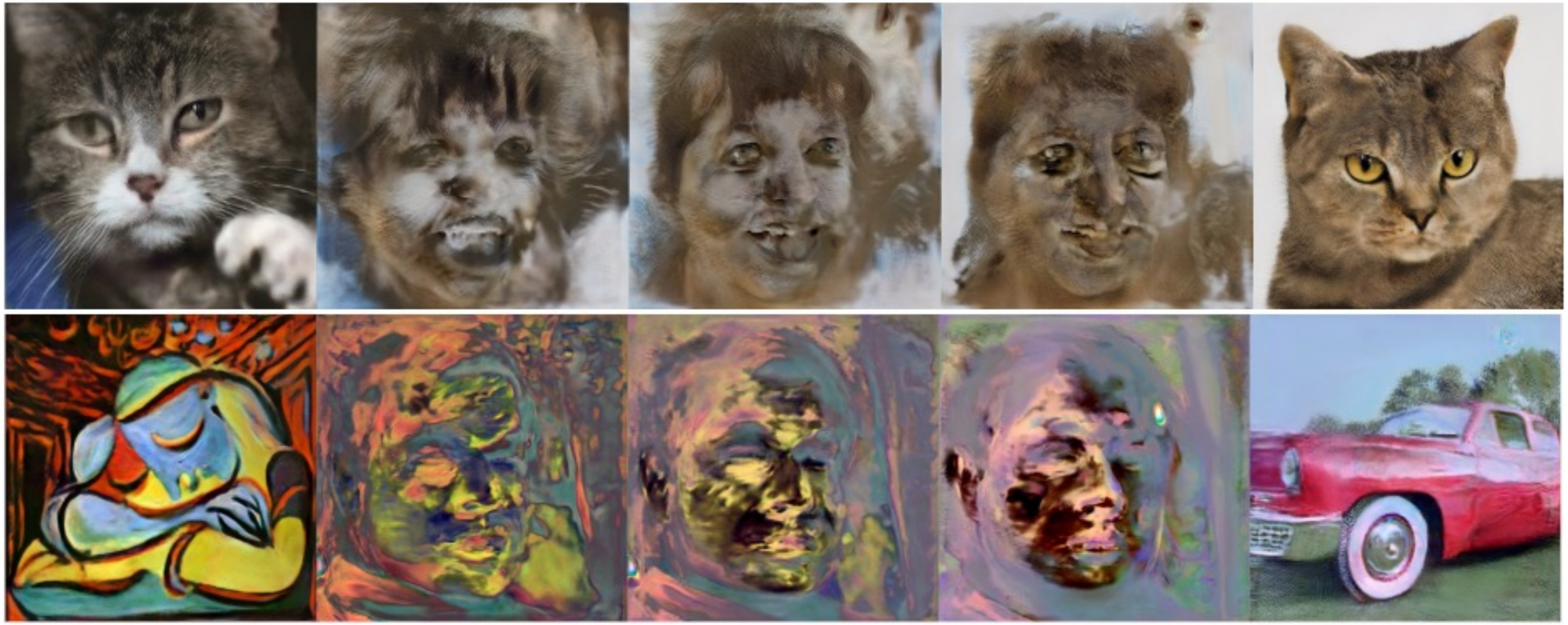
Image2StyleGAN [Abdal et al., 2019]

Reconstruction $\neq$ Editing

Interpolations between two images

Image2StyleGAN [Abdal et al., 2019]

Reconstruction $\neq$ Editing



Interpolations between two images

Image2StyleGAN [Abdal et al., 2019]

How to Improve GANs Projection

- Baseline: Optimizing the latent code

$$z^* = \arg\min_{z} \mathcal{L}(G(z;\theta), x)$$

- Generator fine-tuning:

$$z^*, \theta^* = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x) + R(\theta)$$

- Optimizing intermediate features

$$w_+^* = \arg\min_{w+} \mathcal{L}(g(w_+), x)$$

How to Improve GANs Projection

- Baseline: Optimizing the latent code

$$z^* = \arg\min_z \mathcal{L}(G(z;\theta), x)$$

- Generator fine-tuning:

$$z^*, \theta^* = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x) + R(\theta)$$

- Optimizing intermediate features

$$w_+^* = \arg\min_{w+} \mathcal{L}(g(w_+), x)$$

Used together

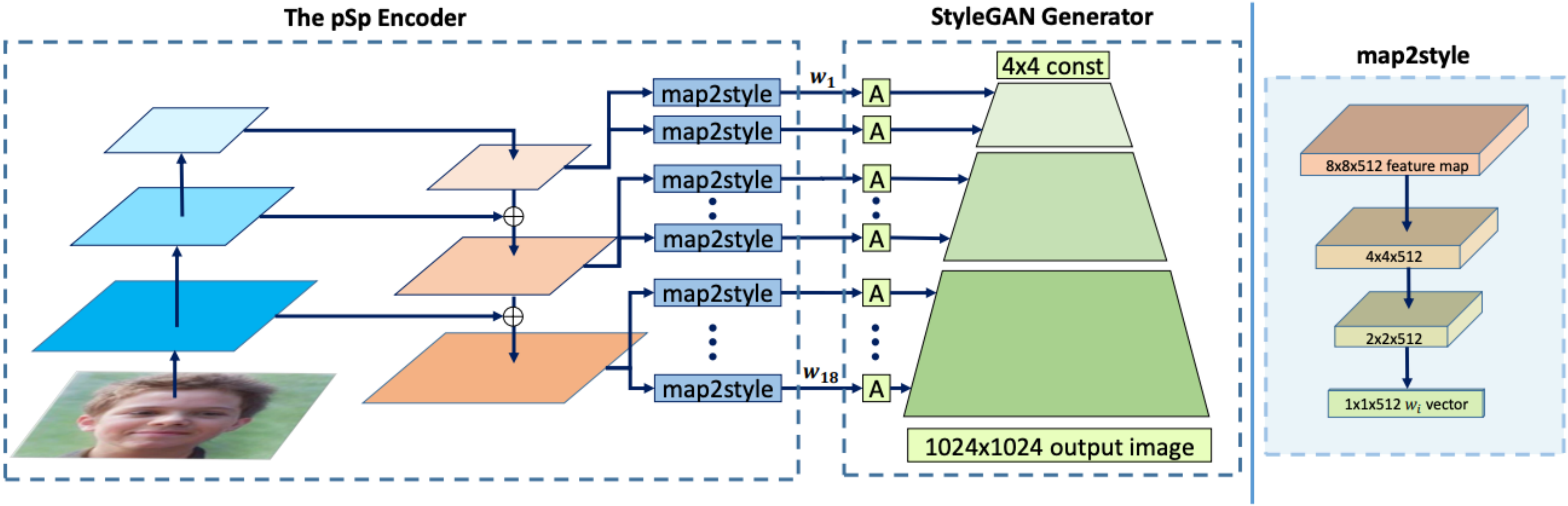
How to Improve GANs Projection

- Baseline: Optimizing the latent code

$$z^* = \arg\min_z \mathcal{L}(G(z;\theta), x)$$

- Training an encoder E(x). Advantages?
  - Faster inference
  - More reliable initialization
- Encoder design depends on
  - Generator architecture.
  - Which latent space: z, w, w+.
  - Pre-trained network weights.

Example: An StyleGAN Encoder



Encoding in Style: a StyleGAN Encoder for Image-to-Image Translation
[Richardson et al., CVPR 2021]

Example: An StyleGAN Encoder

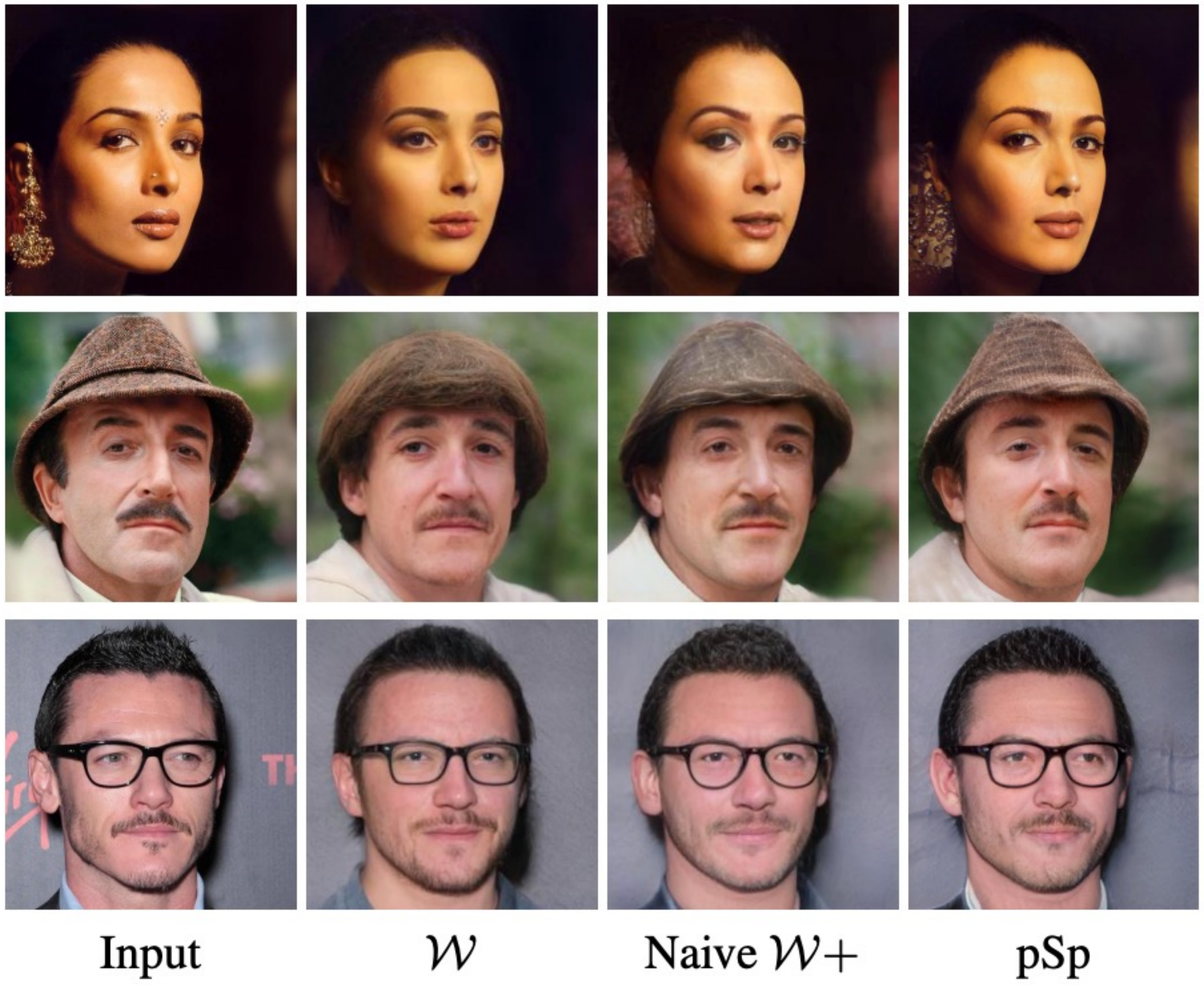


Image Editing with GANs

- Step 1: Image Projection/Reconstruction

$$z^*, \theta^* = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x) + R(\theta)$$

- Step 2: Manipulating the latent code

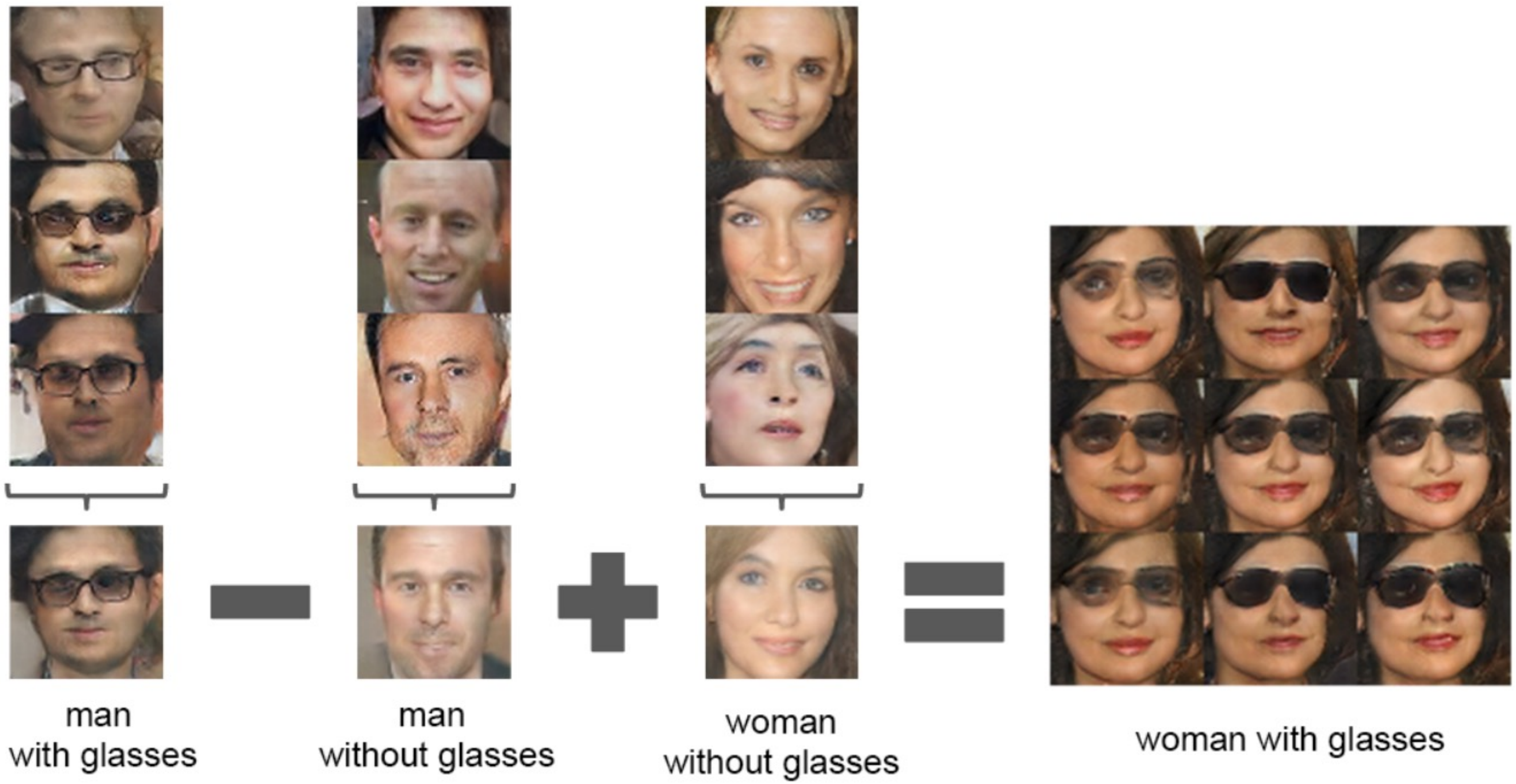
$$z_1 = z_0 + \Delta z$$

- Step 3: Generate the edited result

$$G(z_1)$$

Manipulating Latent code/layer (computing directions offline)

Compute Δz



Step 1: annotate images (manually or via a pre-trained classifier)

Step 2: compute directions

DCGAN [Radford et al. 2016]

Manipulating Latent code/layer (PCA directions)

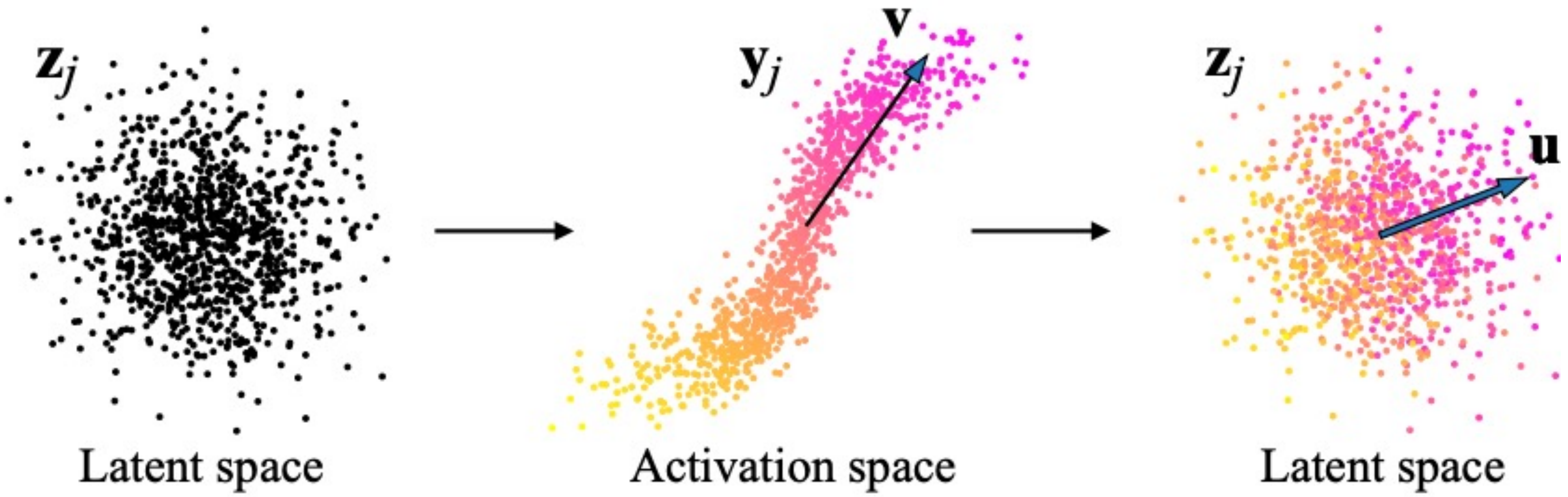
GANSpace: Discovering PCA directions

First compute potential directions (PCA), then name them

GANspace [Härkönen et al. 2020]

GANSpace: Discovering PCA directions

z: latent codes. y: intermediate features.
v: PCA direction in feature space , u: PCA direction in latent space



Also see “Editing in Style: Uncovering the Local Semantics of GANs”, Collins et al., CVPR 2020
“Closed-Form Factorization of Latent Semantics in GANs”, Shen and Zhou. CVPR 2021

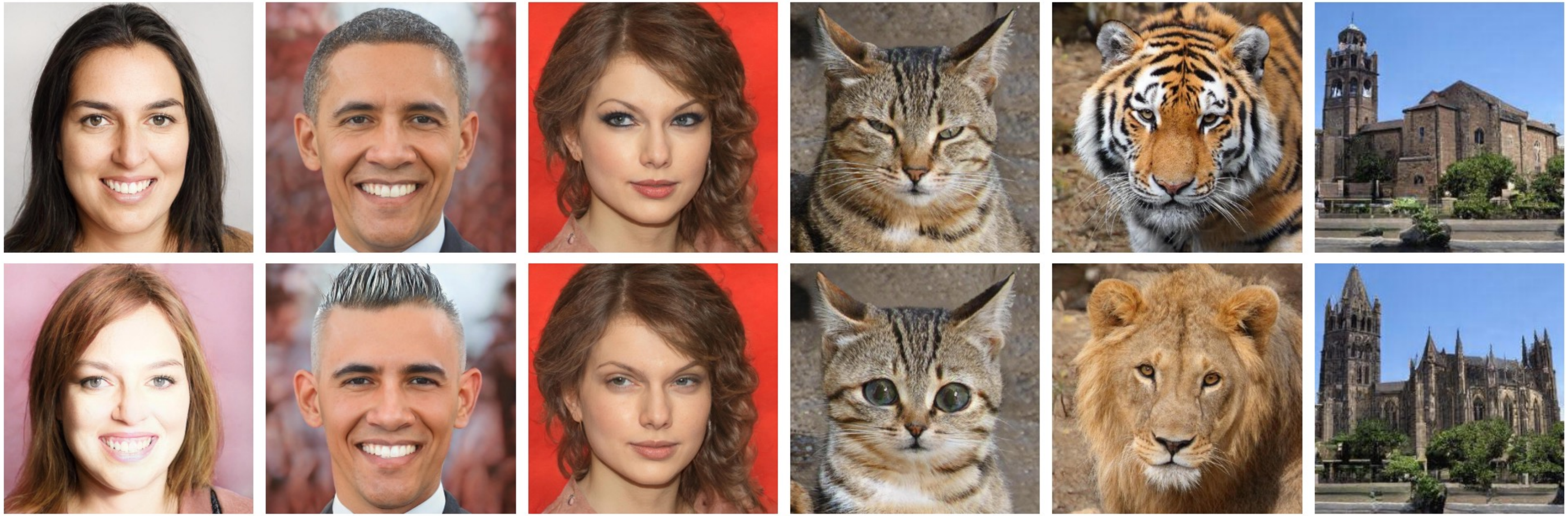
GANspace [Härkönen et al., 2020]

GANSpace: Discovering PCA directions

GANspace [Härkönen et al., 2020]

Manipulating Latent code/layer (Text-guided optimization)

CLIP-guided Directions



“Emma Stone” “Mohawk hairstyle” “Without makeup” “Cute cat” “Lion” “Gothic church”

$$\underset{w \in \mathcal{W}+}{\arg\min}\ D_{\text{CLIP}}(G(w), t) + \lambda_{\text{L2}} \|w - w_s\|_2 + \lambda_{\text{ID}} \mathcal{L}_{\text{ID}}(w)$$

Output is close to the text Close to the original latent Output is close to input

StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery [Patashnik et al., ICCV 2021]

CLIP-guided Directions

$$\underset{w \in \mathcal{W}+}{\arg\min}\ D_{\text{CLIP}}(G(w), t) + \lambda_{\text{L2}} \left\| w - w_s \right\|_2 + \lambda_{\text{ID}} \mathcal{L}_{\text{ID}}(w)$$

Output is close to the text — Close to the original latent — Output is close to input

StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery [Patashnik et al., ICCV 2021]