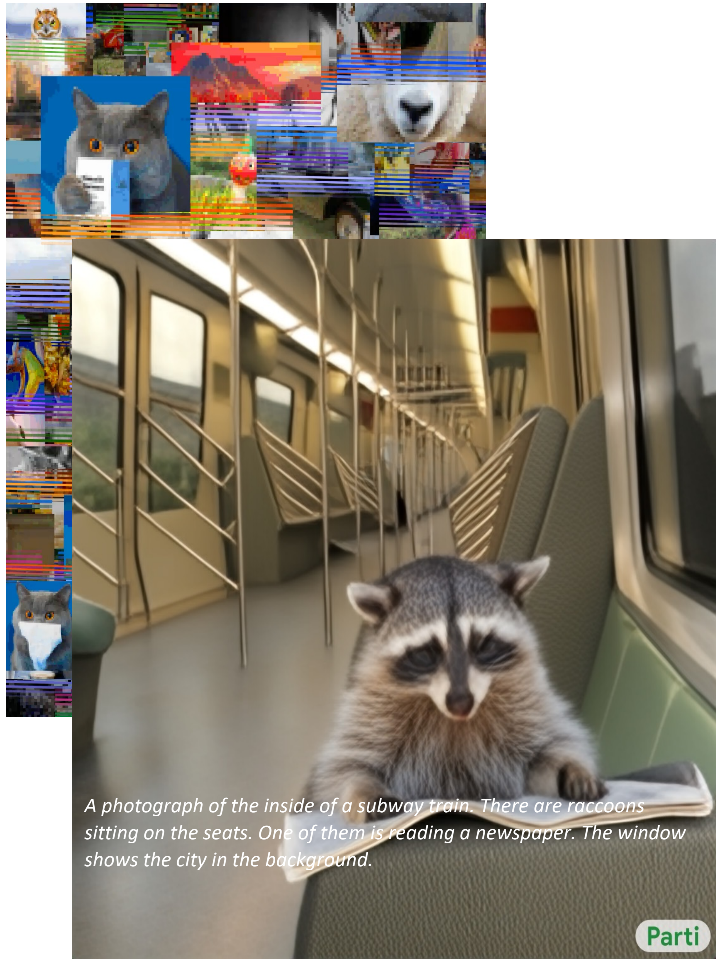
Image Editing with Optimization (part I)

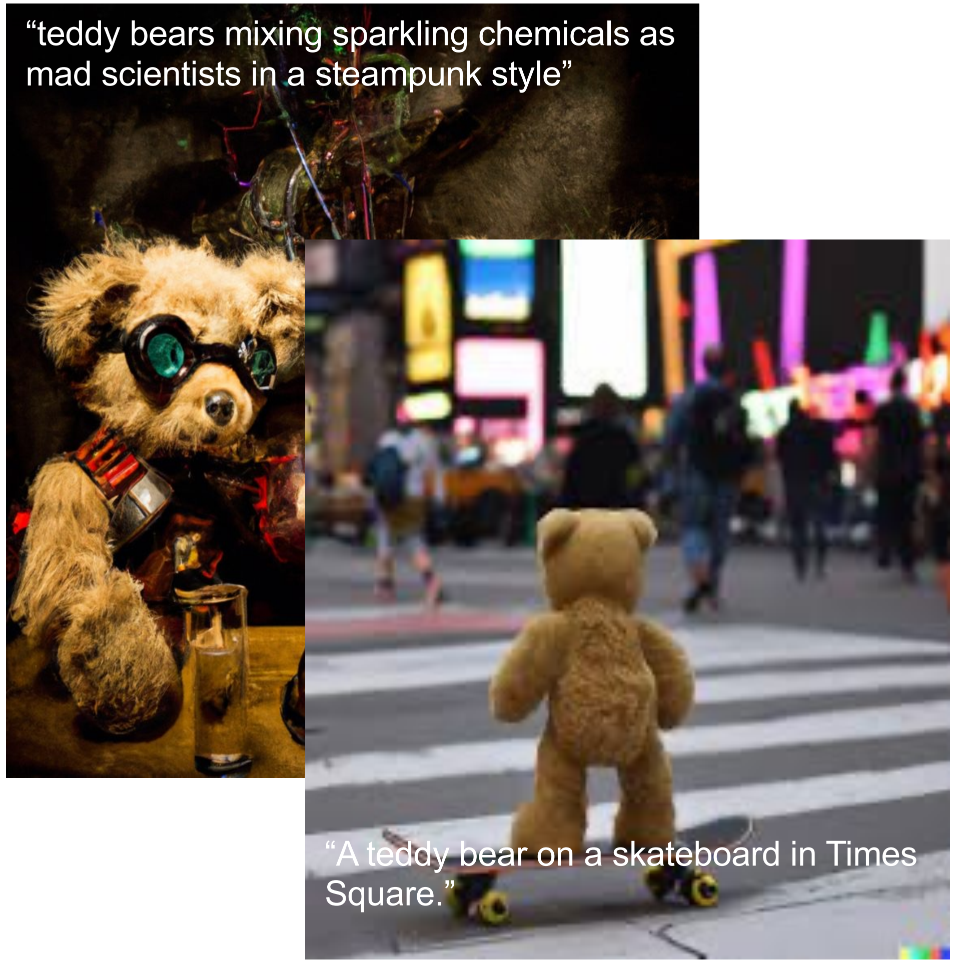
Jun-Yan Zhu

16-726, Spring 2023

Text-to-Image Everywhere



Autoregressive models
(Image GPT, Parti)



Diffusion models
(DALL-E 2, Imagen)

GANs, Masked GIT
(GigaGAN, MUSE)

How could we improve it?

- Better generative modeling techniques: VAEs, GANs, diffusion, AR, Hybrid
- Better text encoders: RNN/LSTM -> Transformers (CLIP, T5)
- Better generator architectures: RNN/LSTM -> CNN -> CNN + Transformer
- Better ways to connect text and image: concatenation -> AdaIN -> cross-attention
- More data + GPU/TPU computing: a few hundred A100.
- Bigger model sizes: 1B-20B.

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

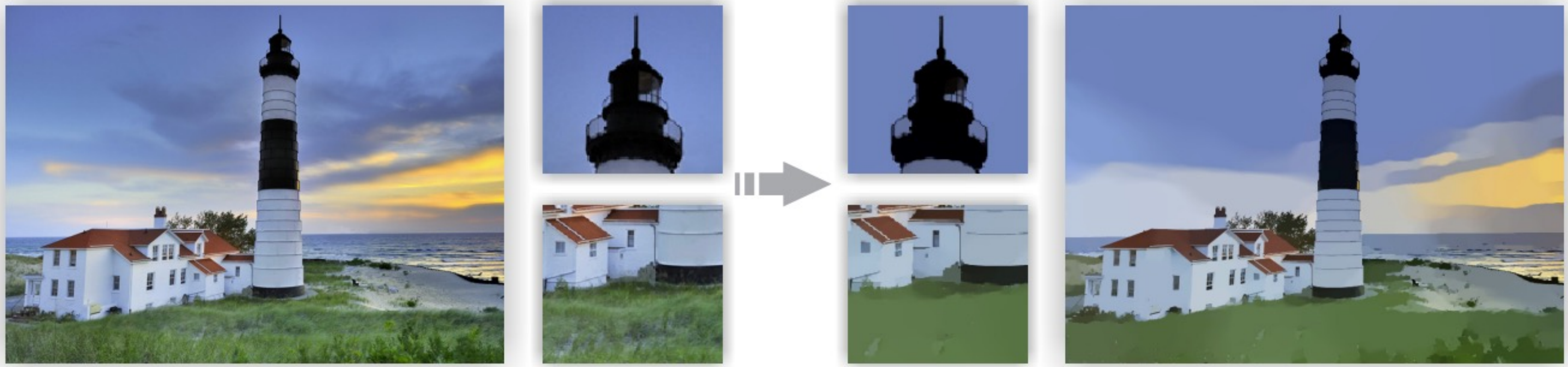


Image Smoothing via L0 Gradient Minimization [Xu et al., SIGGRAPH Asia 2011]

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

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

Image Editing with Optimization

Colorization using Optimization [Levin et al., SIGGRAPH 2004]

YUV color space (Y is fixed)
constant: scribbles
variables: rest of the pixels

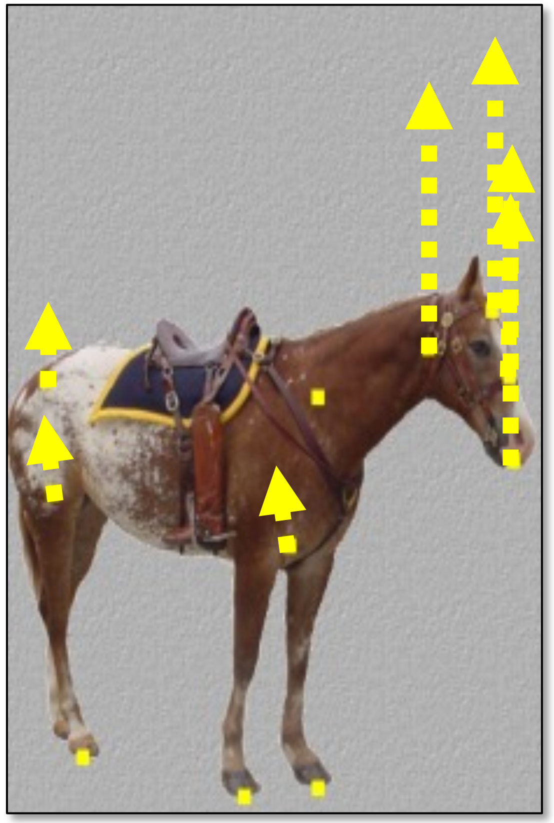
$$\sum_{\mathbf{r}} \left(U(\mathbf{r}) - \sum_{\mathbf{s} \in N(\mathbf{r})} w_{\mathbf{rs}} U(\mathbf{s}) \right)^2$$

visual similarity between r and s
Intensity, location, edge, motion, etc.

the color of pixel r

the color of pixel s (s is r's neighbor)

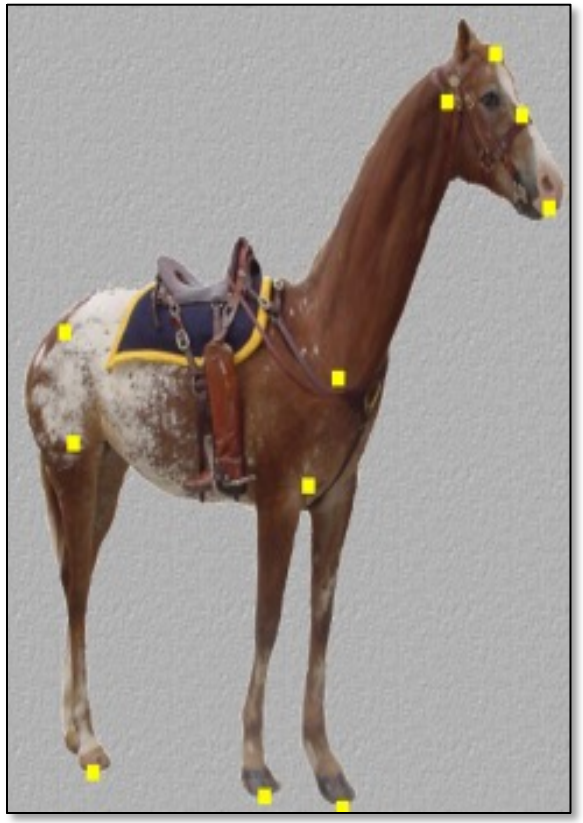
Image Editing with Optimization



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

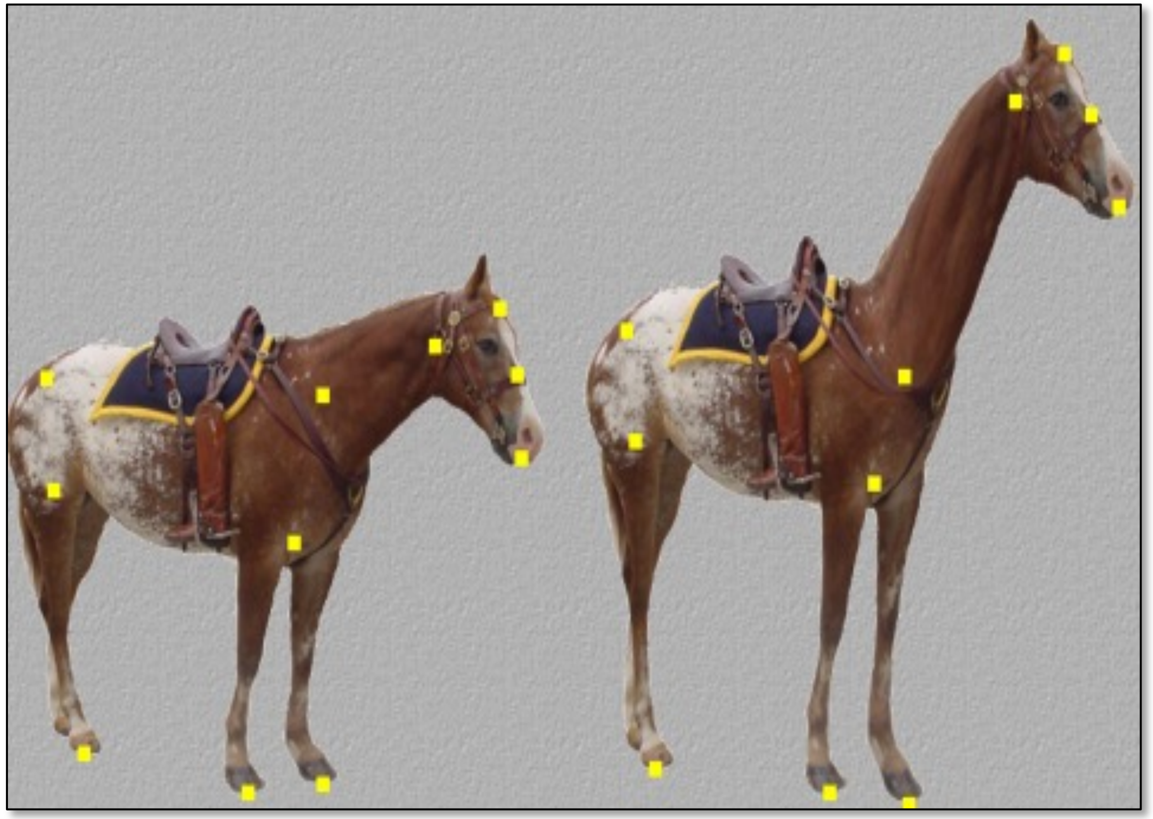


Image Warping

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:

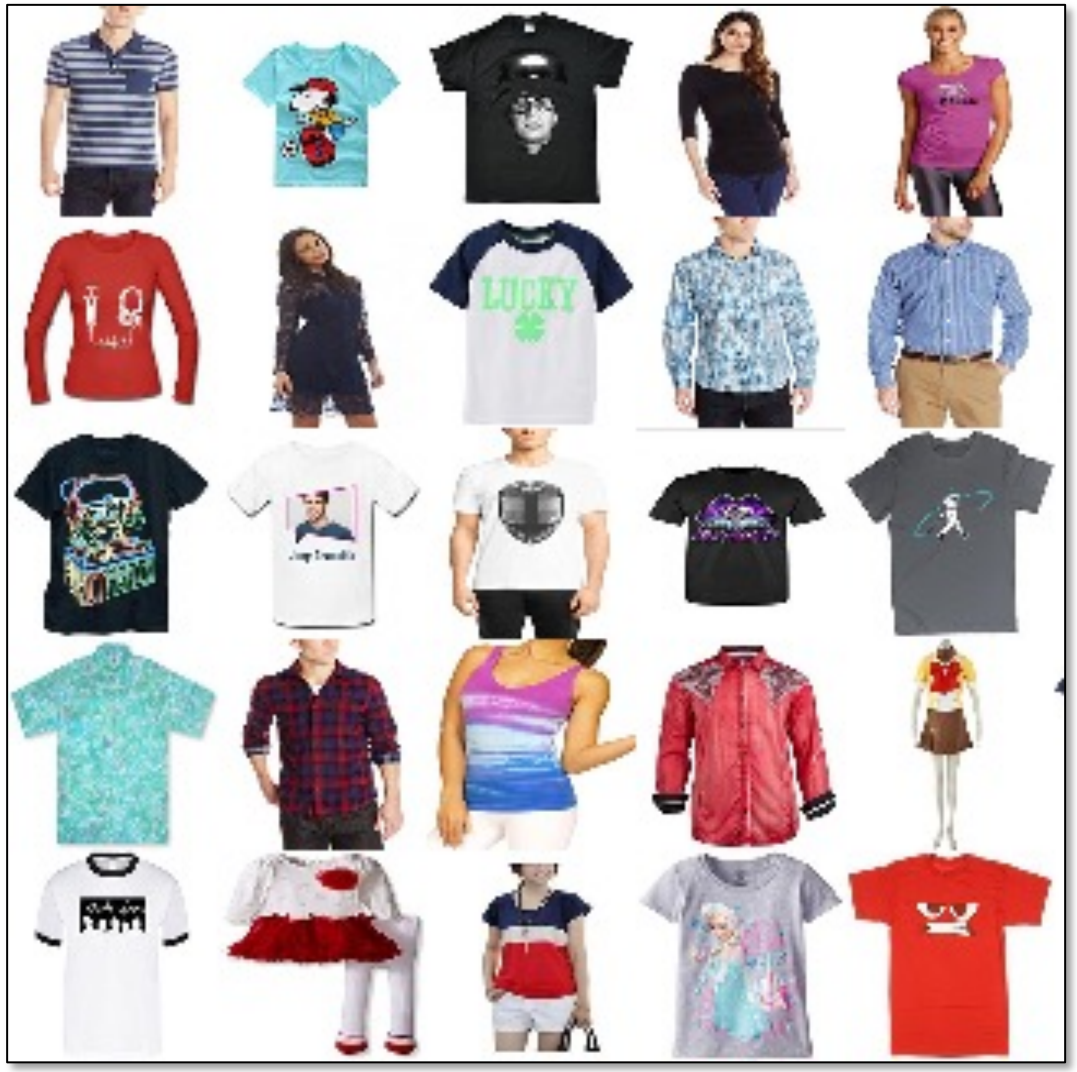
- stay close to the input.
- satisfy user's constraint.
- Lie on the natural image manifold

Learning Natural Image Manifold

- Deep generative models: $G(z) : z \rightarrow x$
  - 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)

  …

GAN as Manifold Approximation



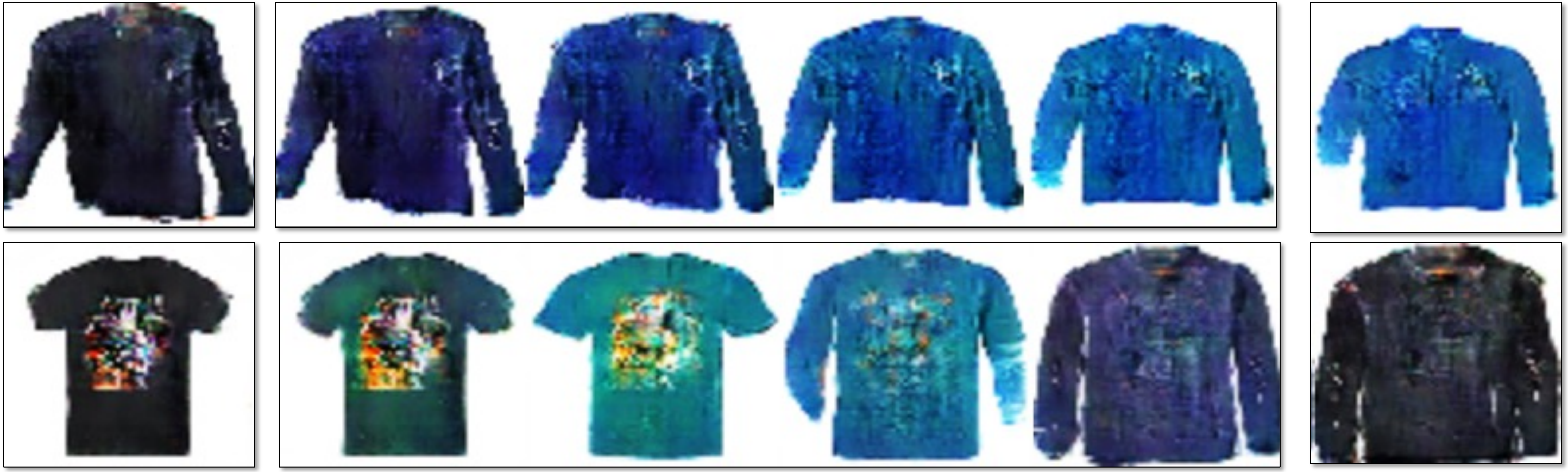
Sample training images
from "Amazon Shirts"

Random image samples
from Generator $G(z)$

DCGAN [Radford et al. 2015]

Traverse on the GAN Manifold

$G(z_0)$ Linear Interpolation in z space: $G(z_0 + t \cdot (z_1 - z_0))$ $G(z_1)$



Limitations of DCGAN:
- not photo-realistic enough, low resolution
- produce images randomly, no user control

DCGAN [Radford et al. 2015]

Changing Variables

- Traditional method: Optimizing the image

user constraint

$$\hat{y}^* = \arg\min_{\hat{y}} \mathcal{L}(x, y, \hat{y})$$

input output

- New method: Optimizing the latent code

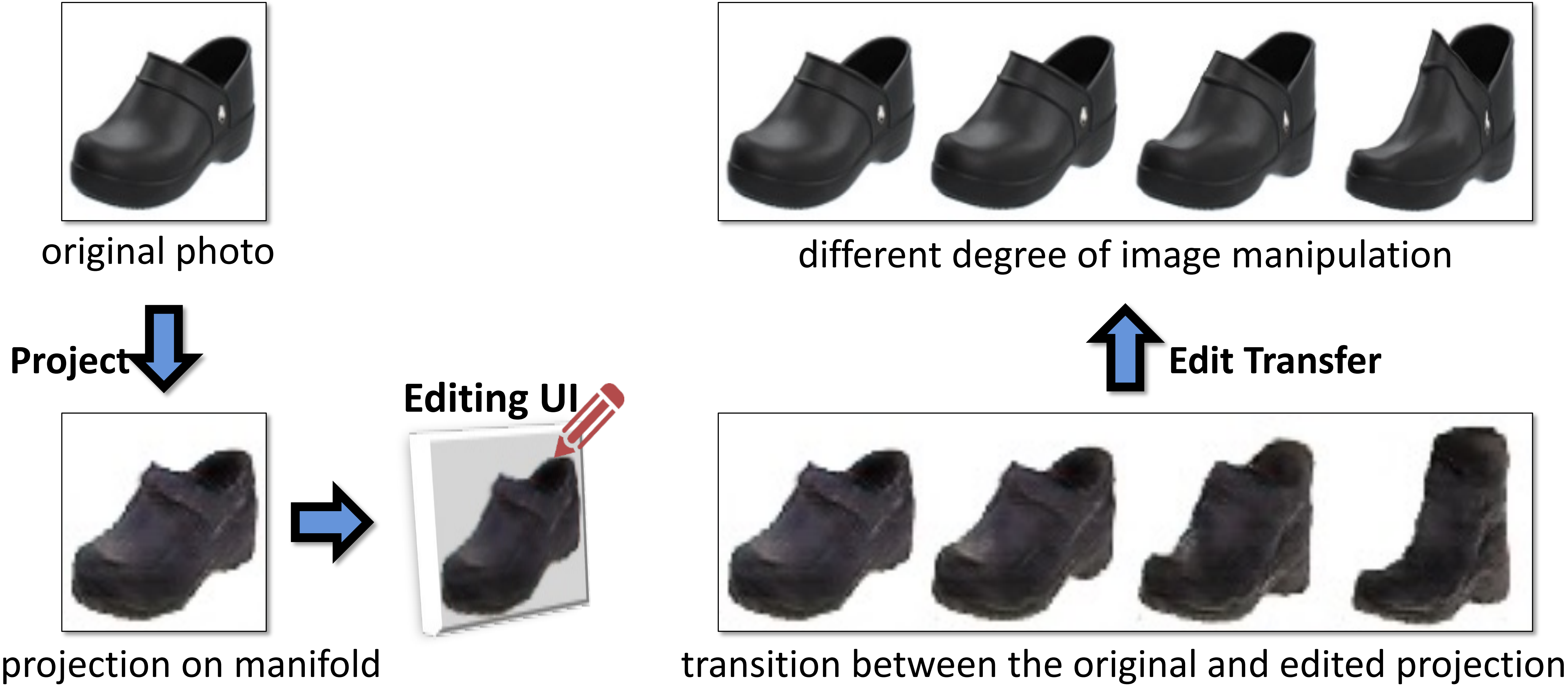
user constraint

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

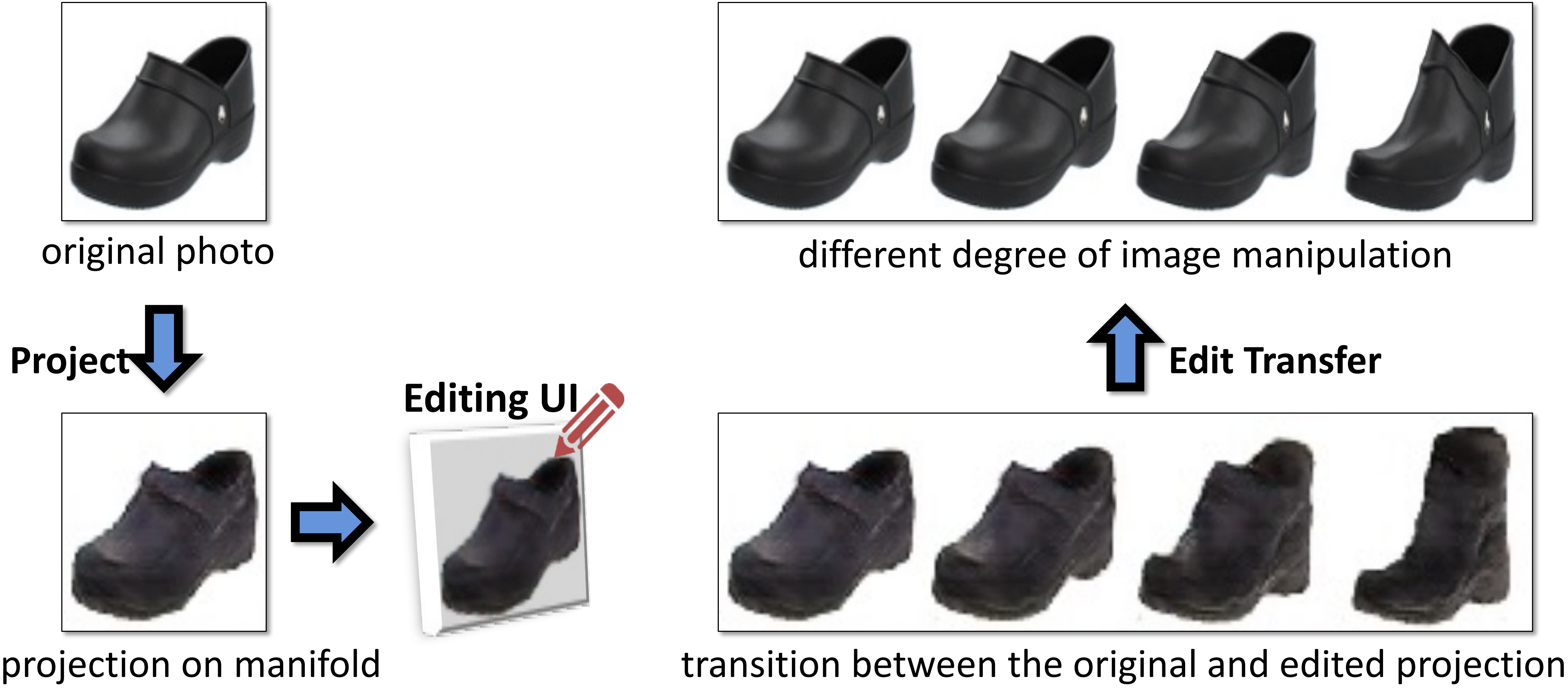
input Latent code

Generator

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

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

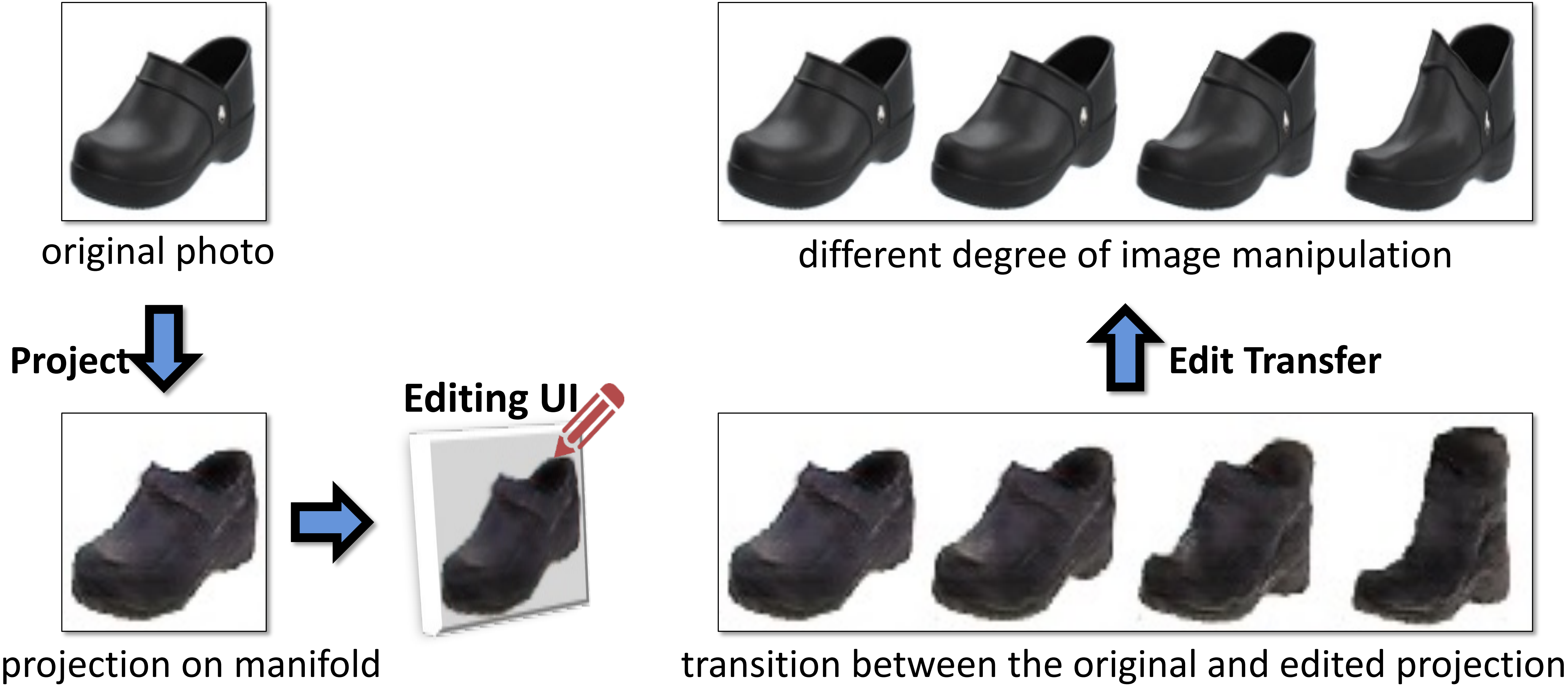
Inverting Network $z = E(x)$

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

Hybrid Method

Use the network as initialization for the optimization problem

Manipulating the Latent Code



Manipulating the Latent Code

$\boxed{\text{constraint violation loss } L_g}$

$\boxed{\text{user guidance image}}$

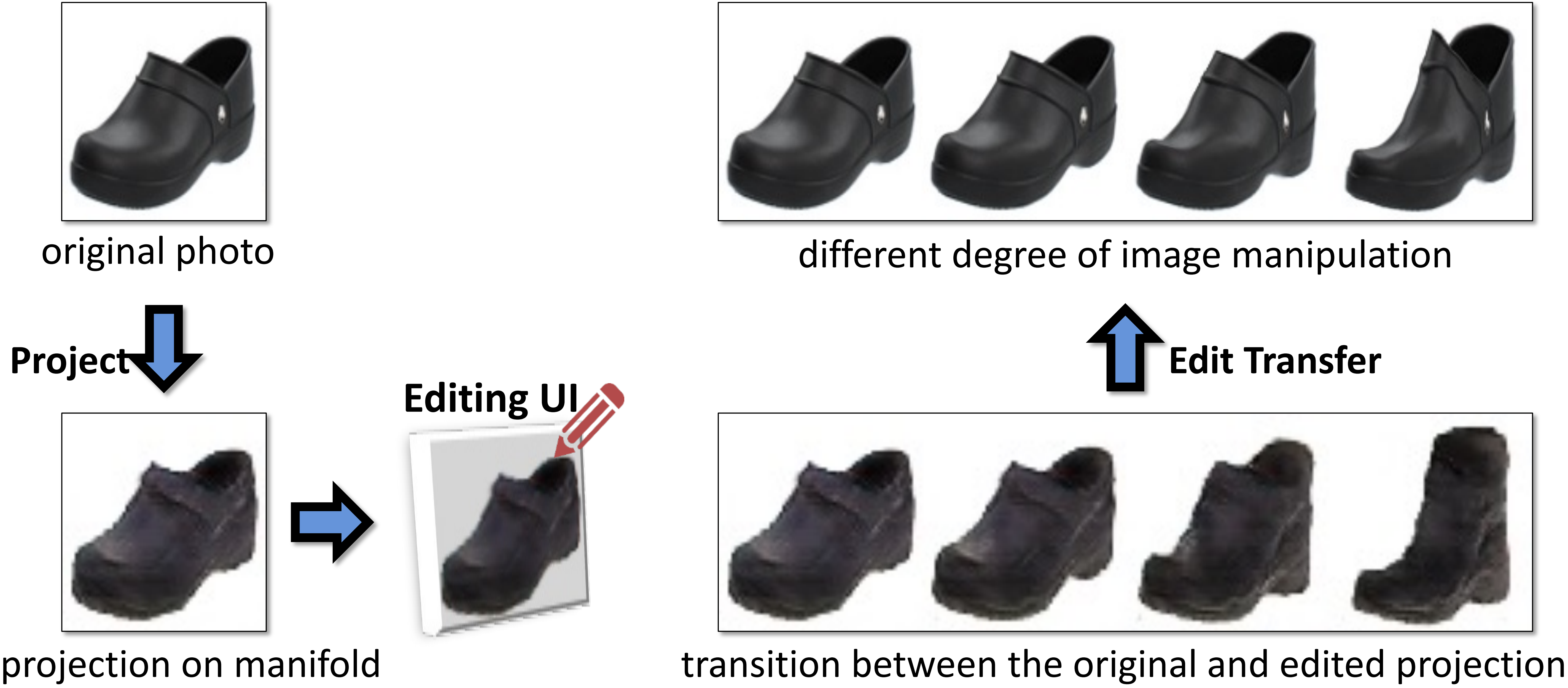
Objective: $$z^* = \arg\min_{z\in\mathbb{Z}} \Big\{ \underbrace{\sum_g (\mathcal{L}_g(G(z), v_g)}_{\text{data term}} + \underbrace{\lambda_s \cdot \|z - z_0\|_2^2}_{\text{manifold smoothness}} \Big\}.$$

Guidance v_g

$G(z)$

z_0

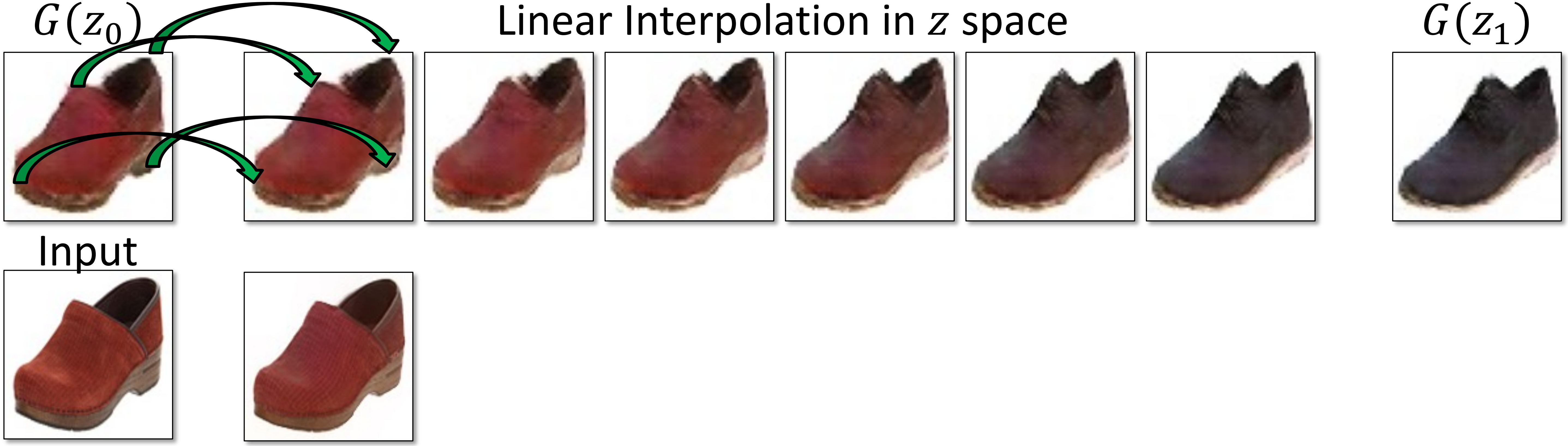
Post-Processing



Edit Transfer

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

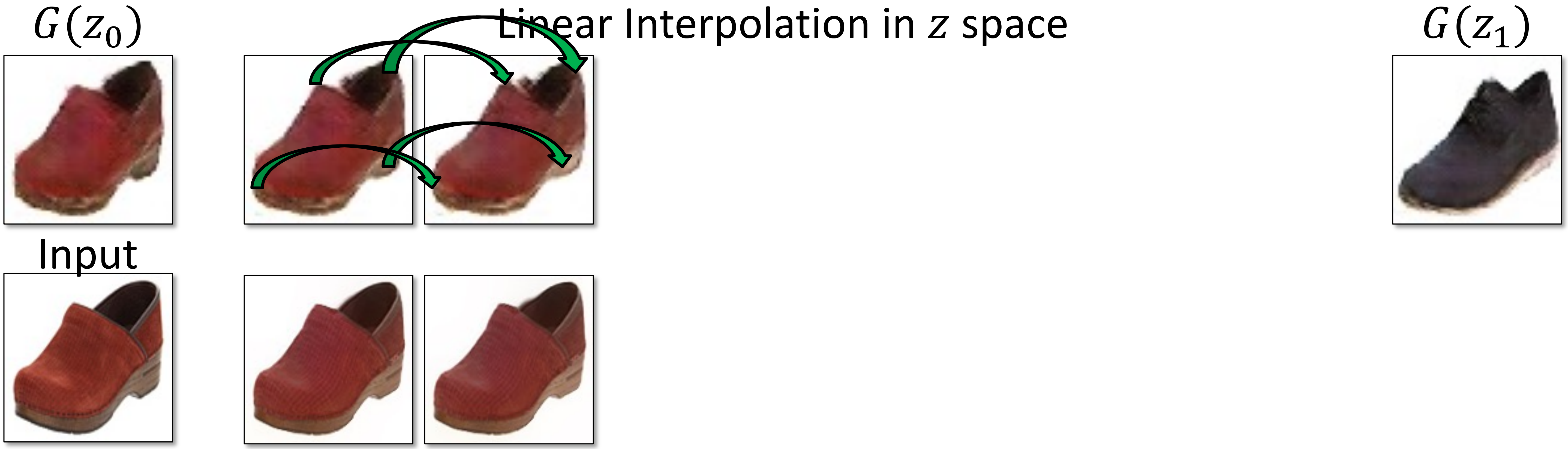
$$\iint \underbrace{\|I(x,y,t) - \boxed{A} \cdot I(x+\boxed{u}, y+\boxed{v}, t+1)\|^2}_{\text{data term}} + \underbrace{\sigma_s(\|\nabla u\|^2 + \|\nabla v\|^2)}_{\text{spatial reg}} + \underbrace{\sigma_c \|\nabla A\|^2}_{\text{color reg}} dxdy$$



Edit Transfer

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

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



Edit Transfer

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

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

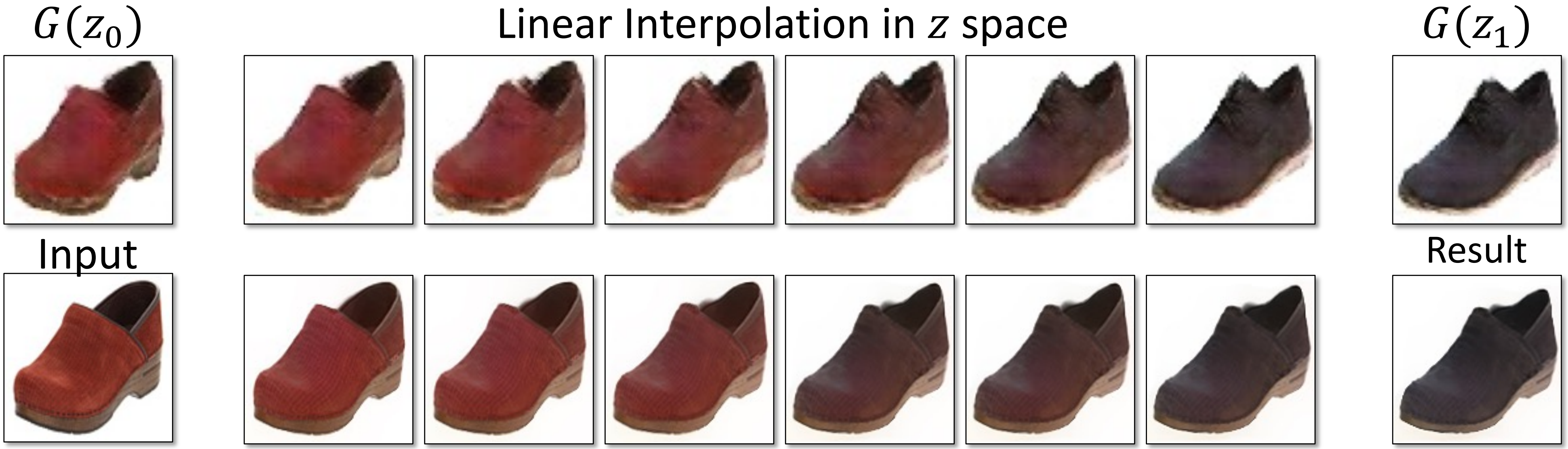


Image Manipulation Demo

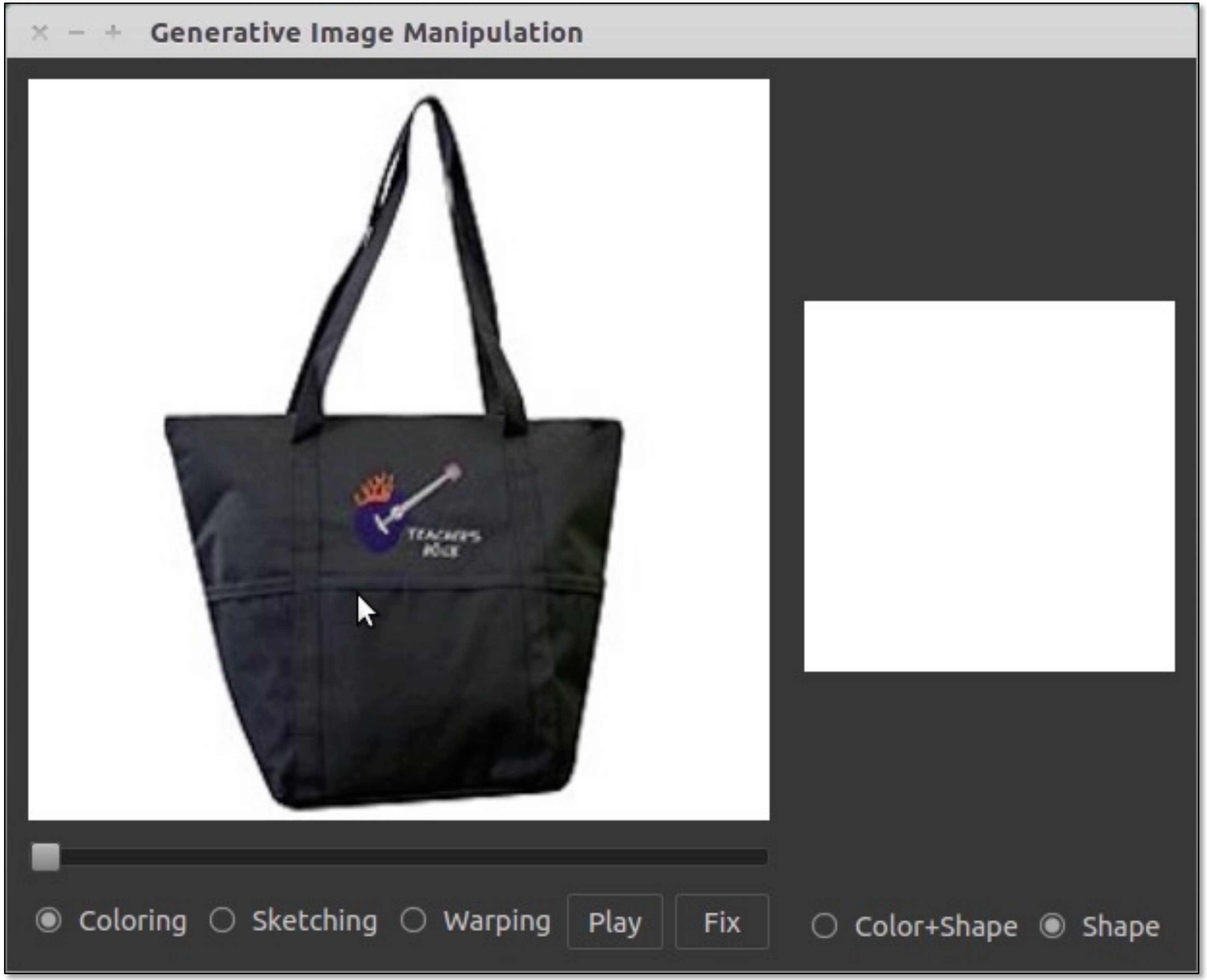
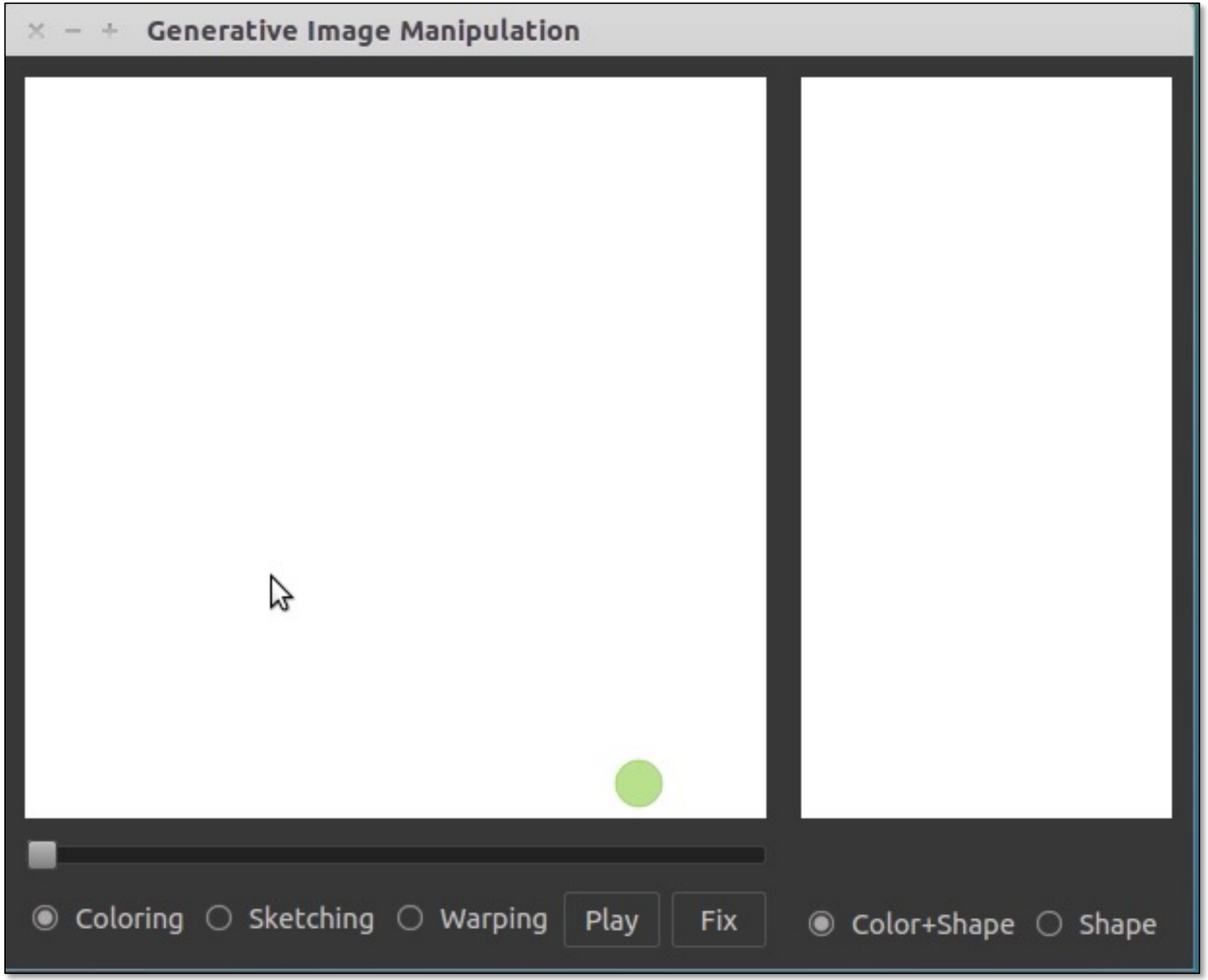
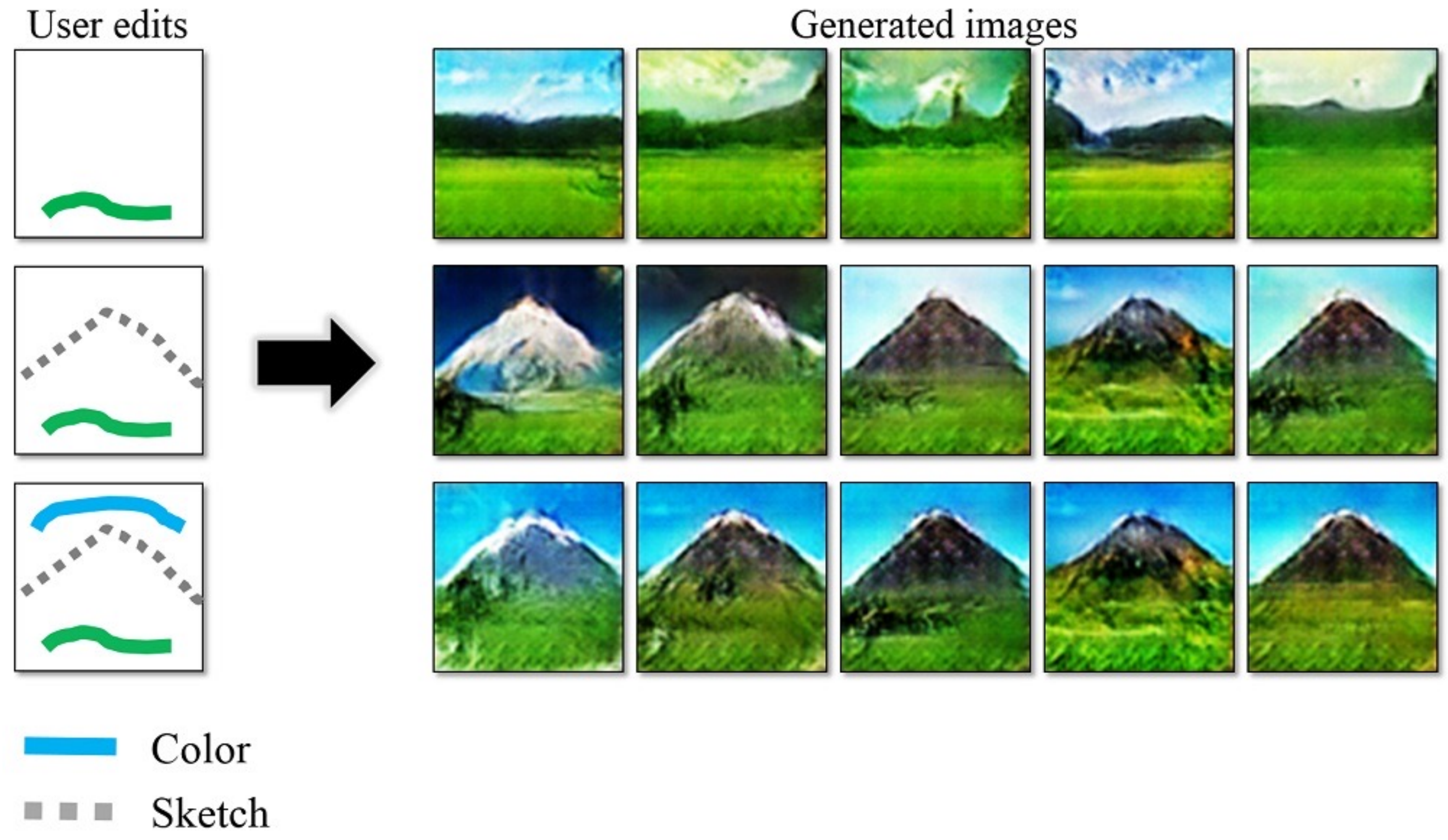


Image Manipulation Demo

Interactive Image Generation





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

Changing Variables

- Traditional method: Optimizing the image

user constraint

$$\hat{y}^* = \arg\min_{\hat{y}} \mathcal{L}(x, y, \hat{y})$$

input result

- New method: Optimizing the latent code

user constraint

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

input Latent code

Generator

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

$$G(z_1)$$

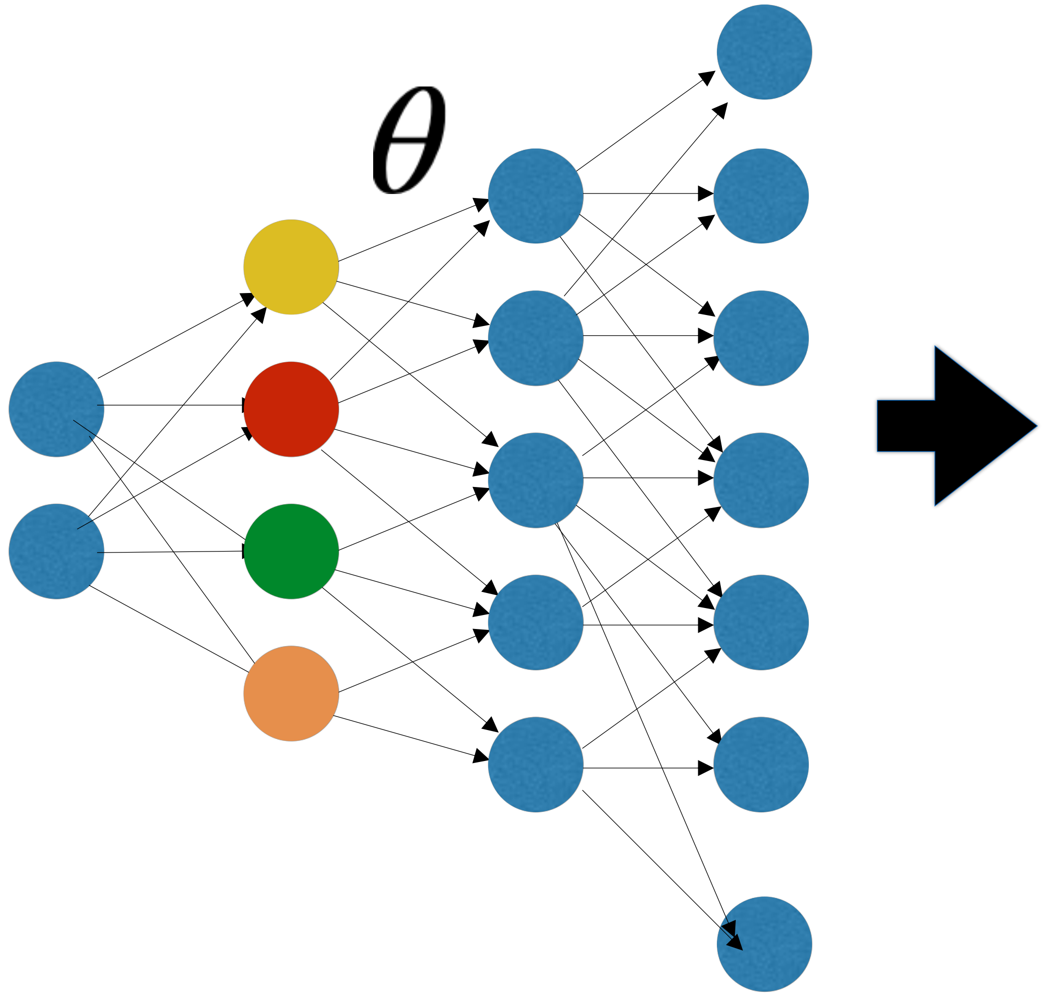
Image Projection with GANs

Image Reconstruction (high-res images, Big Models)

Original image x

Image Reconstruction (high-res images, Big Models)

Random vector z ?



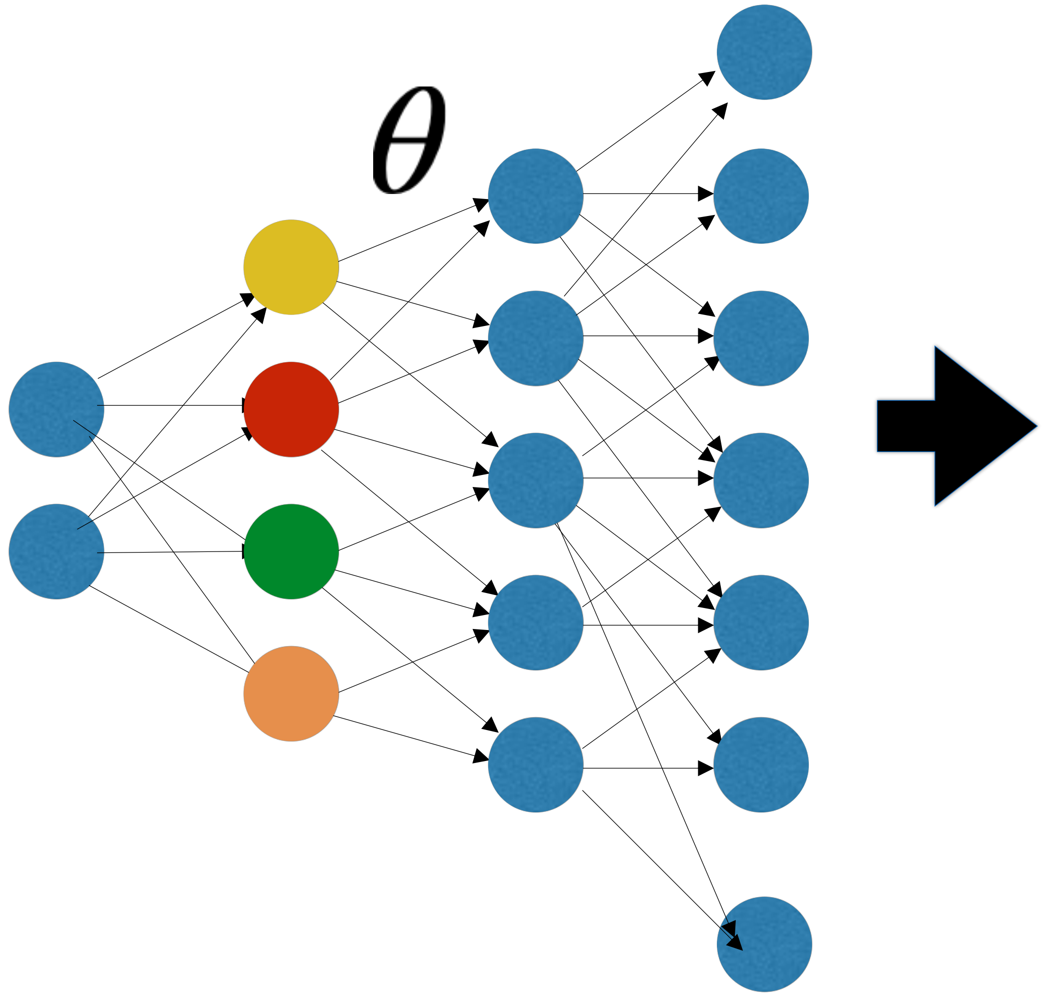
Original image x

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

iGAN [Zhu et al. 2016]

Image Reconstruction (high-res images, Big Models)

Random vector z^*



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...

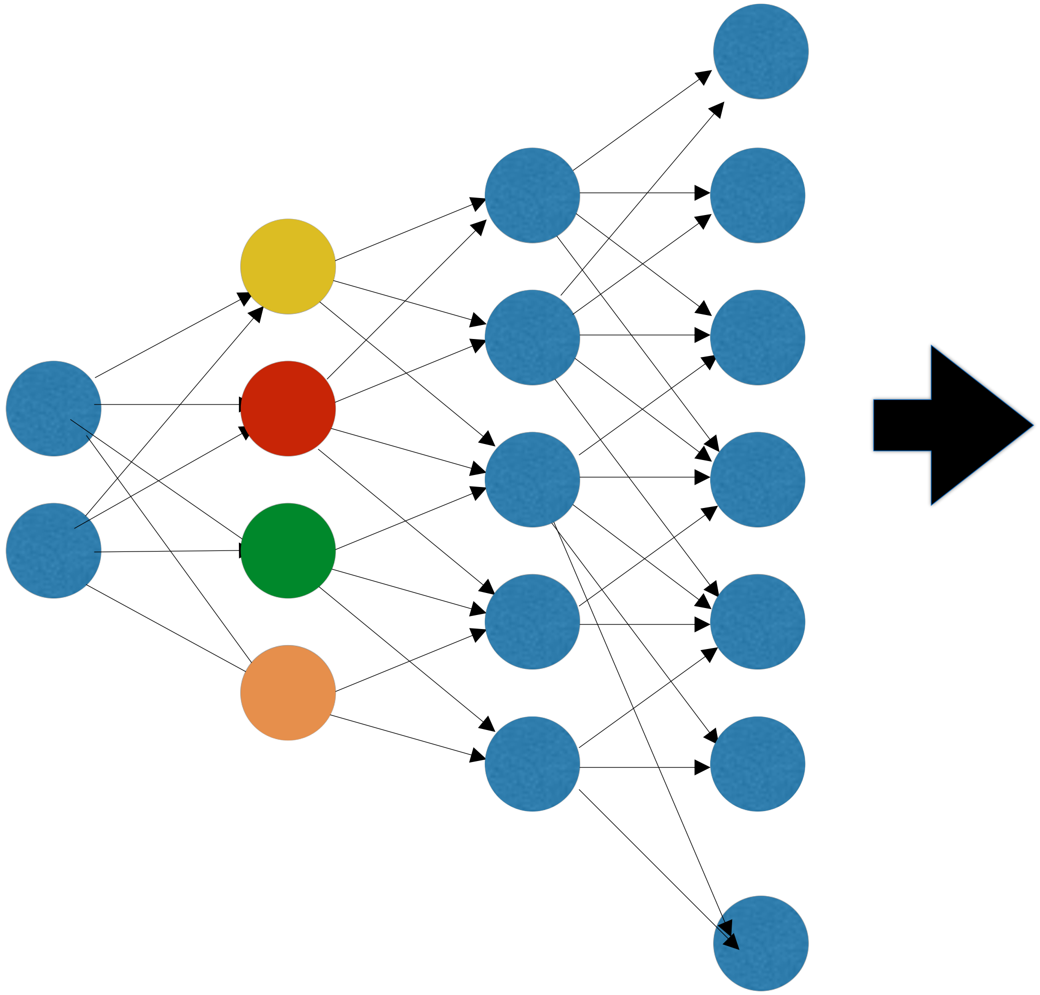
Original image

GAN reconstructed image

Original image

Reconstructing a Real Photo

Random vector z^*



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

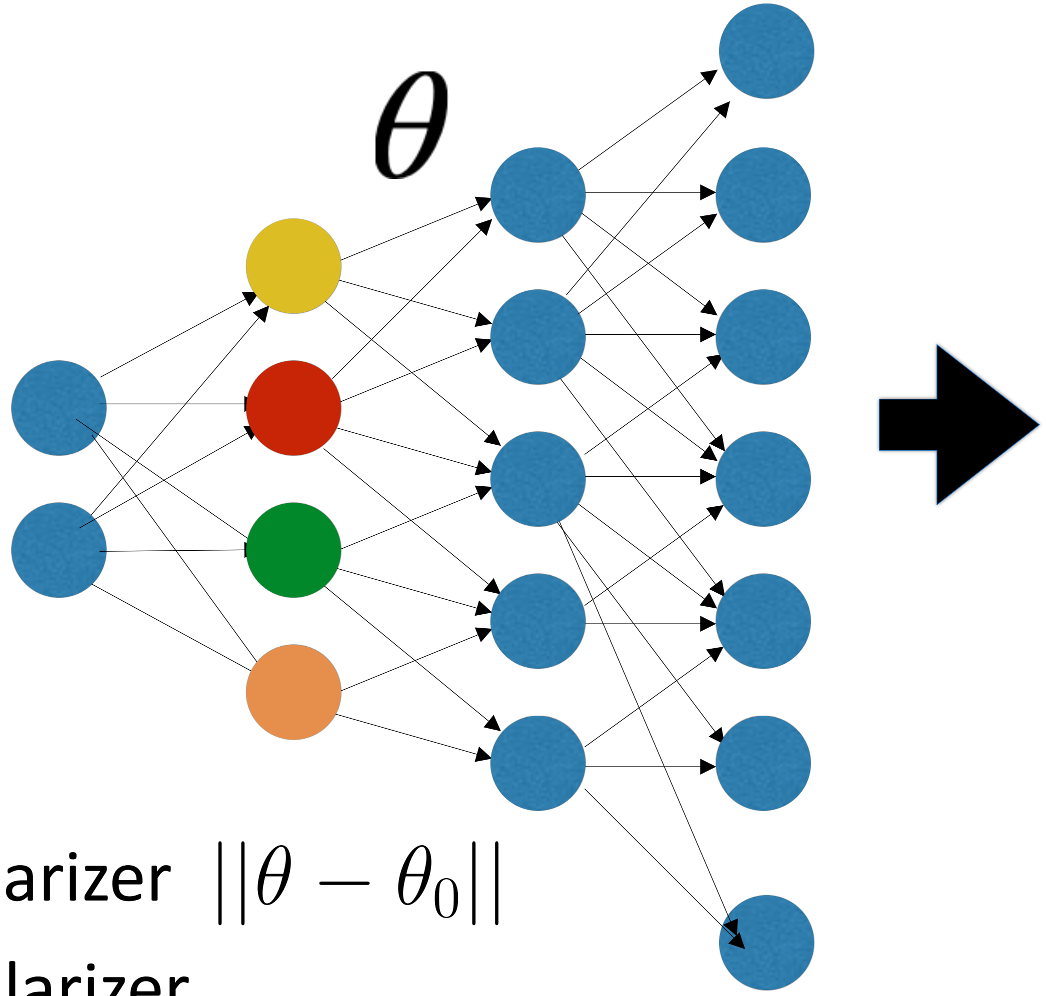
$$z^* = \arg\min_z \mathcal{L}(G(z;\theta), x)$$

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

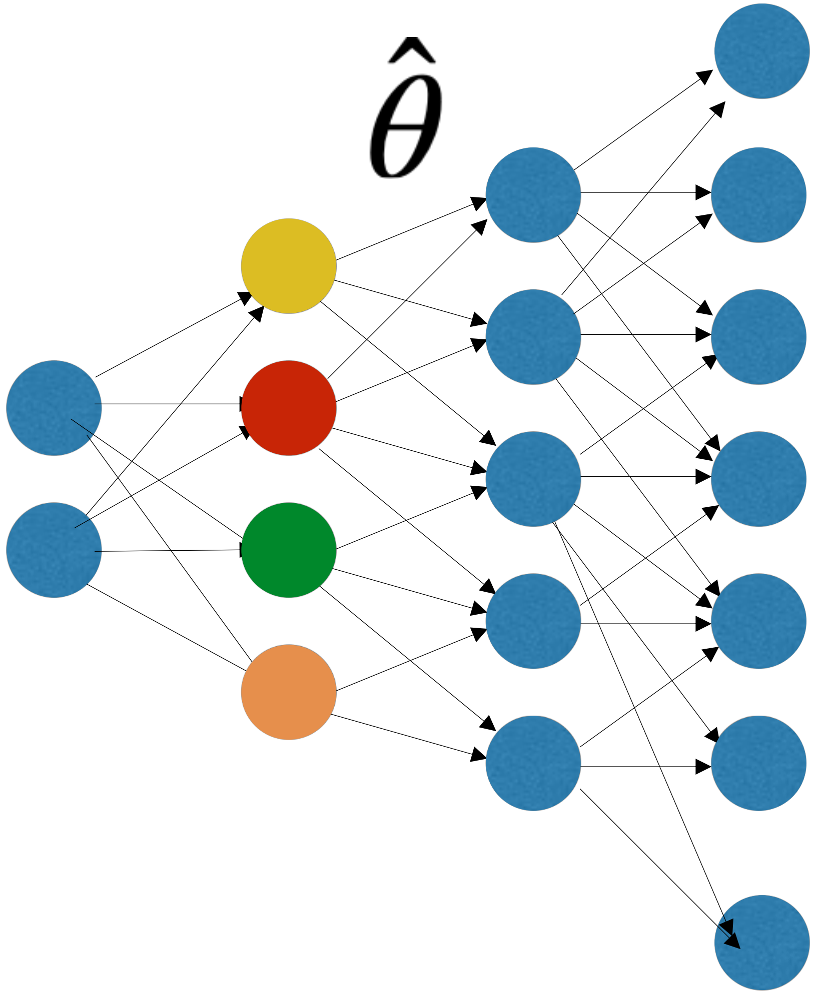
$$z^*, \theta^* = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x)$$

⟵ Regularizer

Reconstructing a Real Photo

Original image

Random vector z^*



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

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

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.]