Image Editing with Optimization (part II)

Jun-Yan Zhu

16-726, Spring 2022

© StyleCLIP [Patashnik et al. 2021]

Image Editing with Optimization

input photo

Image Editing Program

User Input

result

Desired output:

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

Photo: [Tao et al. 2014]

Image Editing with Optimization

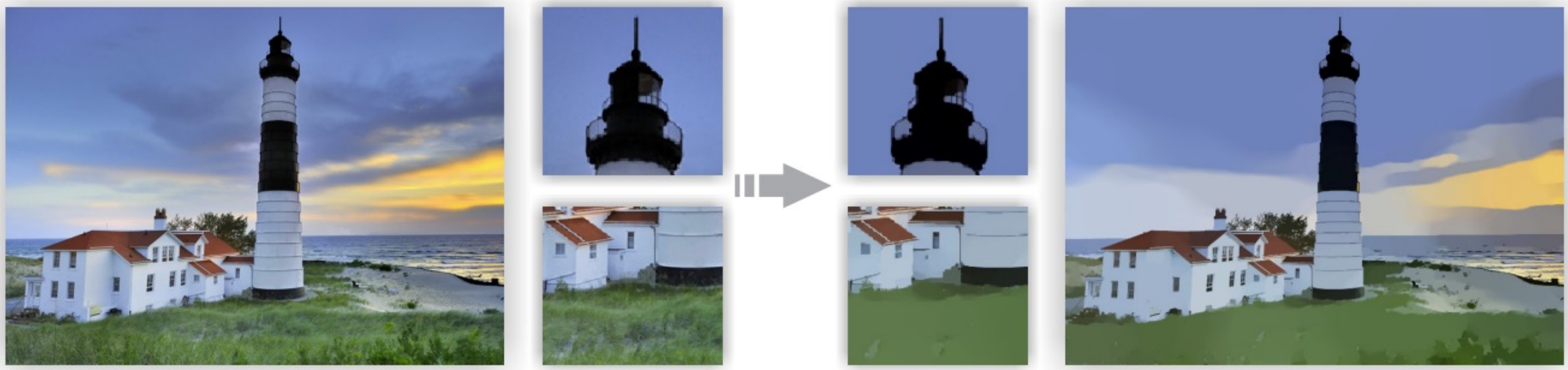


Image Smoothing via L0 Gradient Minimization [Xu et al., SIGGRAPH Asia 2011]

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

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

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

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

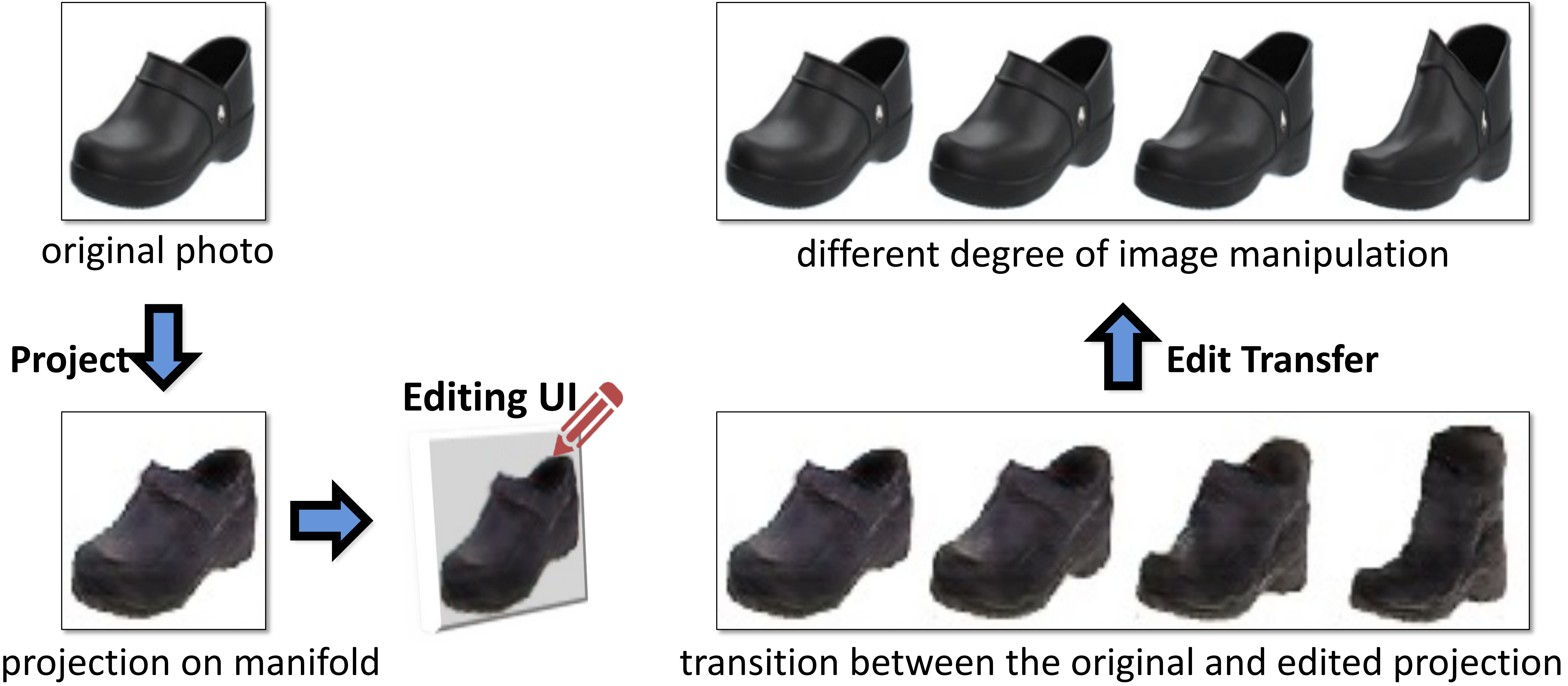
user constraint

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

input Latent code

Generator

Projecting and Editing an Image



Projecting an Image into GAN Manifold

Input: real image x

Output: latent vector z

Optimization

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

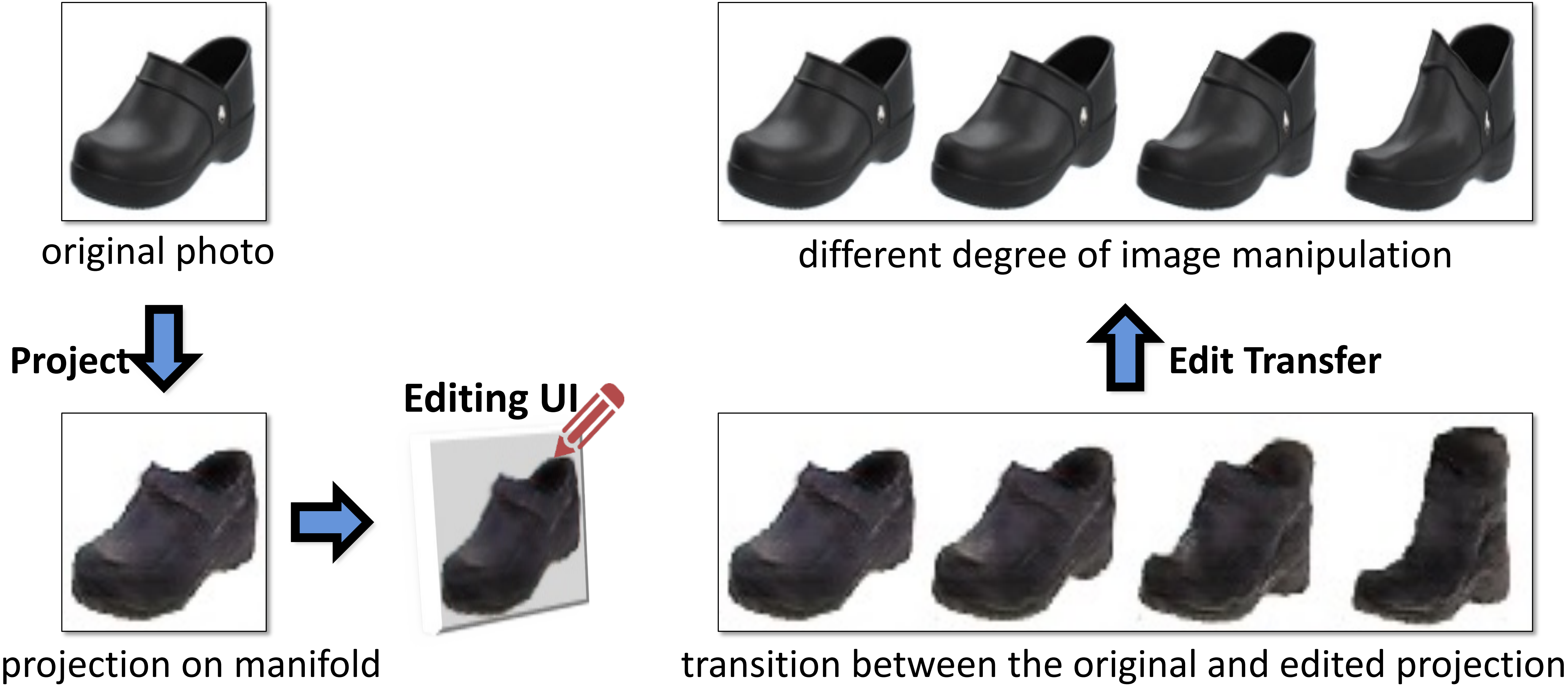
Inverting Network $z = E(x)$

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

Hybrid Method

Use the network as initialization for the optimization problem

Manipulating the Latent Code



Post-Processing

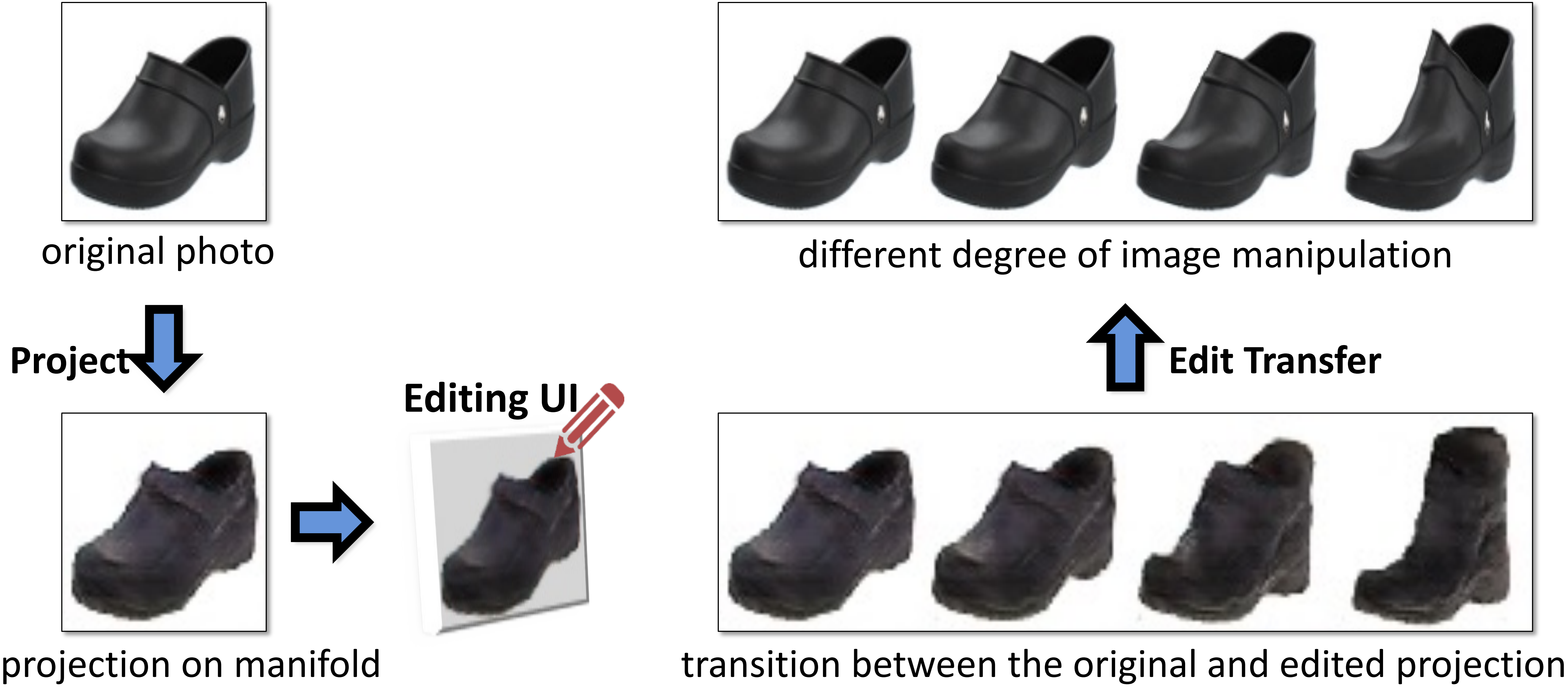


Image Editing with GANs

- 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

Baseline

- Baseline: Optimizing the latent code

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

z^* and z_0 are used interchangeably

Find the Differences…

Original image

GAN reconstructed image

Baseline

- Baseline: Optimizing the latent code

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

How to Improve GANs Projection

- Baseline: Optimizing the latent code

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

- Generator fine-tuning:

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

Generator Fine-tuning (Progressive GANs)

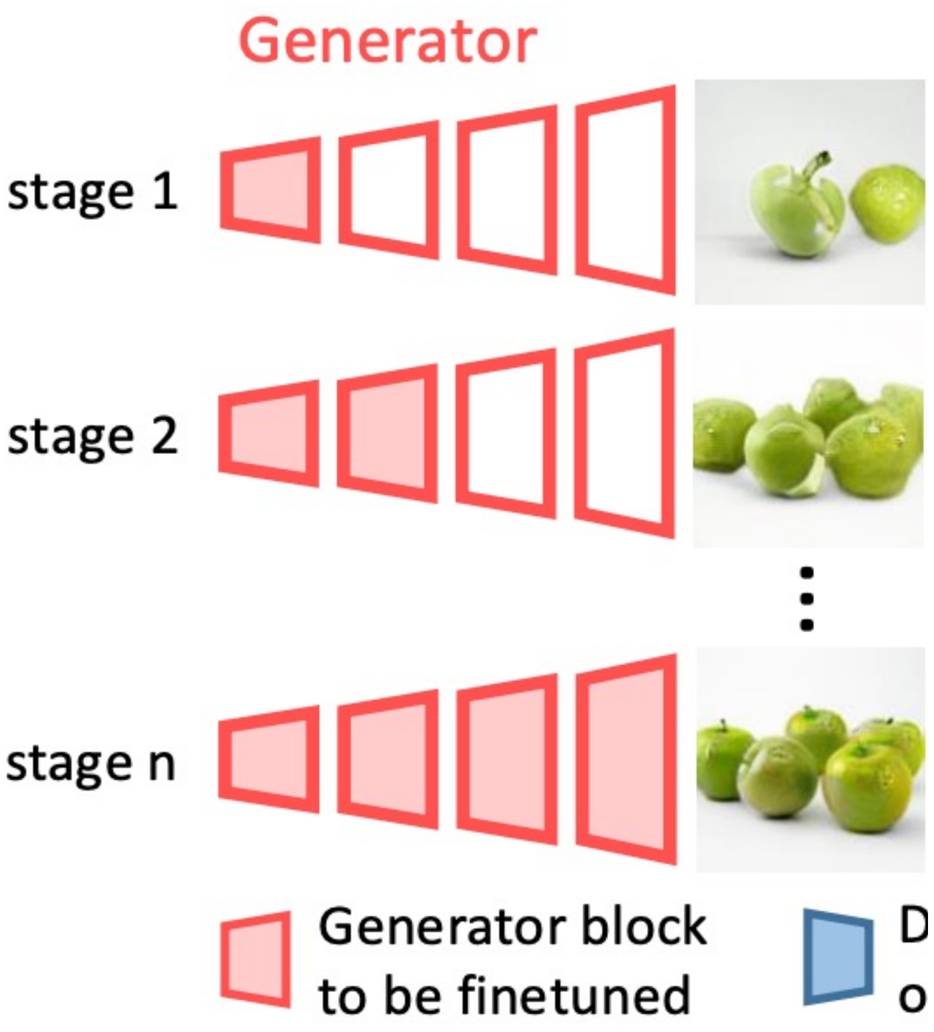
Original image

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.]

Generator Fine-tuning (BigGAN)

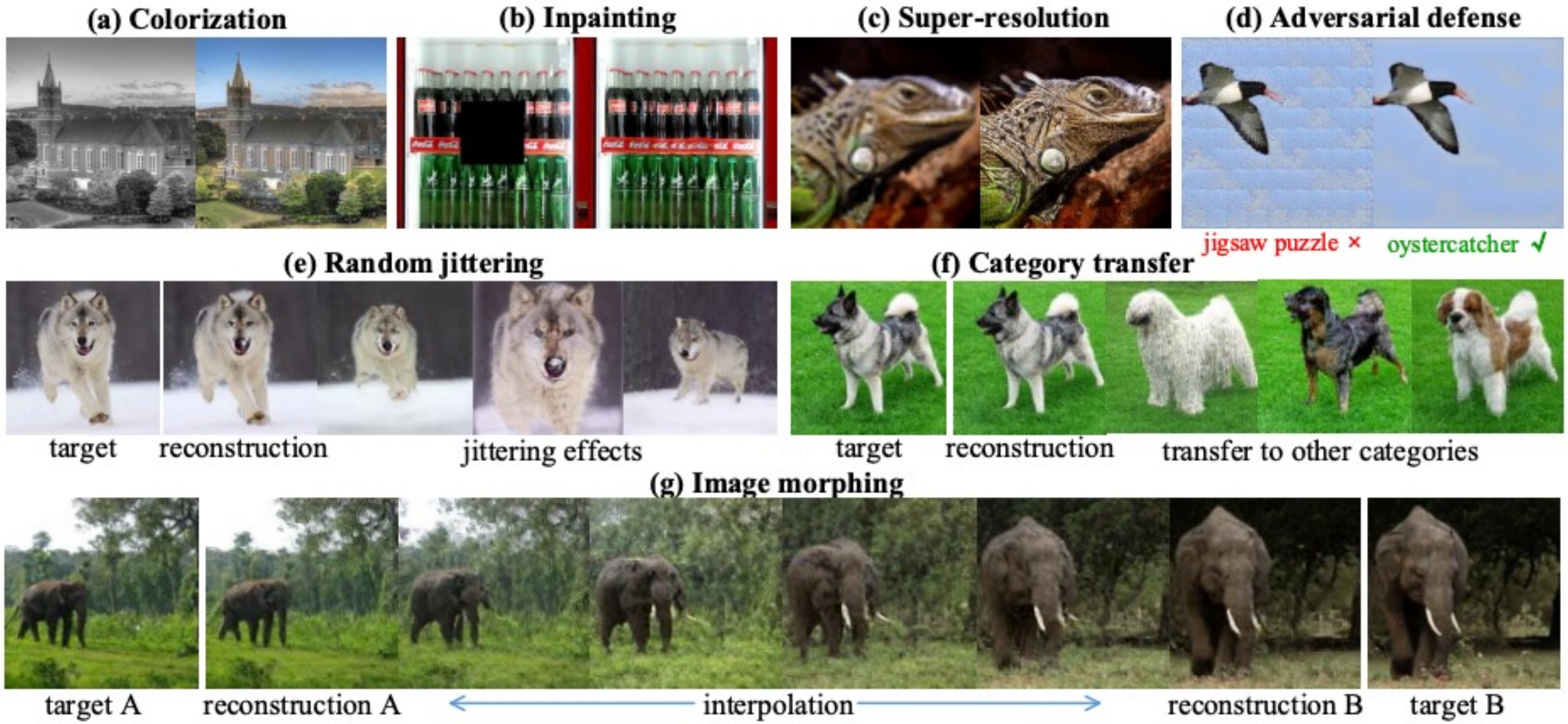


Progressive Reconstruction

- First match semantics
- Then match color and textures

Exploiting Deep Generative Prior for Versatile Image Restoration and Manipulation
[Pan et al., ECCV 2020]

Generator Fine-tuning (BigGAN)



Exploiting Deep Generative Prior for Versatile Image Restoration and Manipulation
[Pan et al., ECCV 2020]

How to Improve GANs Projection

- Baseline: Optimizing the latent code

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

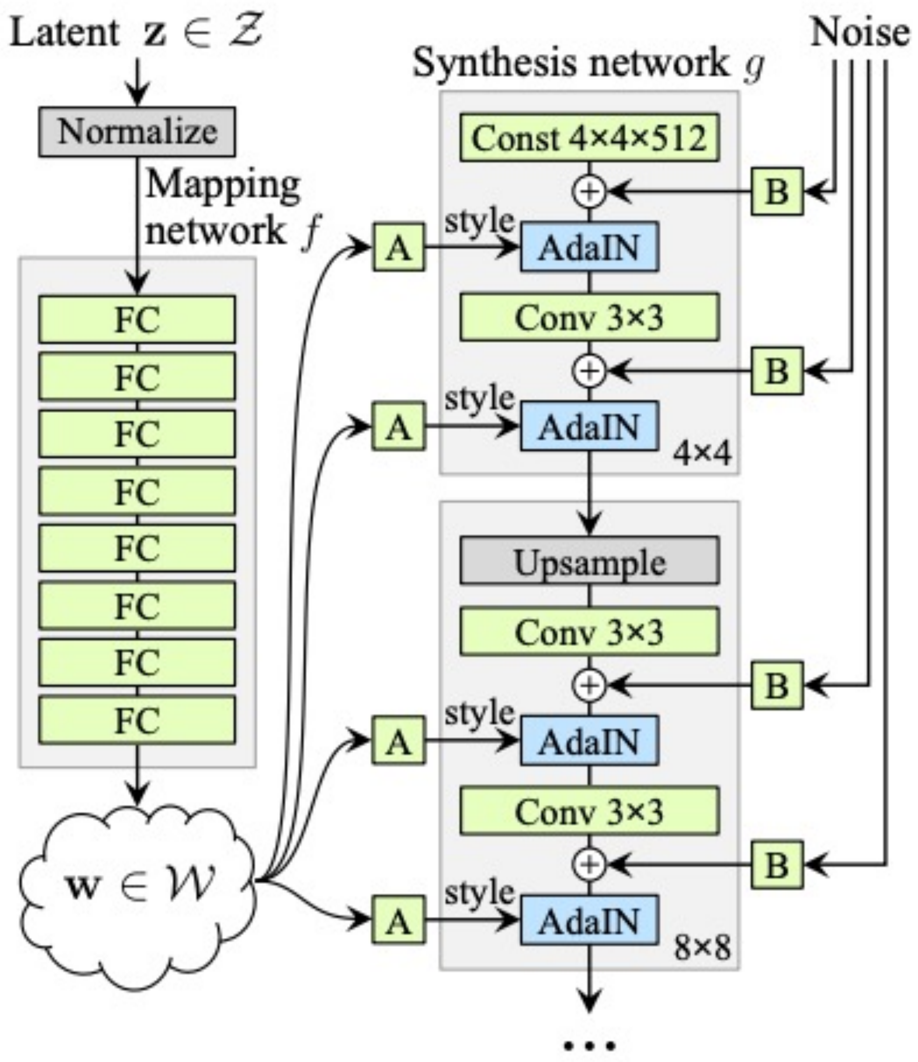
- Generator fine-tuning:

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

- Optimizing intermediate features

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

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

Input

Reconstruction

StyleGAN2 [Karras et al., 2019]

Using Different Layers: w+ Space

All the results are reconstructed via the StyleGAN Face model.

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

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

Used together

Generator Fine-tuning with w+ Space

Pivotal Tuning for Latent-based Editing of Real Images [Roich et al., 2021]

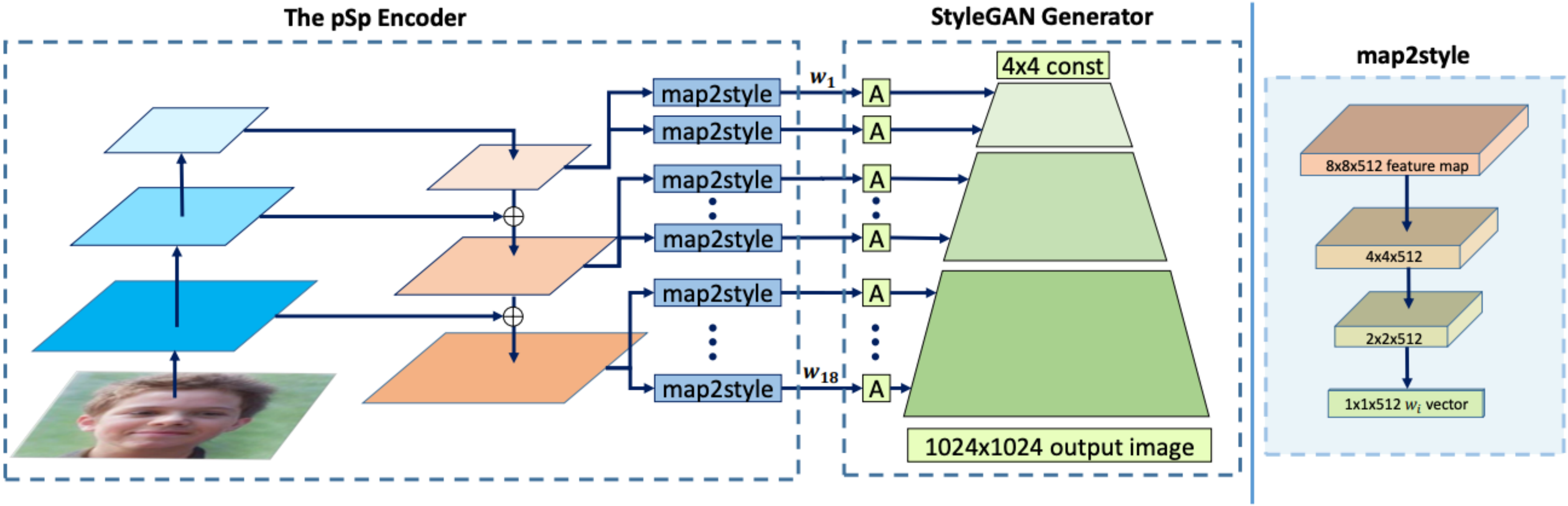
How to Improve GANs Projection

- Baseline: Optimizing the latent code

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

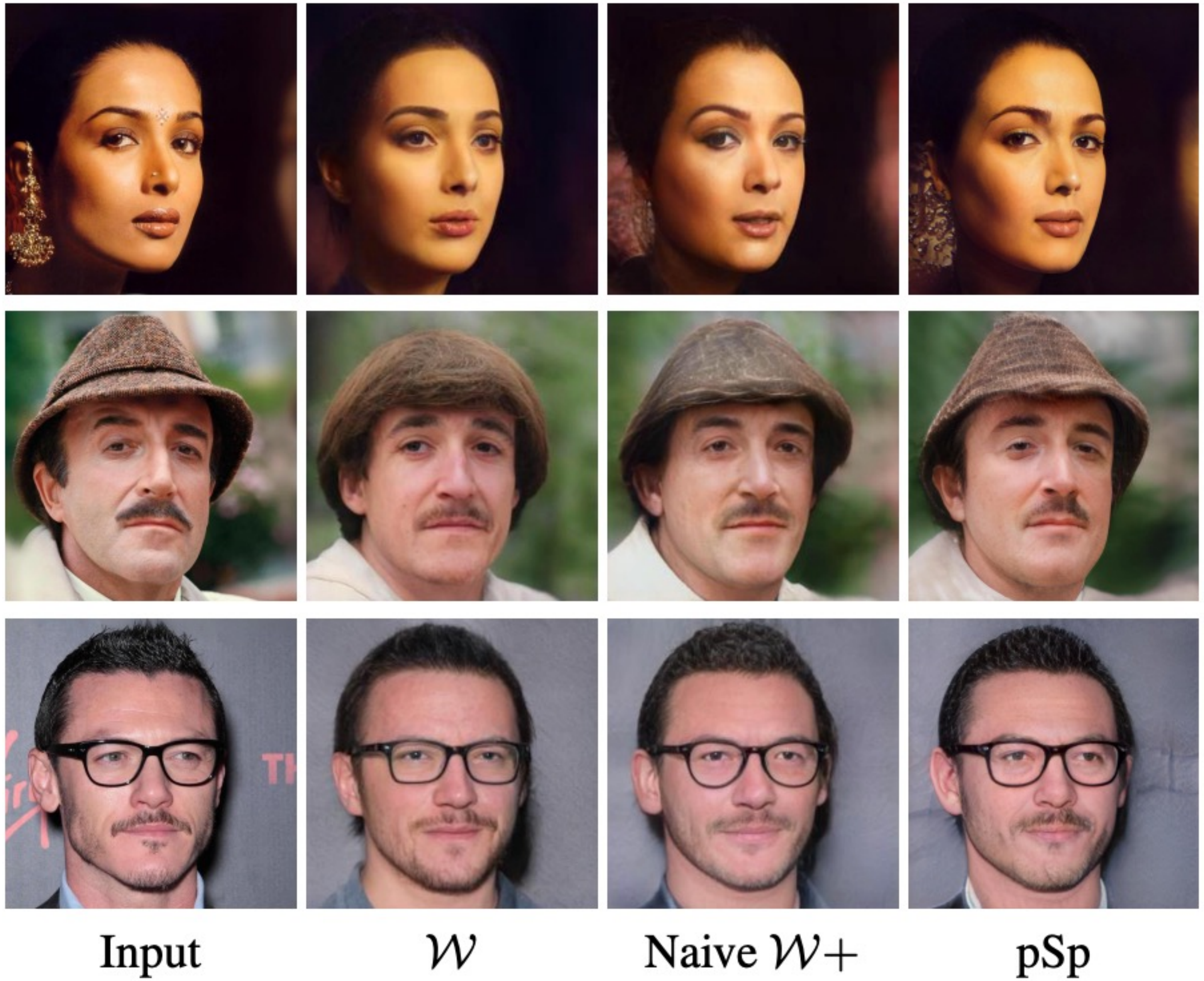
- Training an encoder E(x). Advantages?
  - Faster inference
  - More reliable initialization
- Encoder design depends on
  - Generator architecture.
  - Which latent space: z, w, w+.
  - Pre-trained network weights.

Example: An StyleGAN Encoder



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

Example: An StyleGAN Encoder



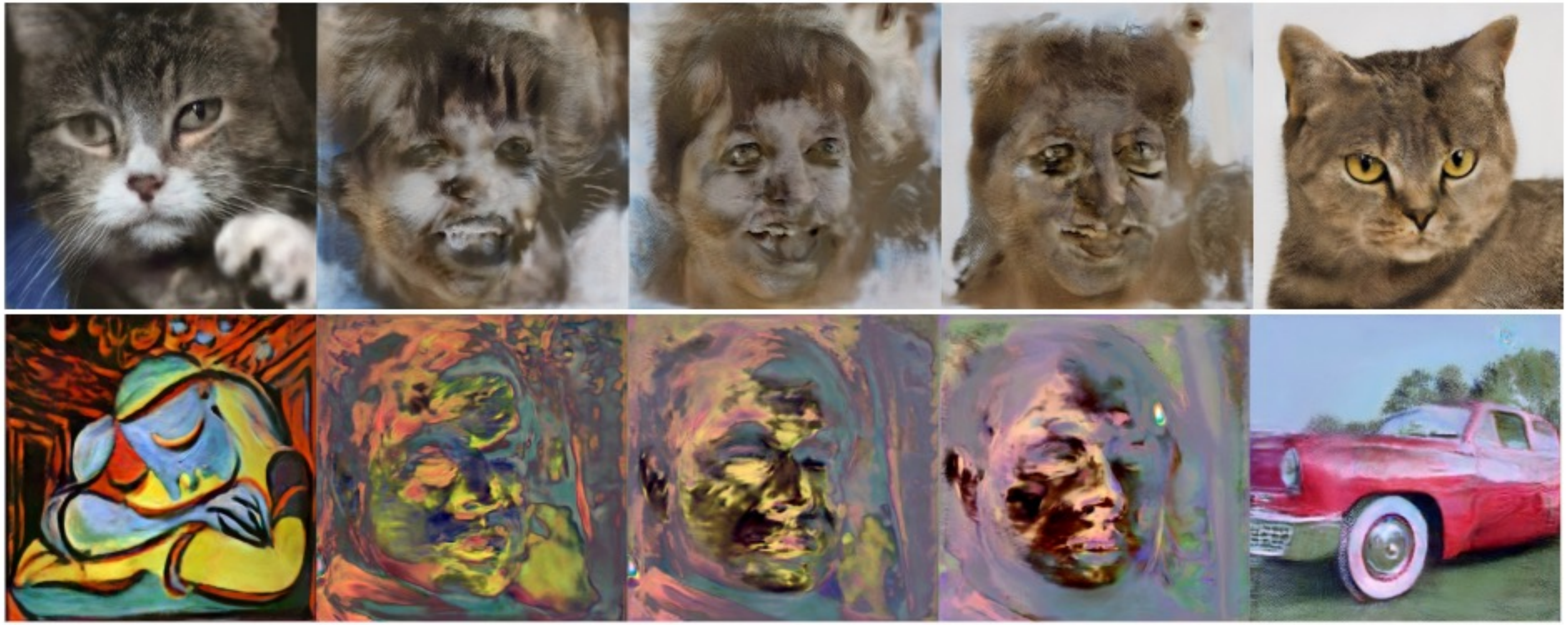
Debugging GANs Projection (HW5)

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

- What can go wrong?
  - Generator: G (cannot generate the image or too deep)
  - Reconstruction loss: L (not a good image distance)
  - Optimization method: SGD, ADAM (local minimum)
    (1) use a more advanced solver: e.g., L-BFGS (Quasi-Newton)
    (2) train an encoder to initialize the latent code. E(x)

- Debugging steps:
  - Reconstruct a generated image
  - Reconstruct a training set real image
  - Reconstruct a validation/test set real image
  - Reconstruct an in-the-wild image (e.g., Internet photo, camera roll)

Reconstruction $\neq$ Editing



Interpolations between two images

Image2StyleGAN [Abdal et al., 2019]

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

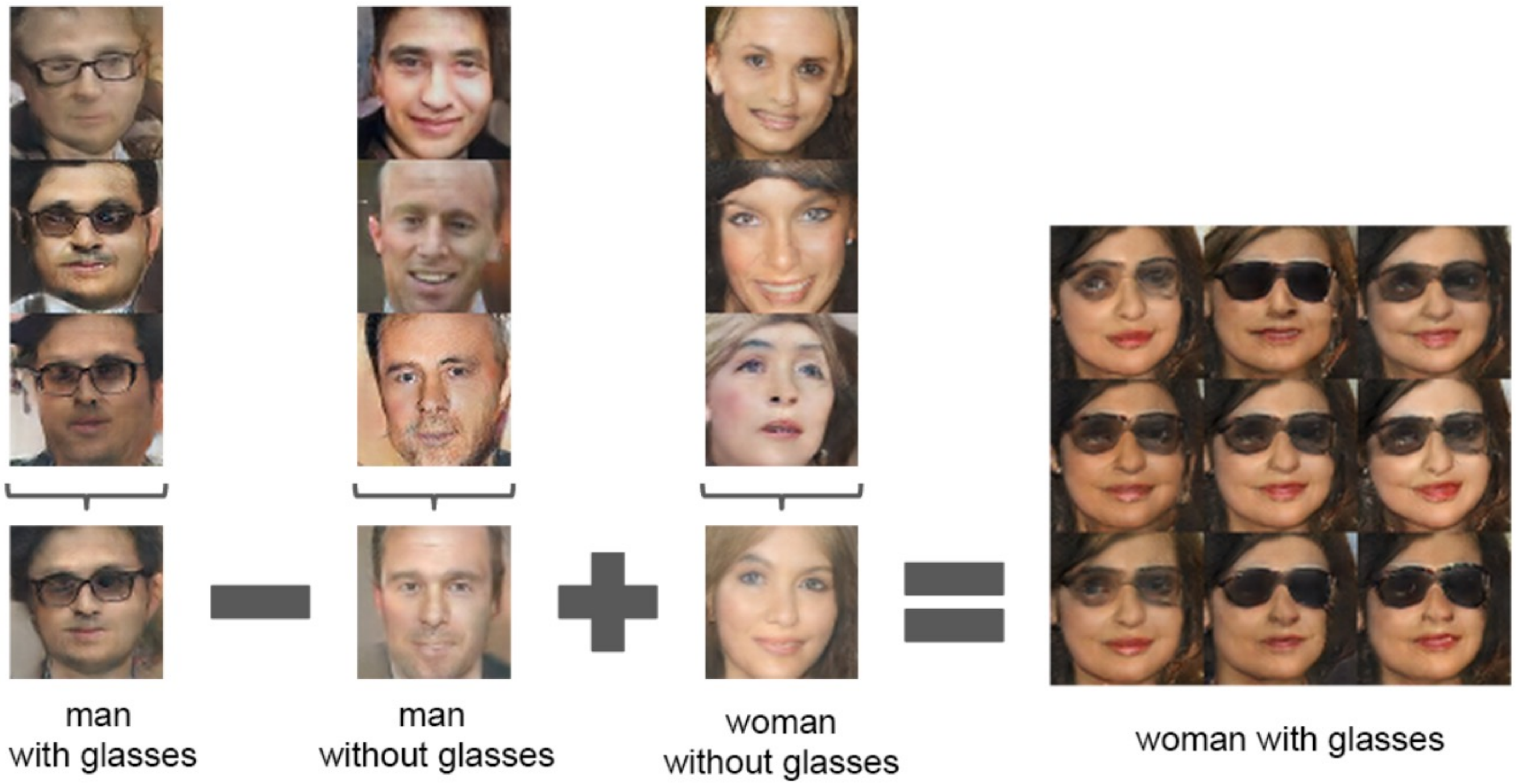
$$z_1 = z_0 + \Delta z$$

- 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)

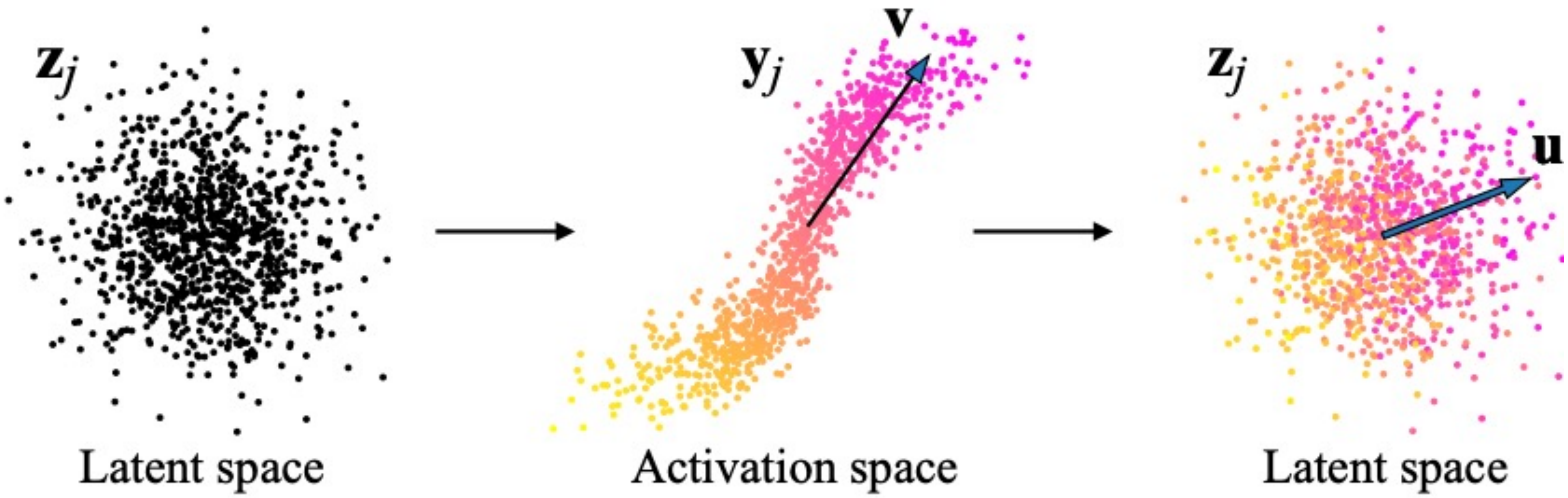
GANSpace: Discovering PCA directions

First find 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

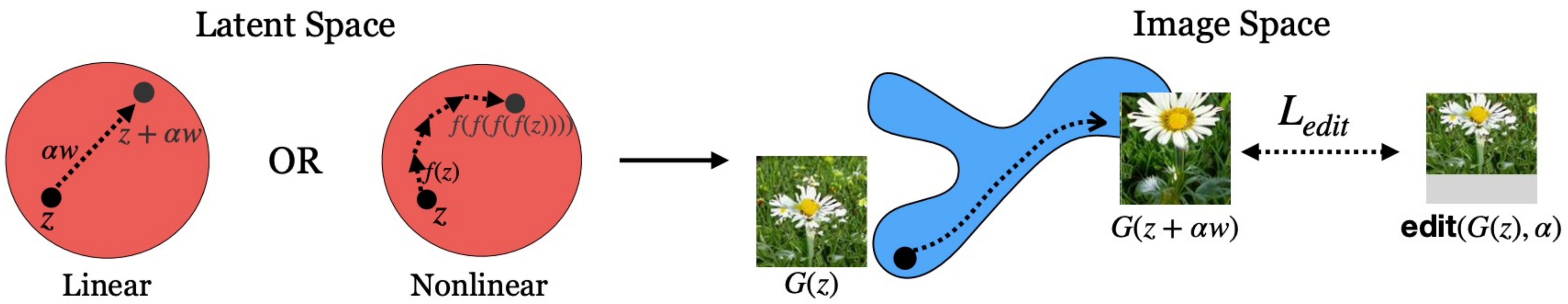
GANspace [Härkönen et al., 2020]

GANSpace: Discovering PCA directions

GANspace [Härkönen et al., 2020]

Manipulating Latent code/layer (offline optimization)

Offline optimization



Given a pre-defined function **edit** and a pre-trained generator **G**

Linear case:
(w is a vector)

$$\arg\min_{w} \mathbb{E}_{z,\alpha}[\mathcal{L}(G(z+\alpha w), \texttt{edit}(G(z), \alpha))]$$

strength

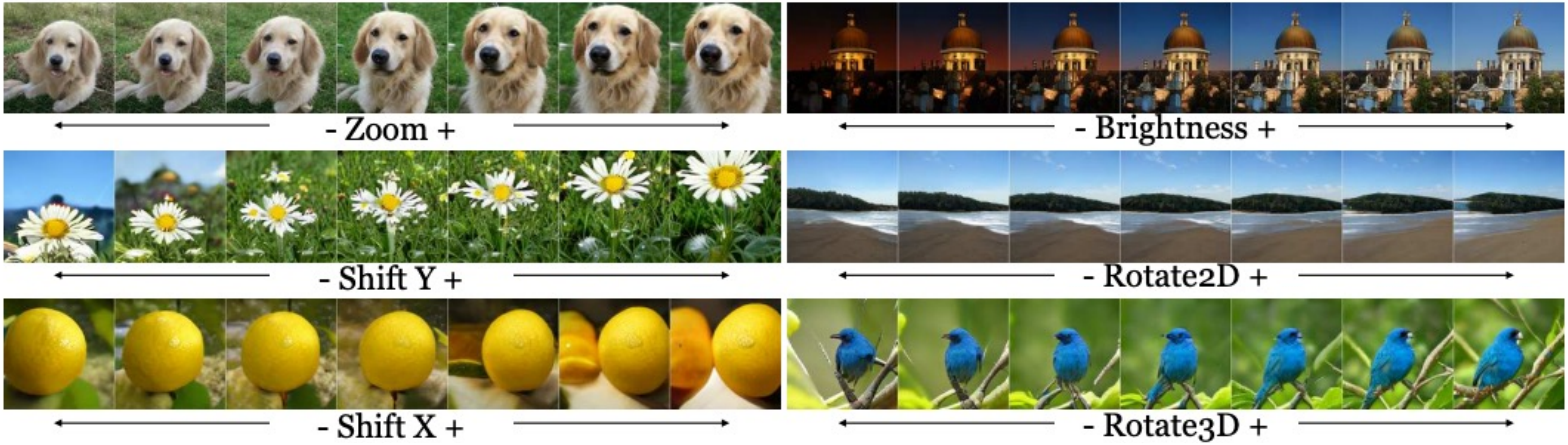
Non-linear case:
(f is a function)
apply it n times

$$\arg\min_{f} \mathbb{E}_{z,n}[||G(f^n(z)) - \texttt{edit}(G(z), n\epsilon)||],$$

strength

[Jahanian*, Chai*, Isola. 2020]

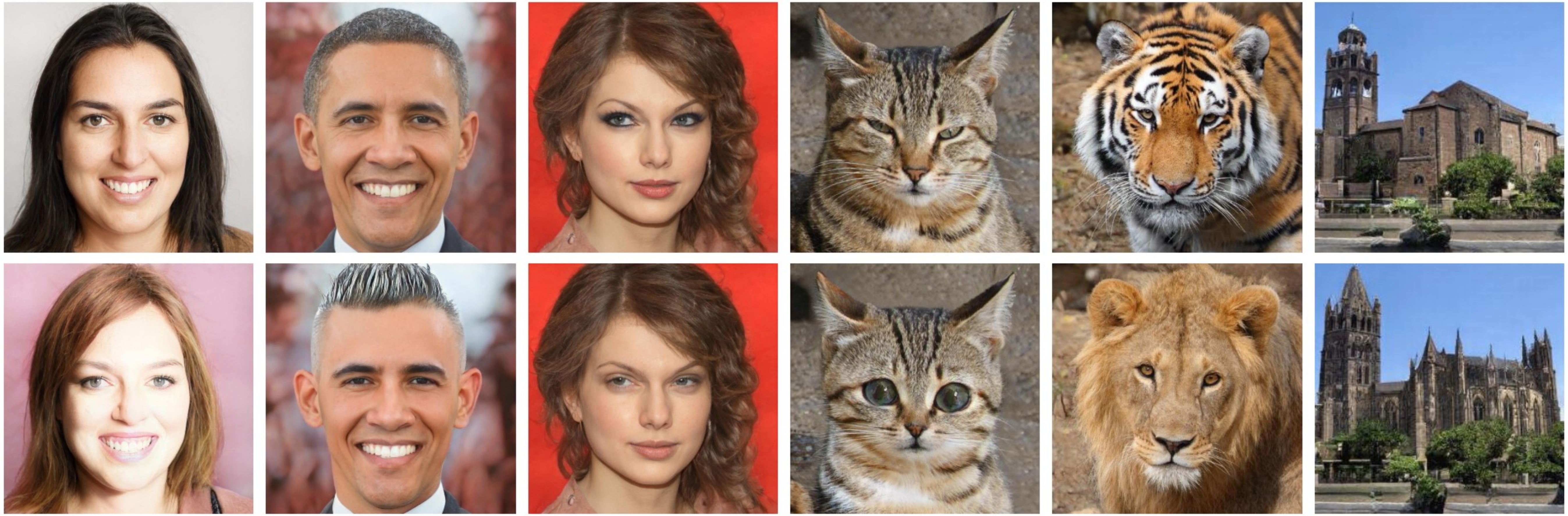
Offline optimization



Requirement: A known **edit** function
(e.g., shift, zoom, rotate)

[Jahanian*, Chai*, Isola. 2020]

CLIP-guided Directions



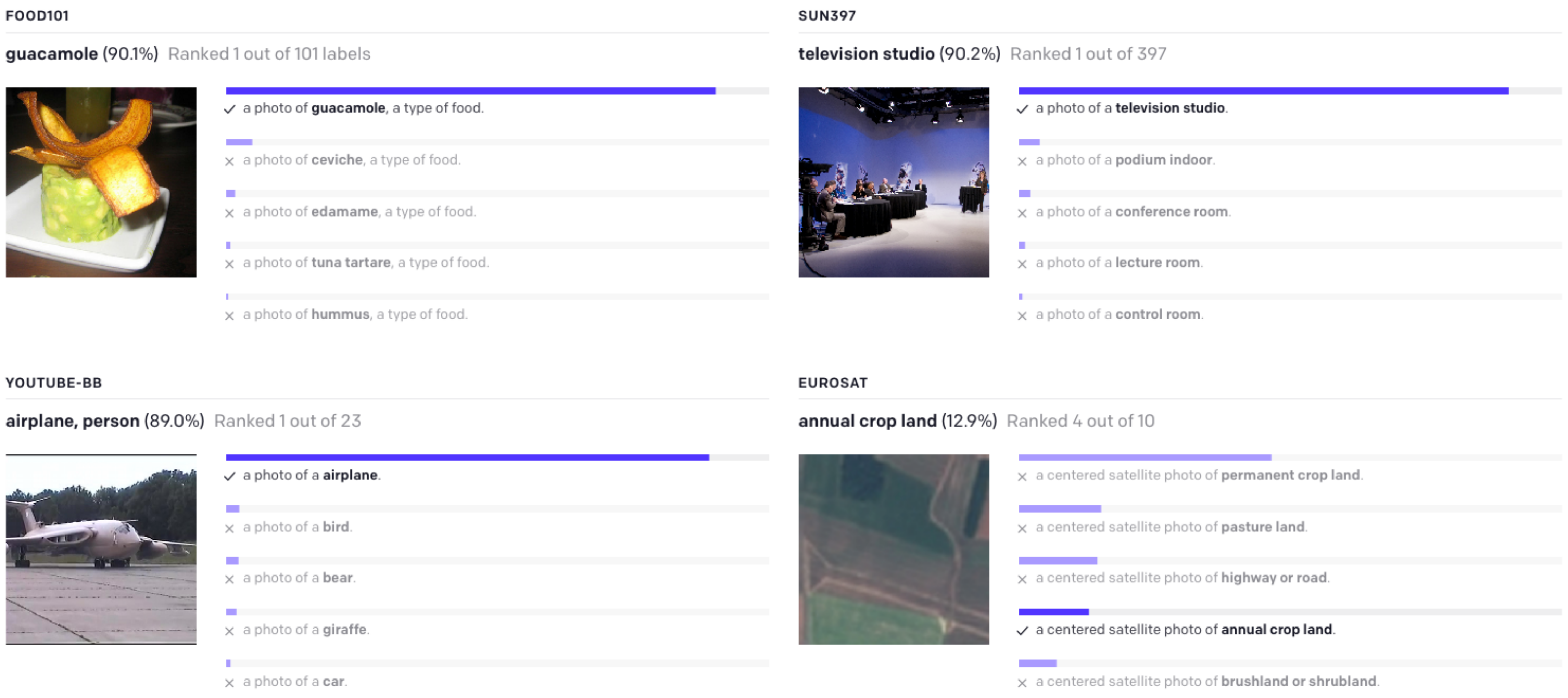
"Emma Stone" "Mohawk hairstyle" "Without makeup" "Cute cat" "Lion" "Gothic church"

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

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]

CLIP: Connecting Text and Images



<u>Input</u>: an image and a caption.

<u>Output</u>: similarity between the text embedding and the image embedding

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]

GAN Inversion Demo

Manipulating network weights

Image Editing with GANs

- Step 1: Image Projection/Reconstruction

$$z_0, \theta_0 = \arg\min_{z,\theta} \mathcal{L}(G(z;\theta), x)$$

- Step 2: Manipulating the network weights

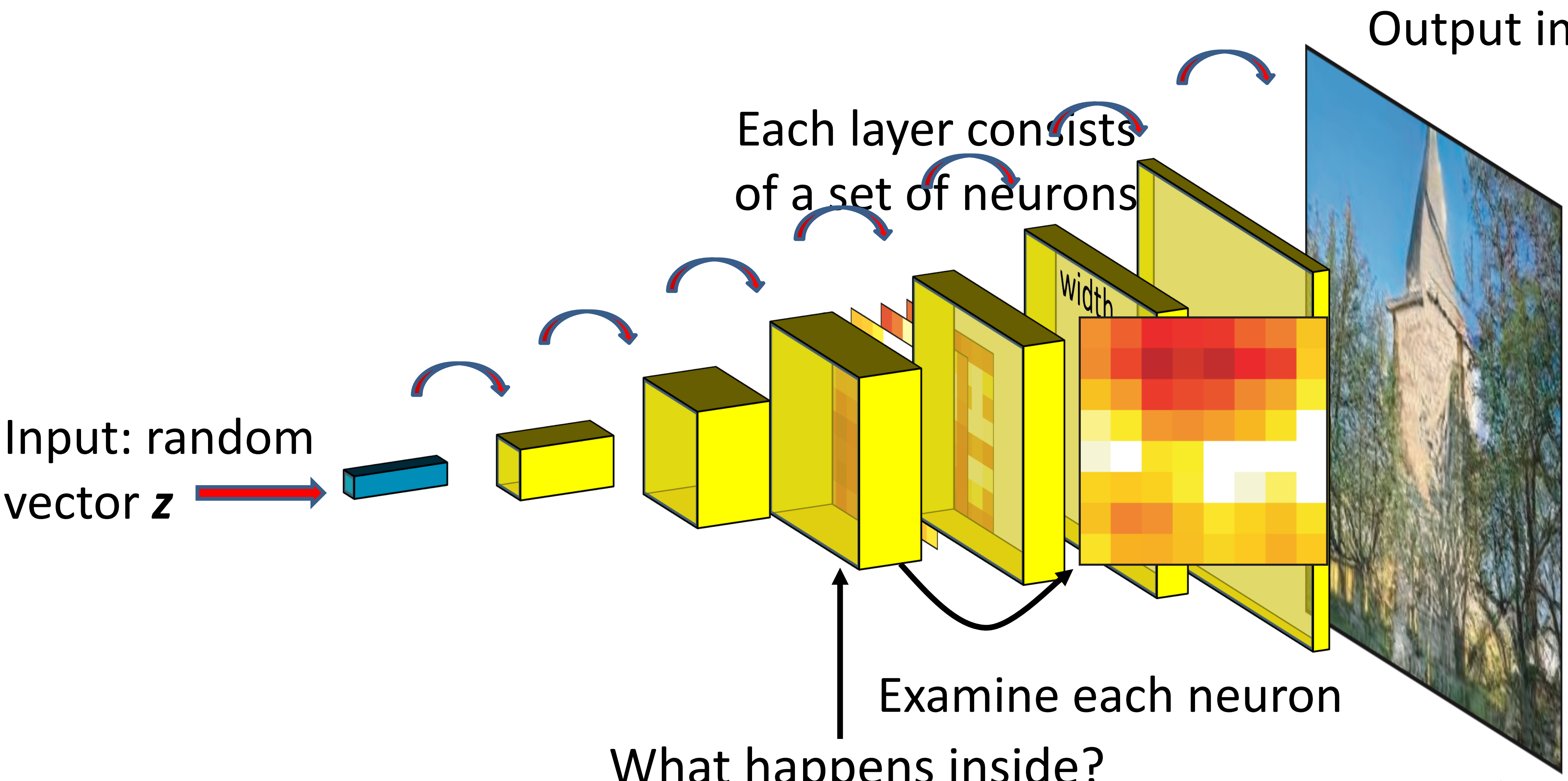
$$\theta_1 = \theta_0 + \Delta\theta$$

- Step 3: Generate the edited result

$$G(z_0; \theta_1)$$

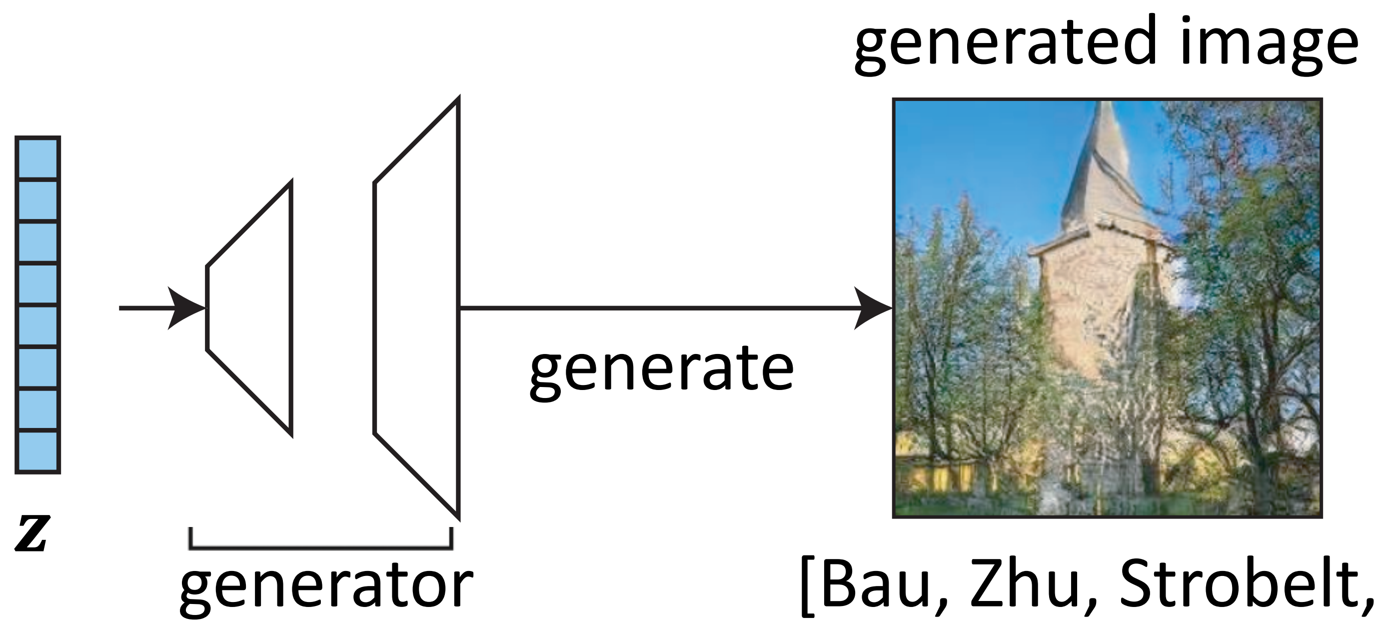
Understanding a Generator

Each step:
Increases spatial resolution



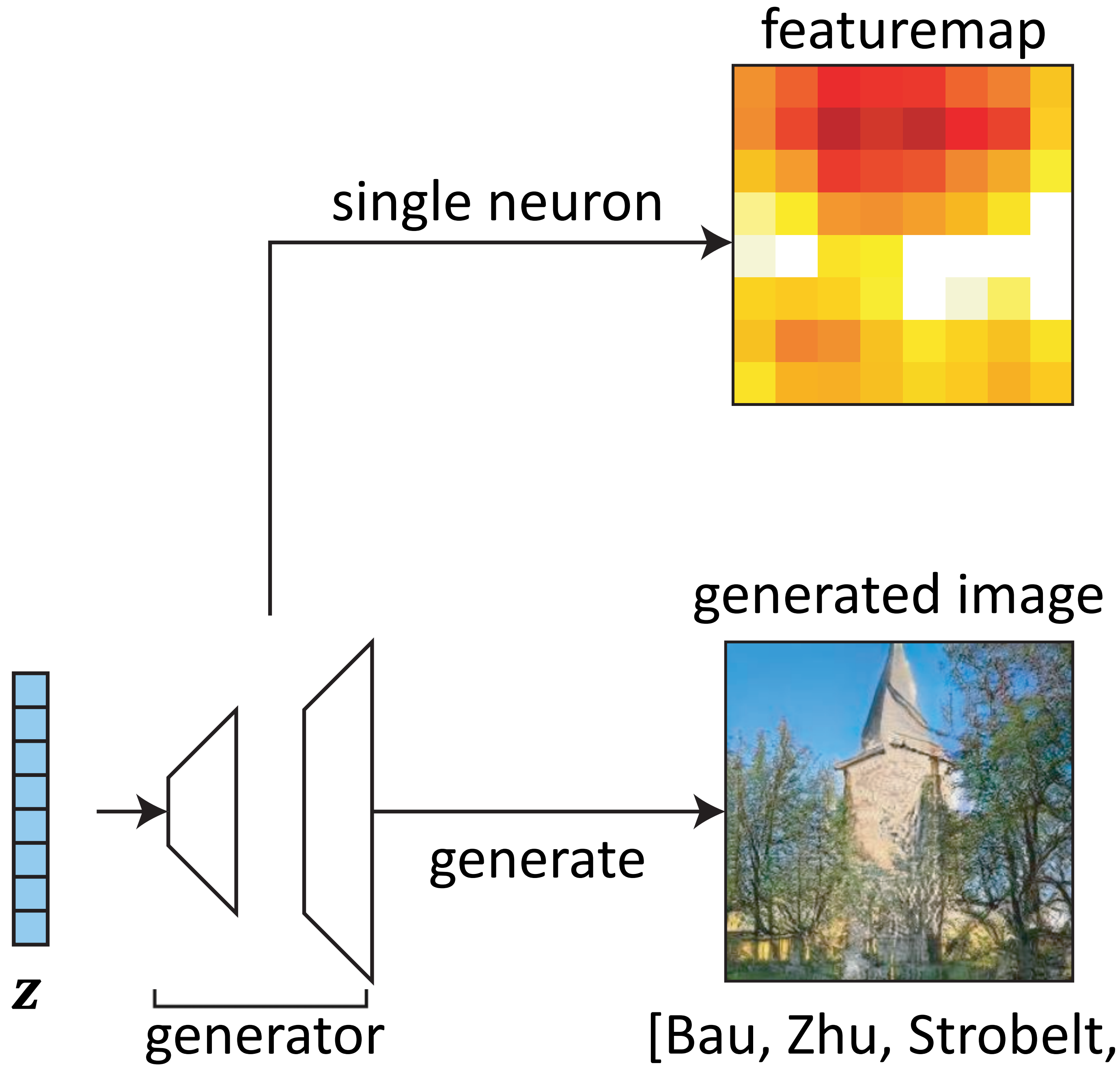
Progressive GANs [Karras et al. 2018]

Which neurons **correlate** to an object class?



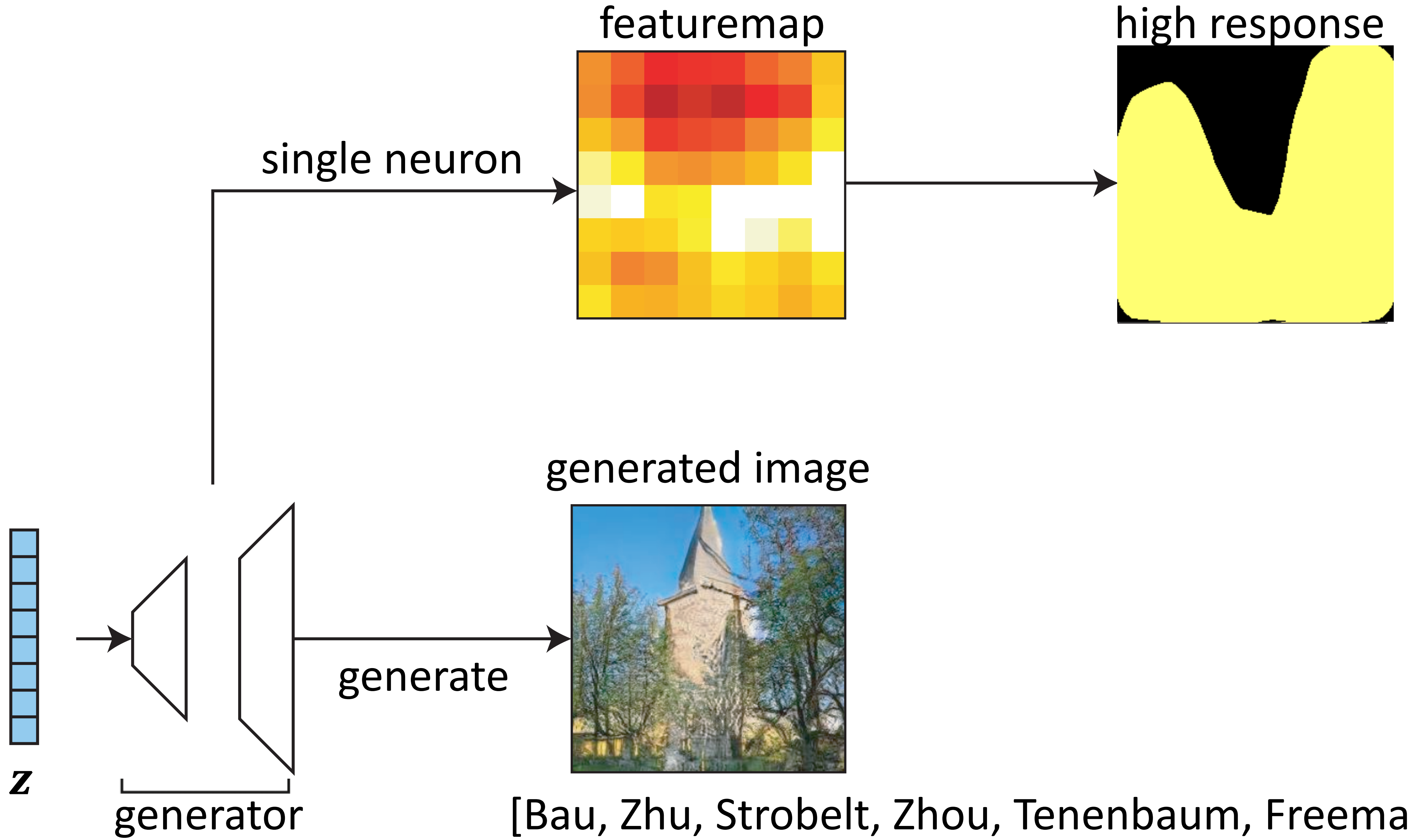
[Bau, Zhu, Strobelt, Zhou, Tenenbaum, Freeman, Torralba. ICLR 2019]

Which neurons **correlate** to an object class?



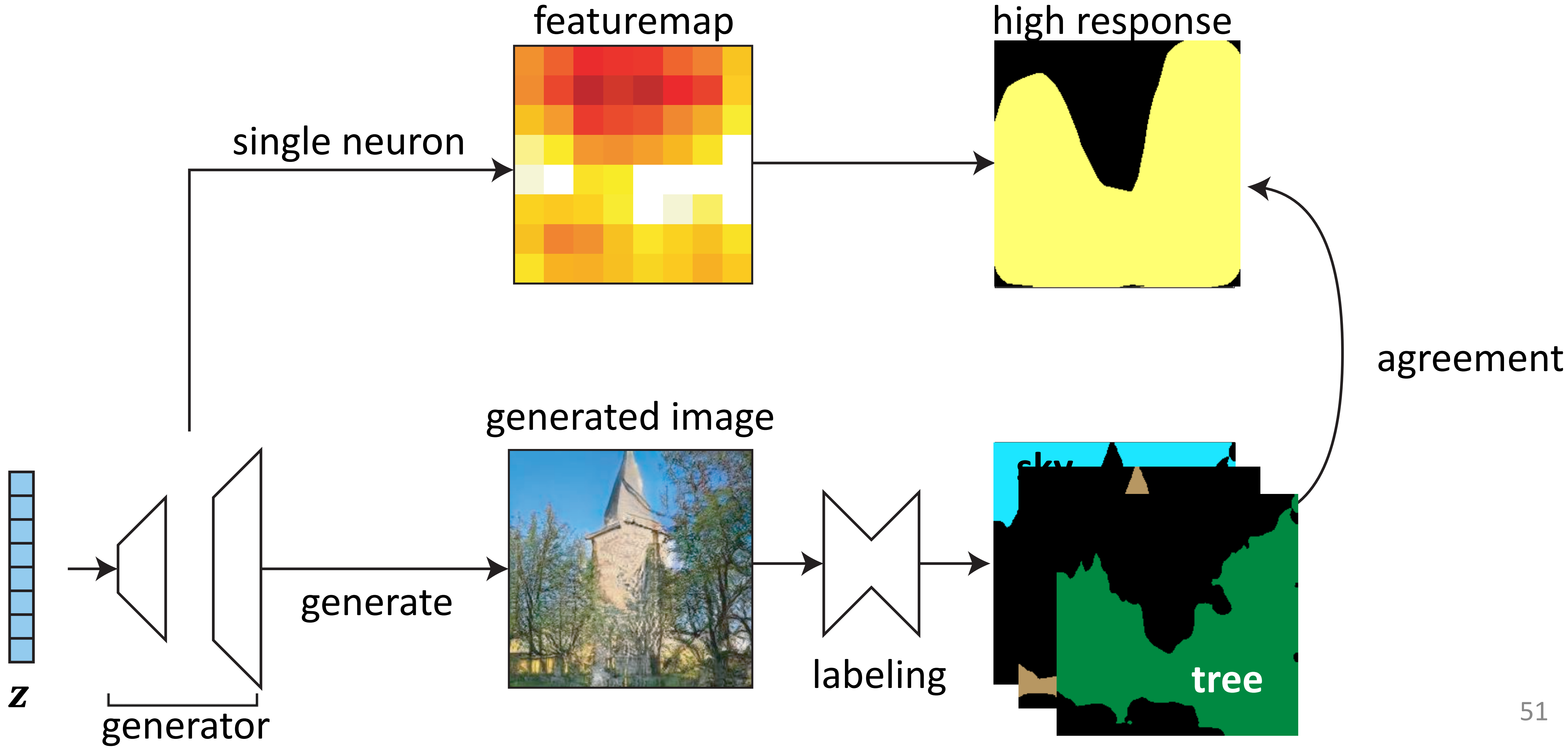
[Bau, Zhu, Strobelt, Zhou, Tenenbaum, Freeman, Torralba. ICLR 2019]

Which neurons **correlate** to an object class?



[Bau, Zhu, Strobelt, Zhou, Tenenbaum, Freeman, Torralba. ICLR 2019]

Which neurons **correlate** to an object class?



Which neurons **correlate** to an object class?

Church samples

Tree Neuron

Dome Neuron

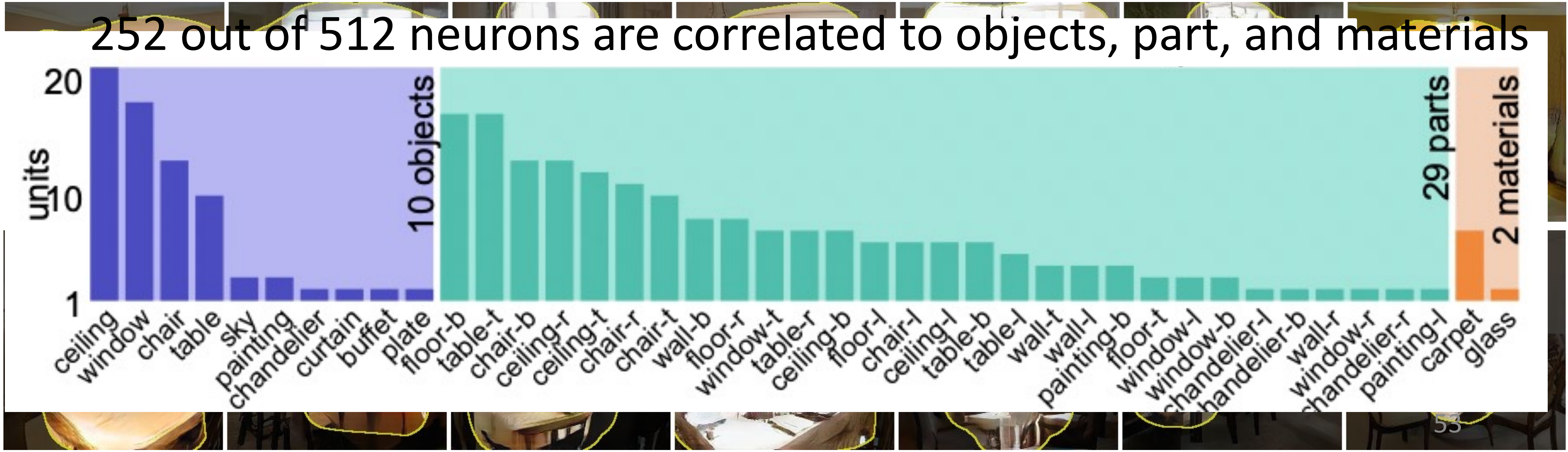
Which neurons **correlate** to an object class?

Dining room samples

252 out of 512 neurons are correlated to objects, part, and materials

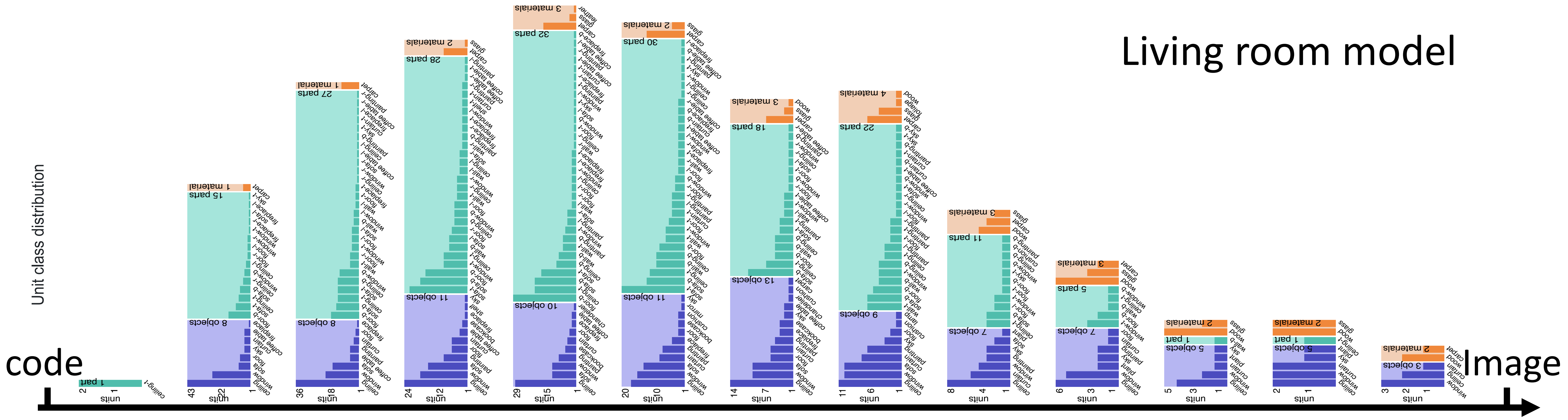
Window Neuron

Table Neuron



Which neurons **correlate** to an object class?

Living room model

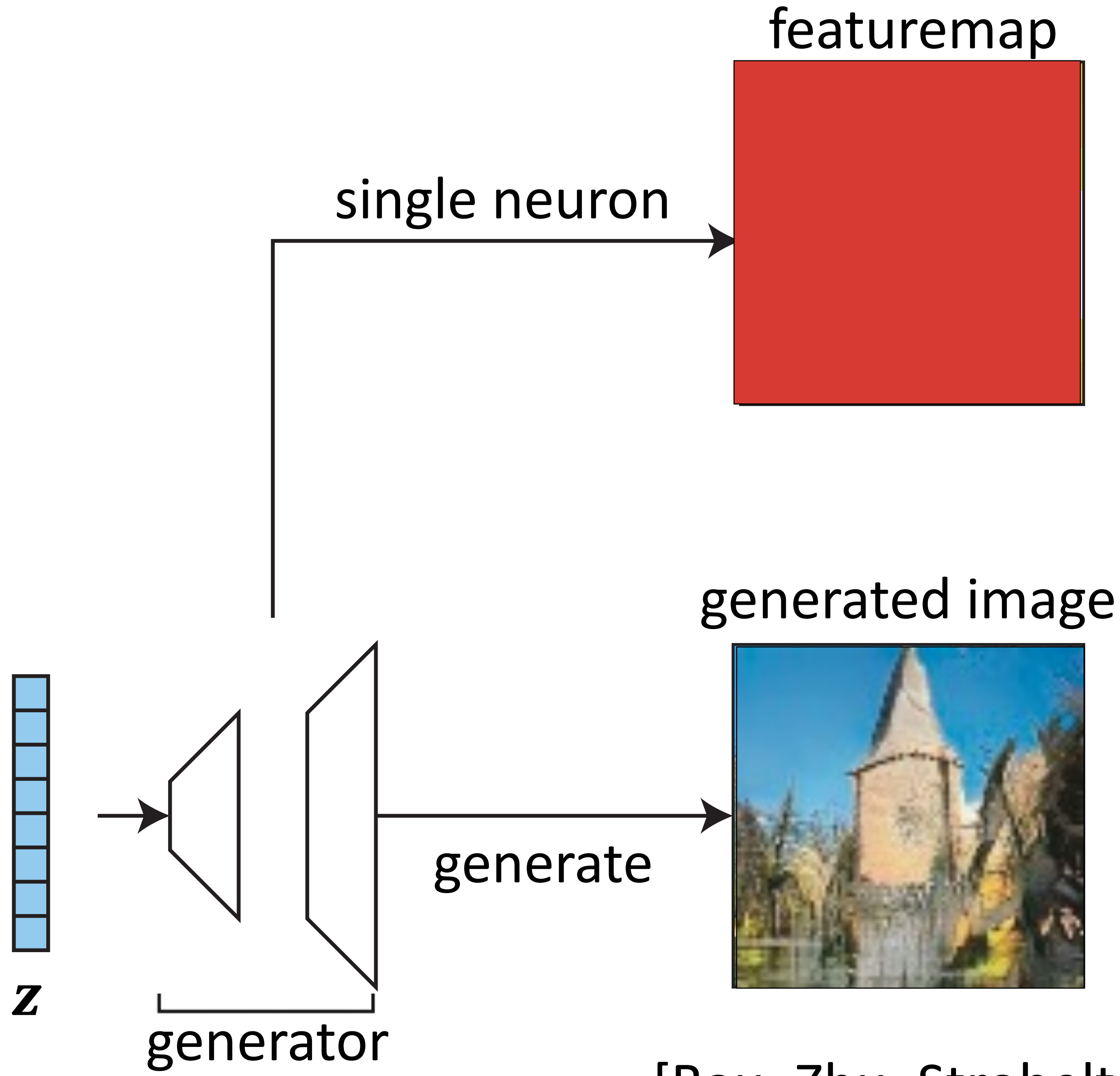


Layout

Object and parts

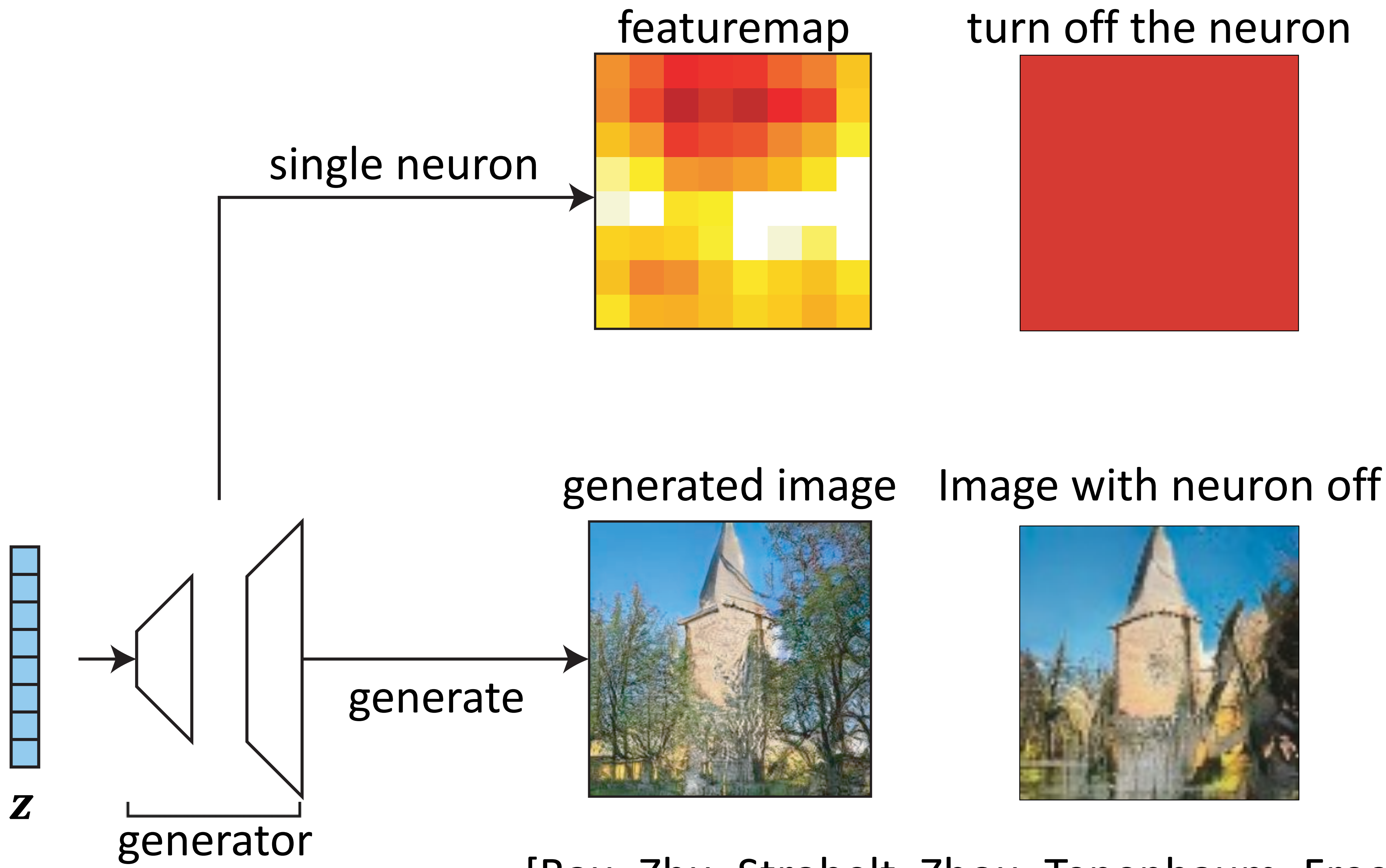
Edges, textures, local structure

Which neurons **cause** an object class?



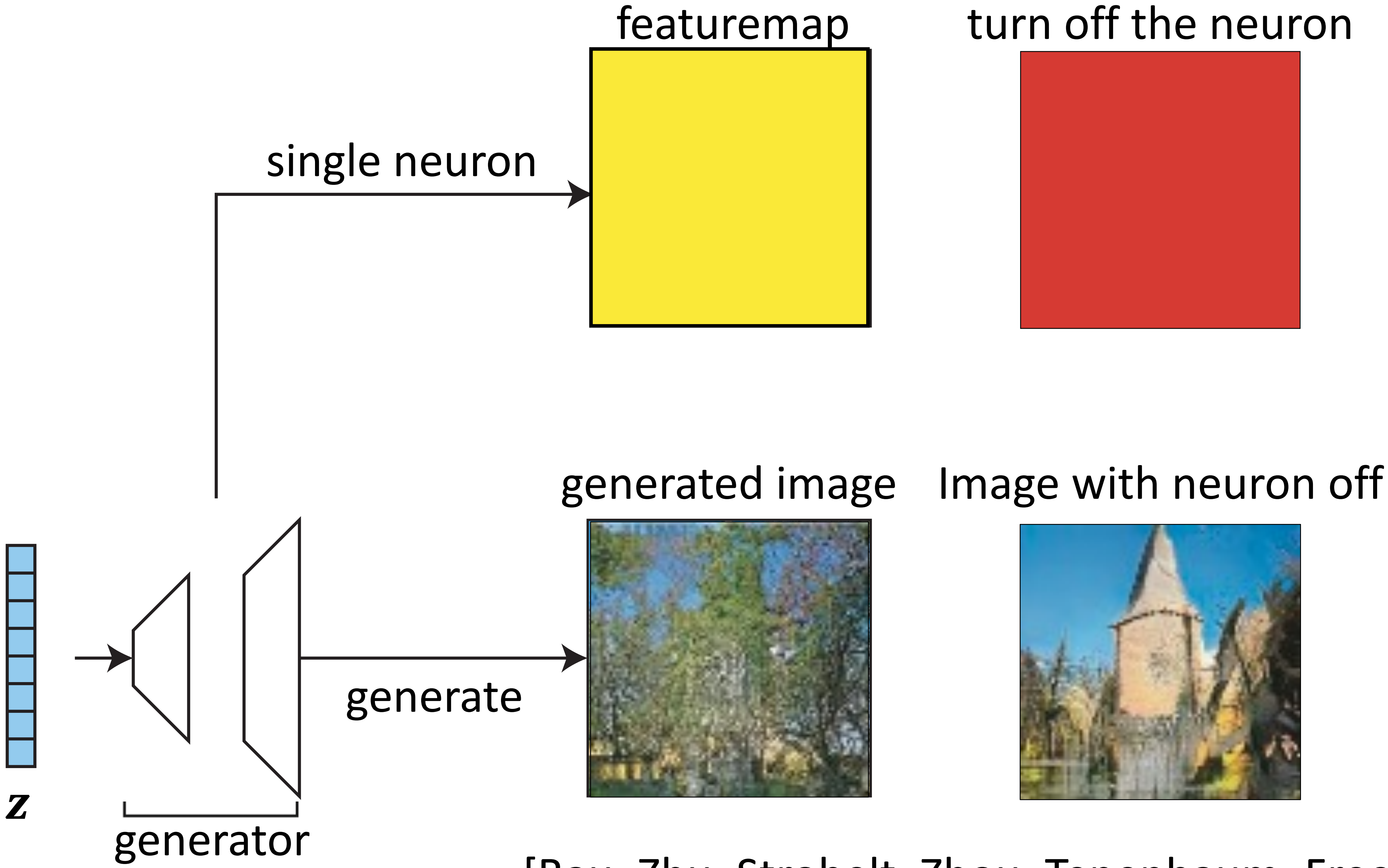
[Bau, Zhu, Strobelt, Zhou, Tenenbaum, Freeman, Torralba. ICLR 2019]

Which neurons **cause** an object class?



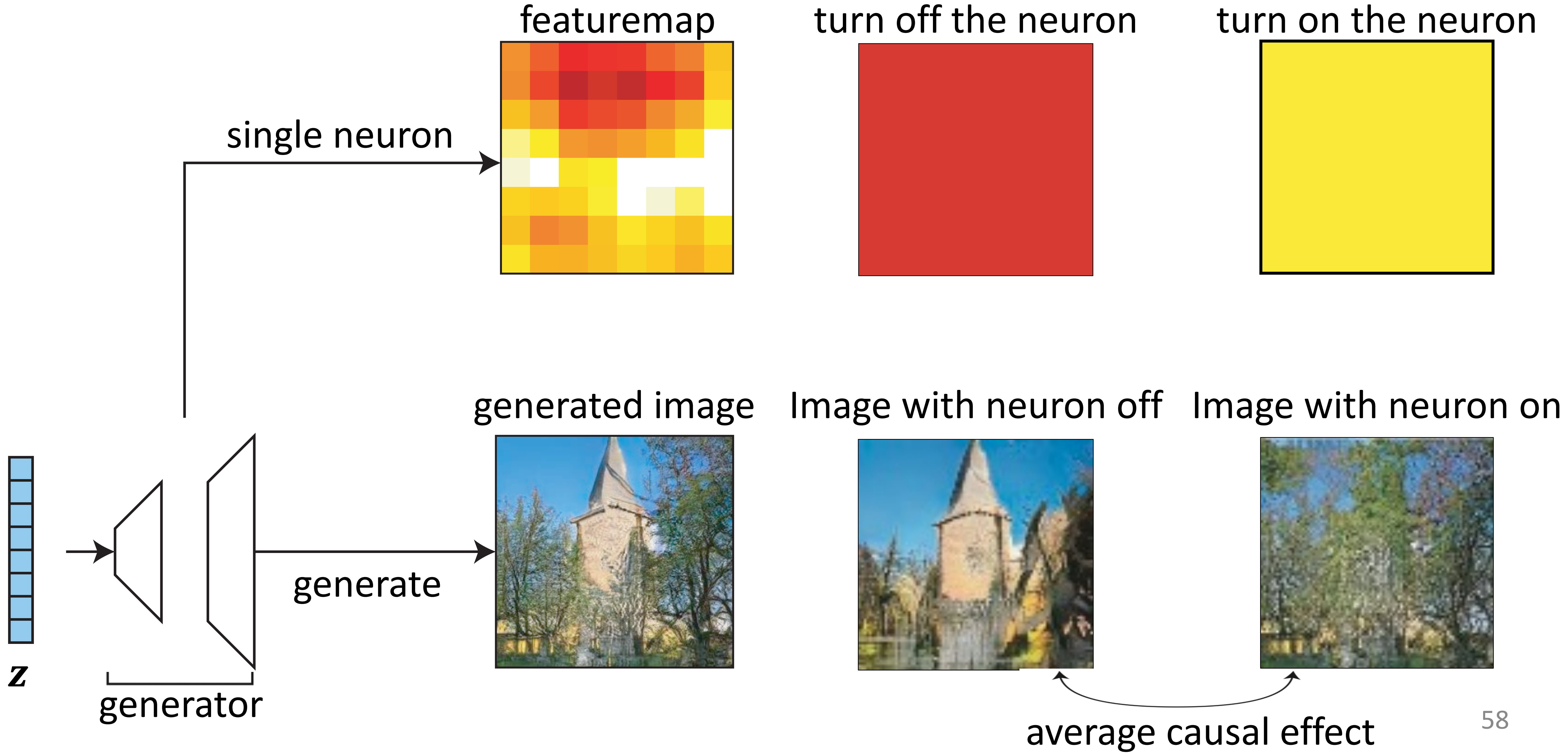
[Bau, Zhu, Strobelt, Zhou, Tenenbaum, Freeman, Torralba. ICLR 2019]

Which neurons **cause** an object class?



[Bau, Zhu, Strobelt, Zhou, Tenenbaum, Freeman, Torralba. ICLR 2019]

Which neurons **cause** an object class?



Which neurons **cause** an object class?

Object-Scene Relationships

Turn off **person** neurons

Object-Scene Relationships

Turn off **window** neurons

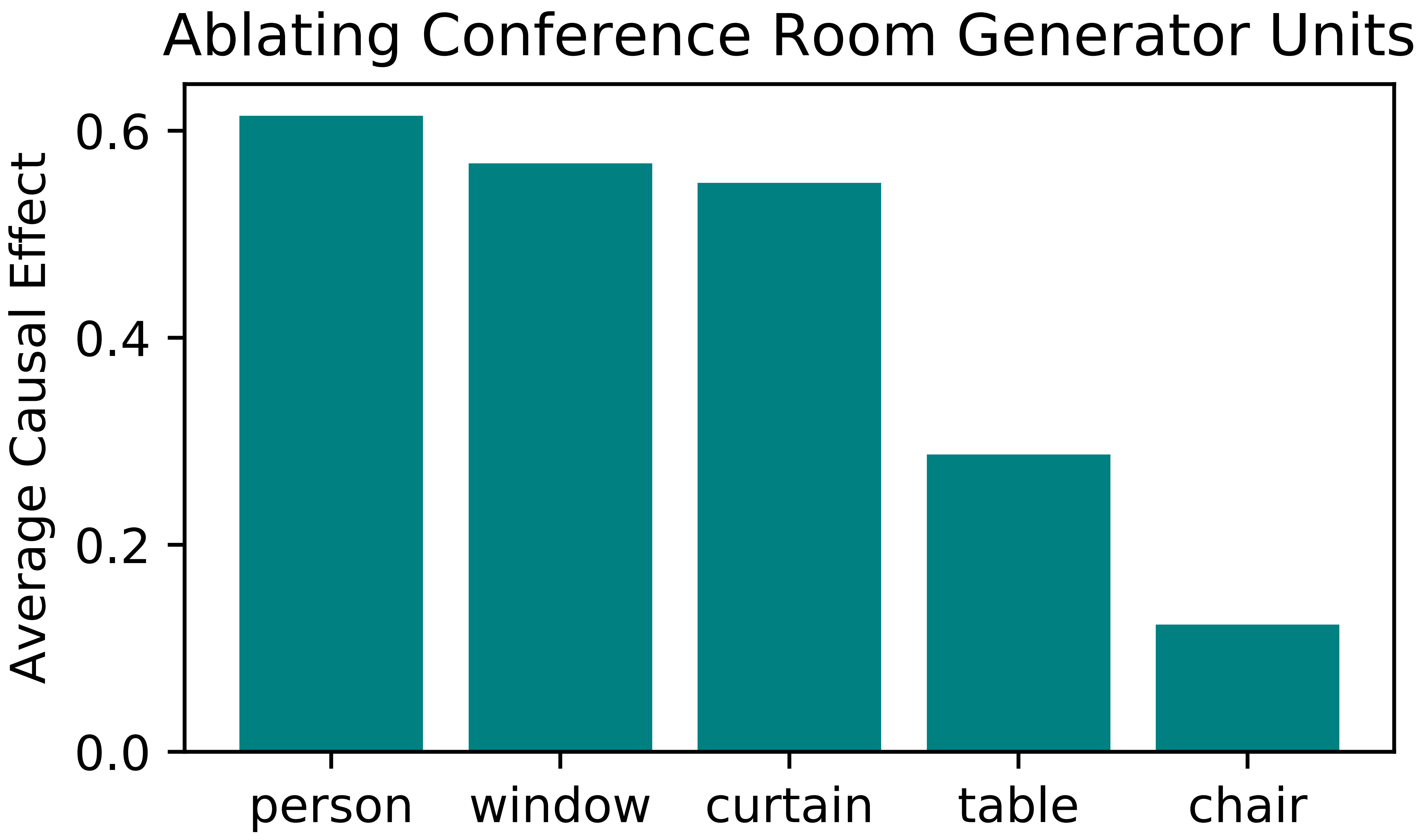
Object-Scene Relationships

Turn off **table** neurons

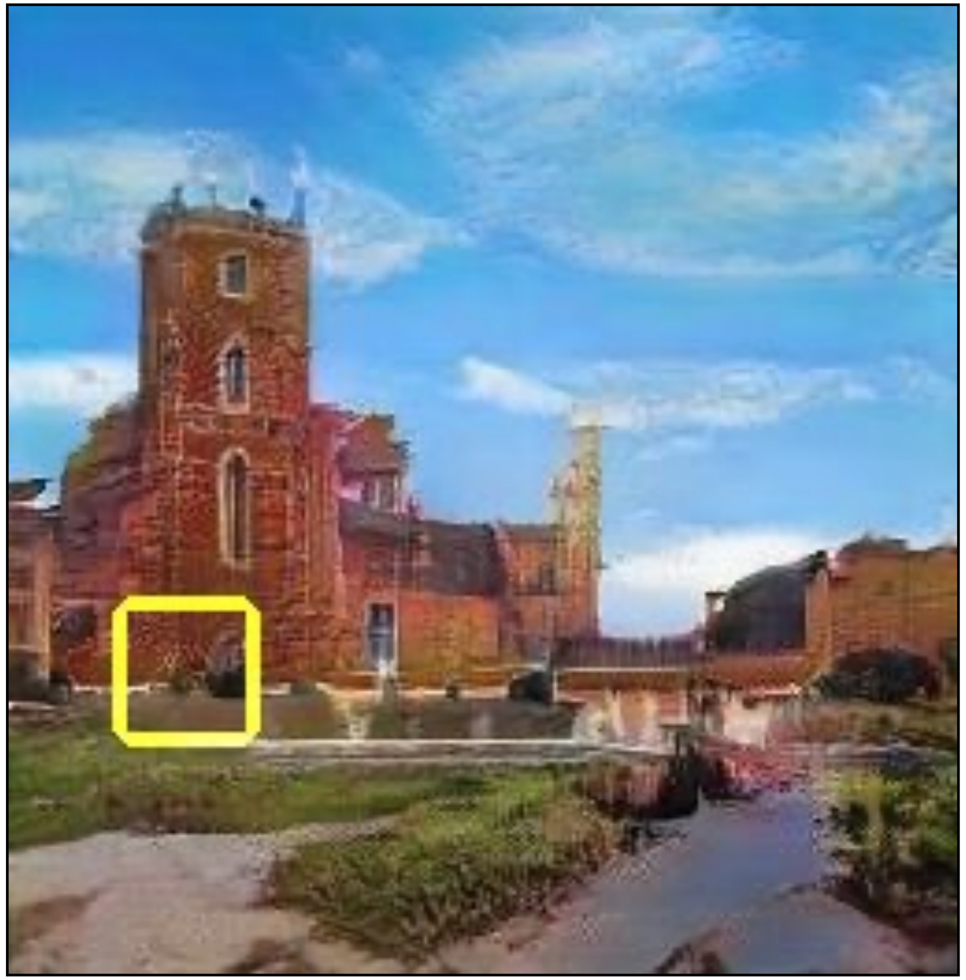
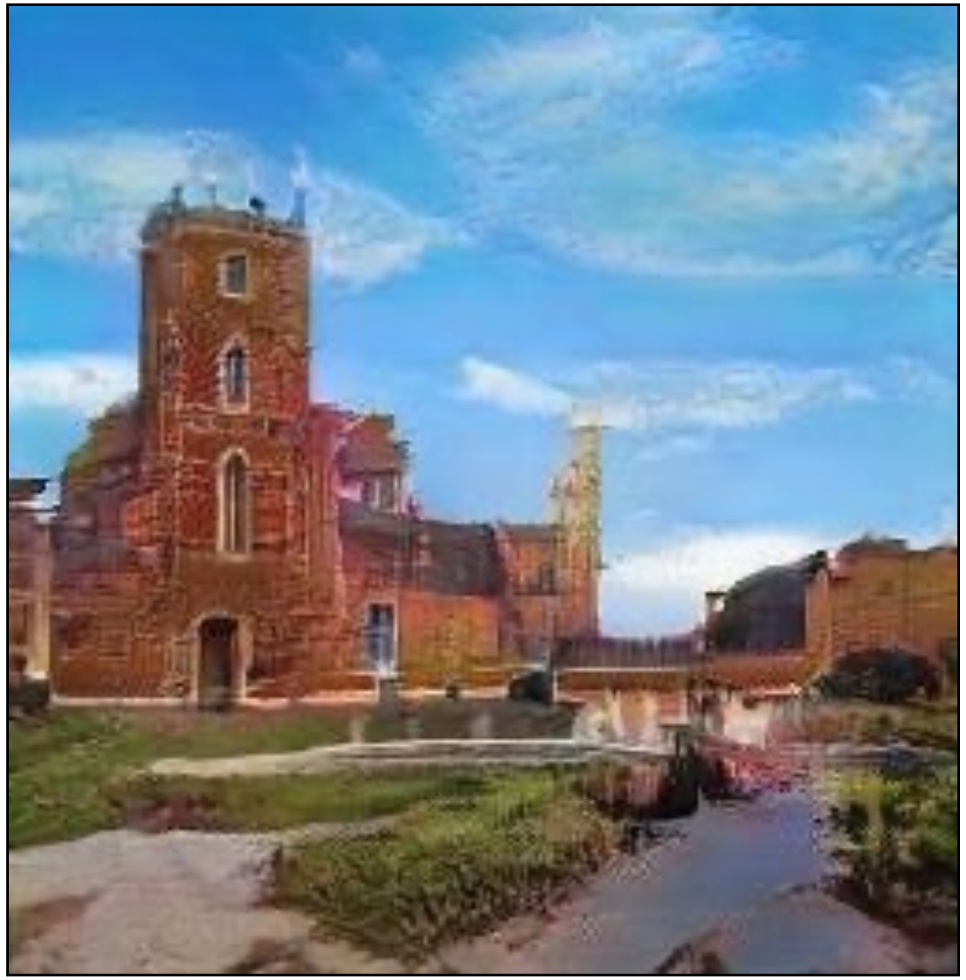
Object-Scene Relationships

Turn off **chair** neurons

Object-Scene Relationships



Object-Scene Relationships

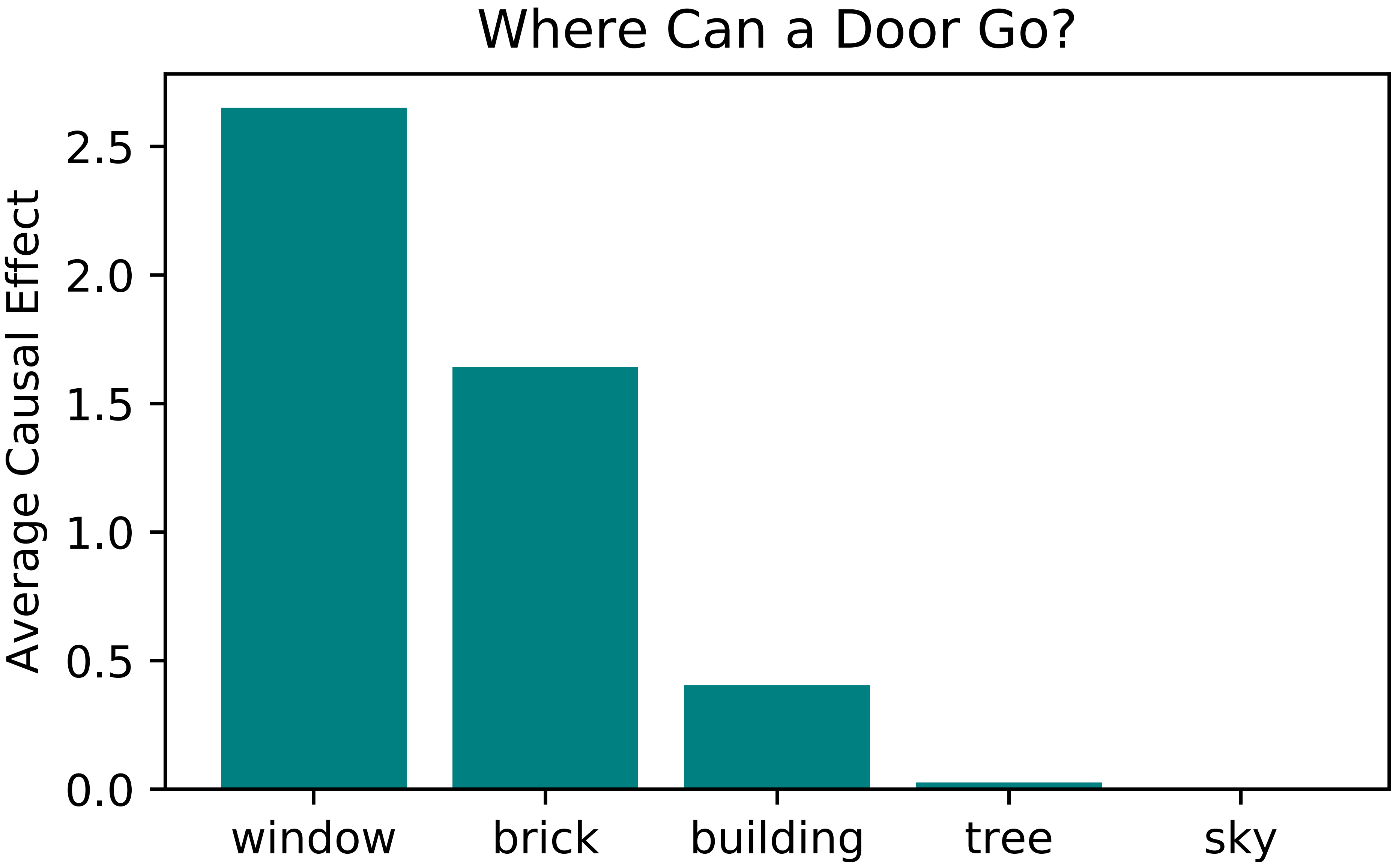



Yellow box: highlight every location where we can insert doors

Object-Scene Relationships

Yellow box: highlight every location where we can insert doors

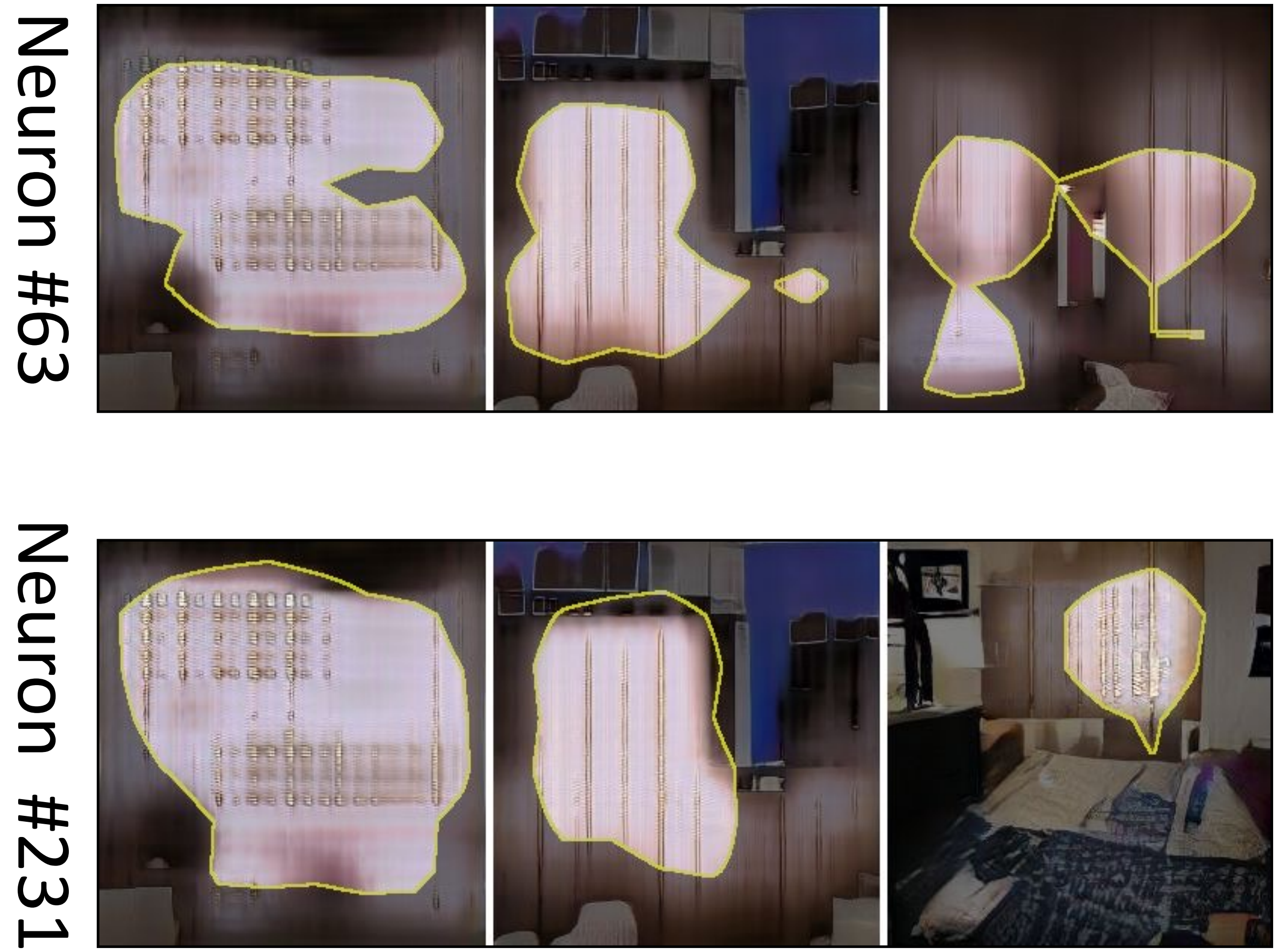
Object-Scene Relationships



Debugging and Improving Models

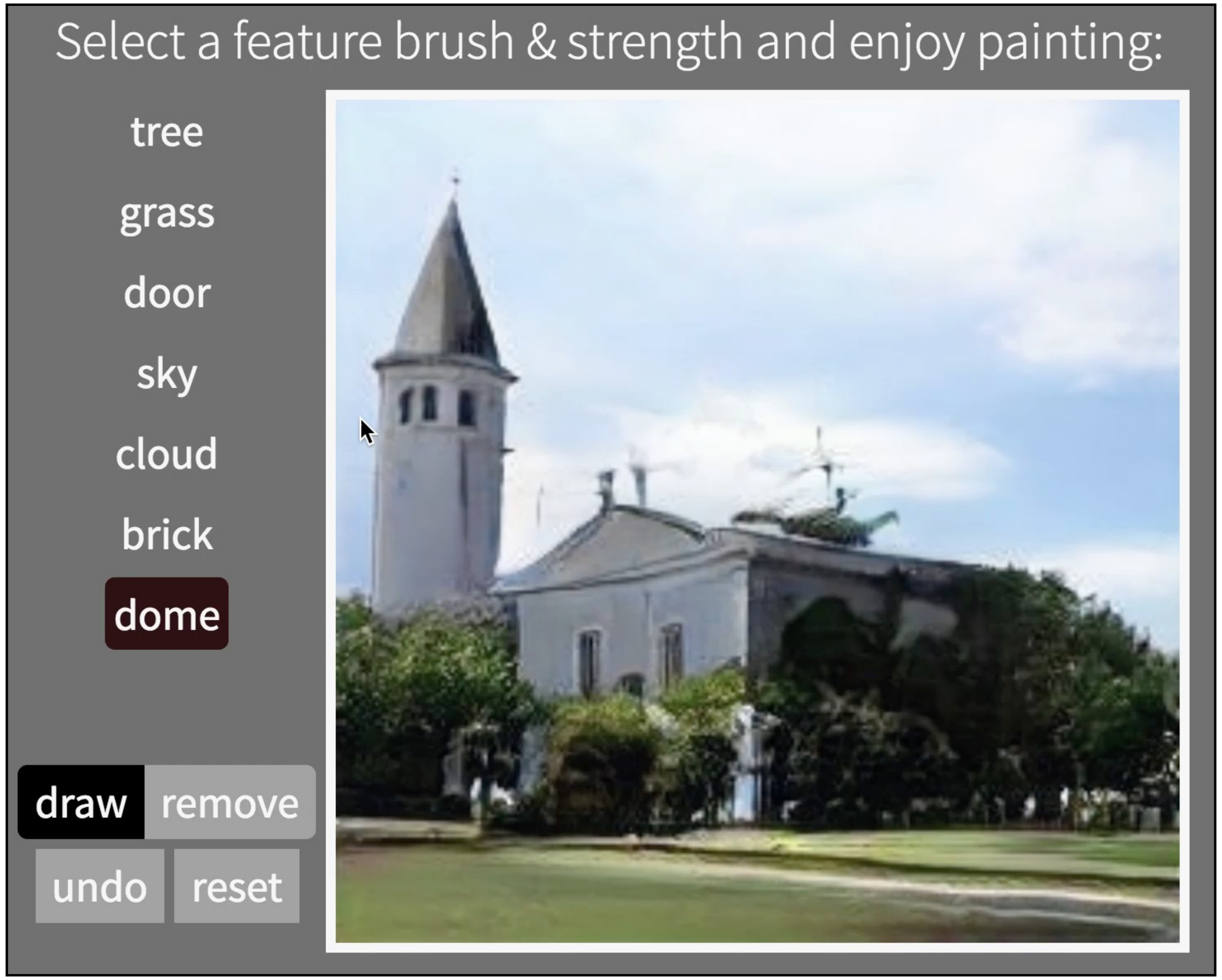
Images with artifacts

Turning off neurons removes artifacts



Example artifact-causing neurons

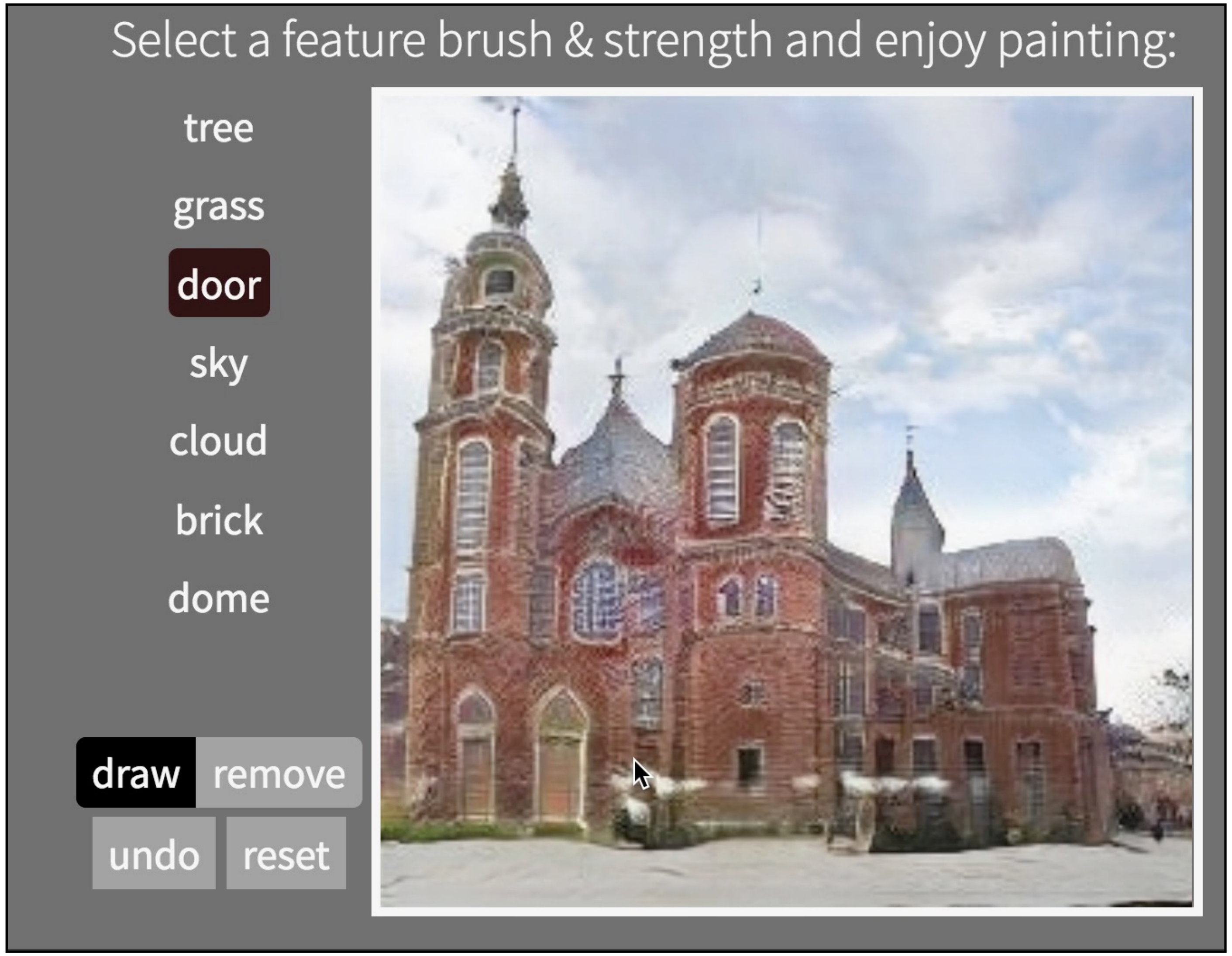
Interactive Painting



Online Demo

http://bit.ly/ganpaint

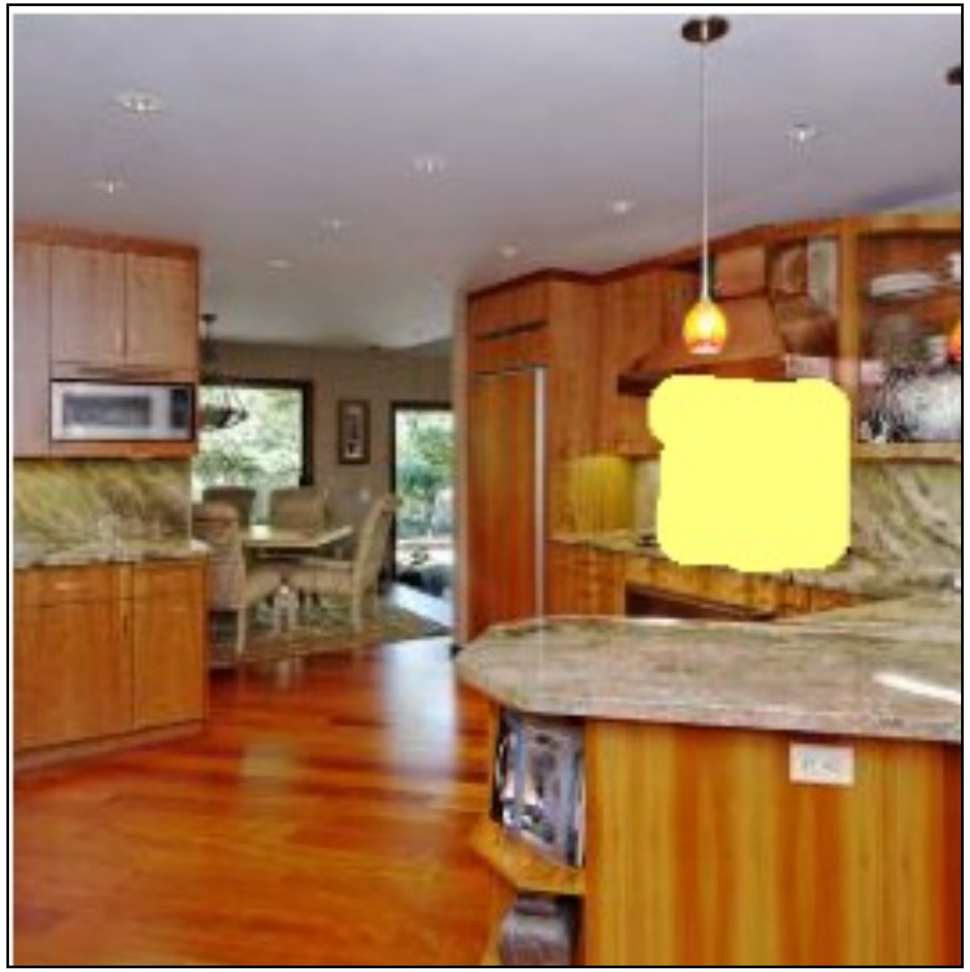
Interactive Painting



Online Demo

http://bit.ly/ganpaint

Manipulating a Real Photo



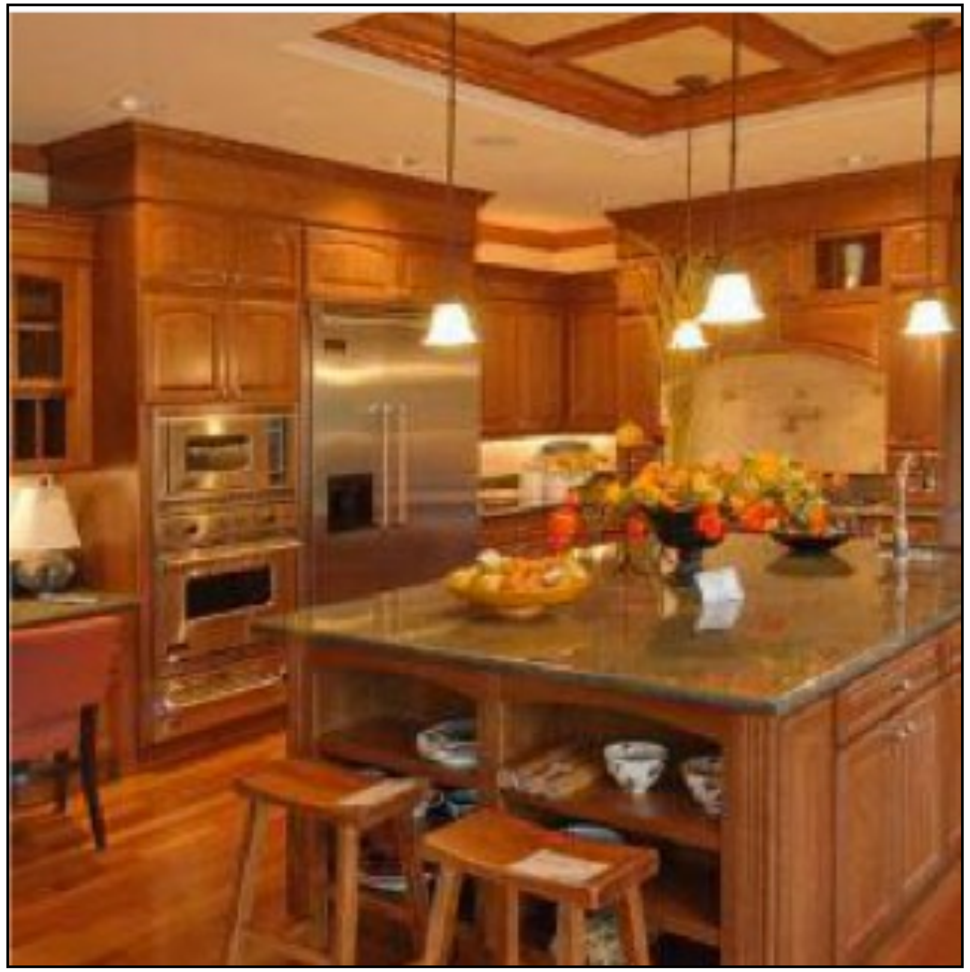
Original image + edits

Editing with $\hat{\boldsymbol{z}}$

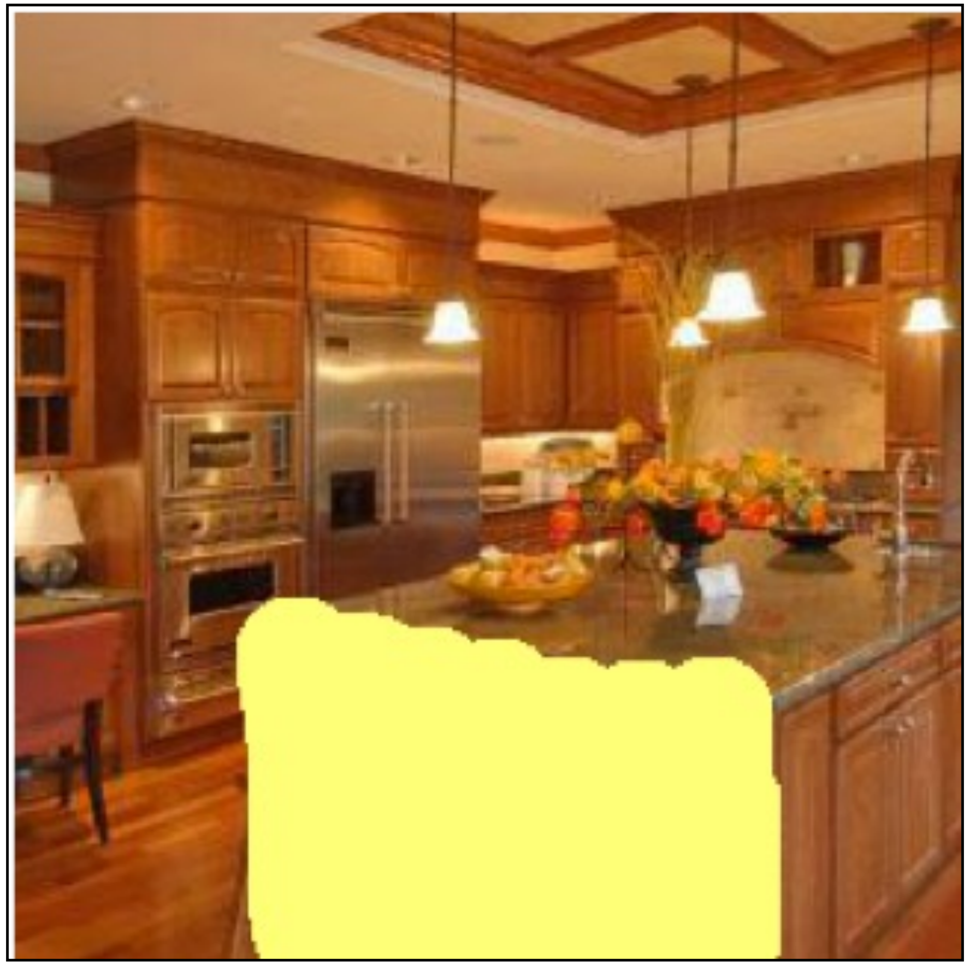
Editing with $\hat{\boldsymbol{z}}$ and $\widehat{\boldsymbol{\theta}}$

Semantic Photo Manipulation [Bau, Strobelt, Peebles, Wulff, Zhou, Zhu, Torralba, **SIGGRAPH** 2019]

Manipulating a Real Photo



Input image



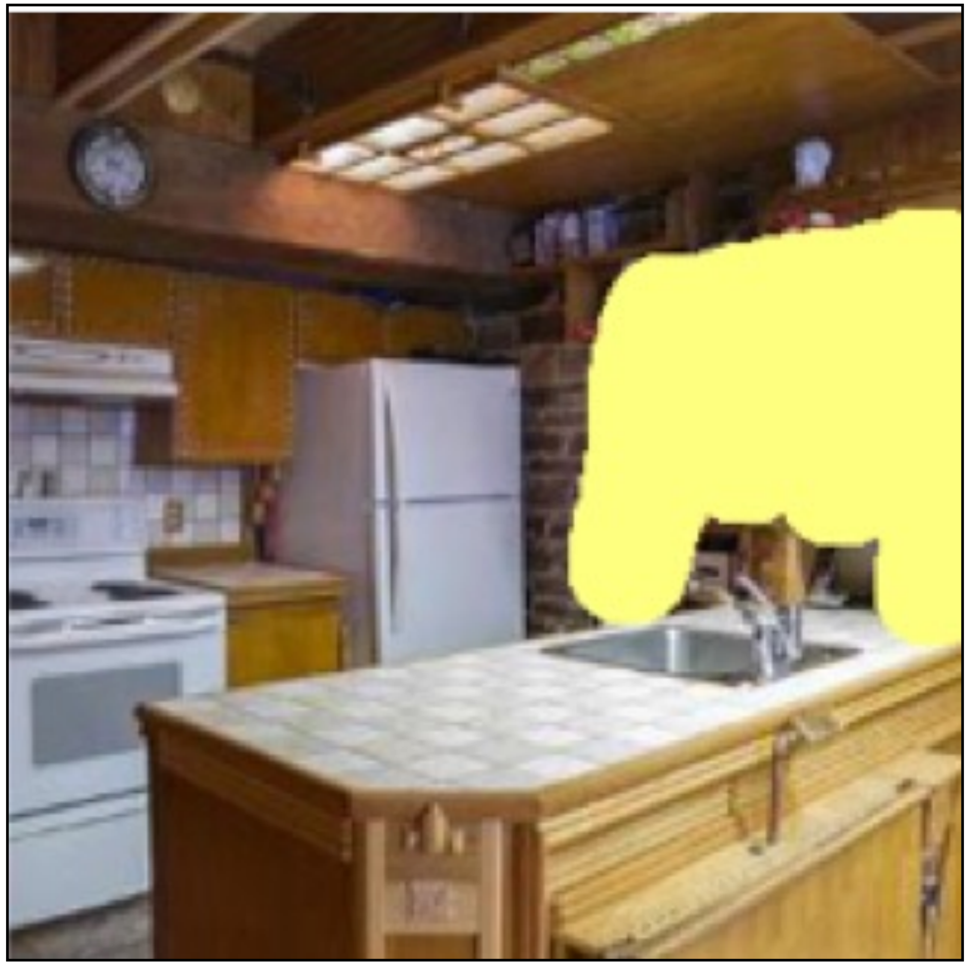
Remove chairs

Output result

Semantic Photo Manipulation [Bau, Strobelt, Peebles, Wulff, Zhou, Zhu, Torralba, **SIGGRAPH** 2019]

Manipulating a Real Photo

Input image



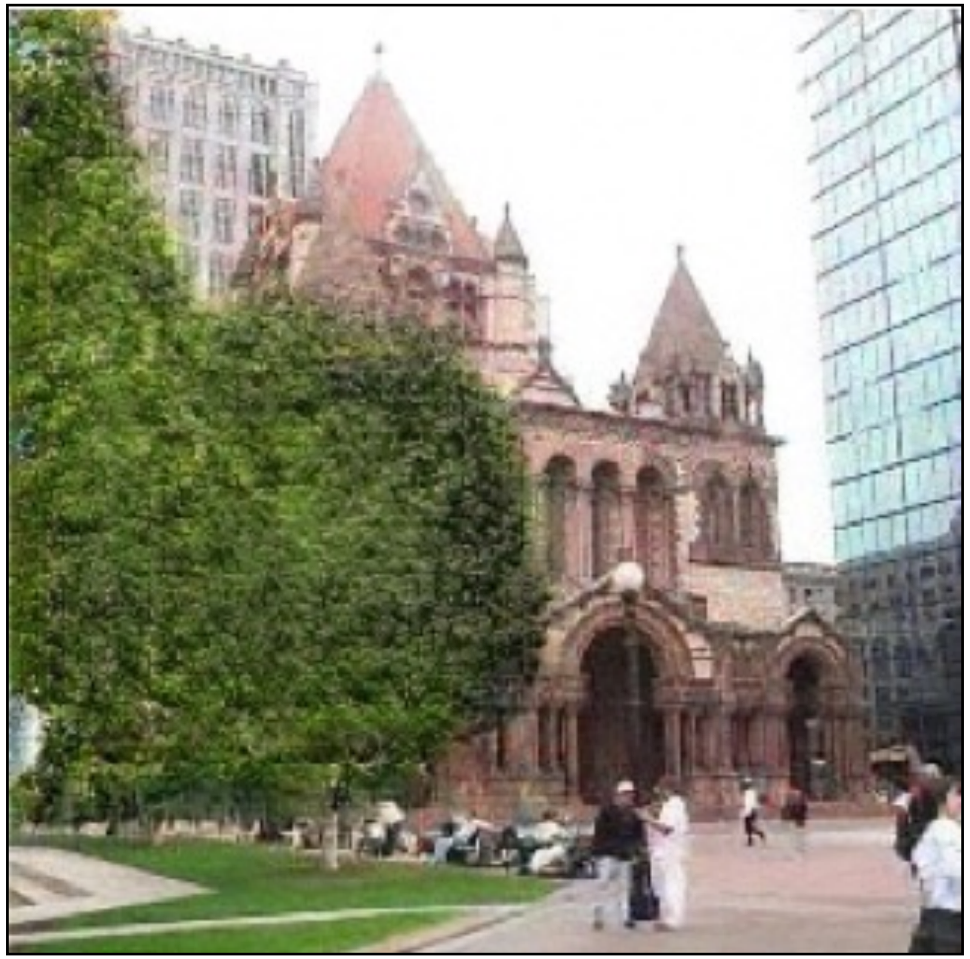
Add windows

Output result

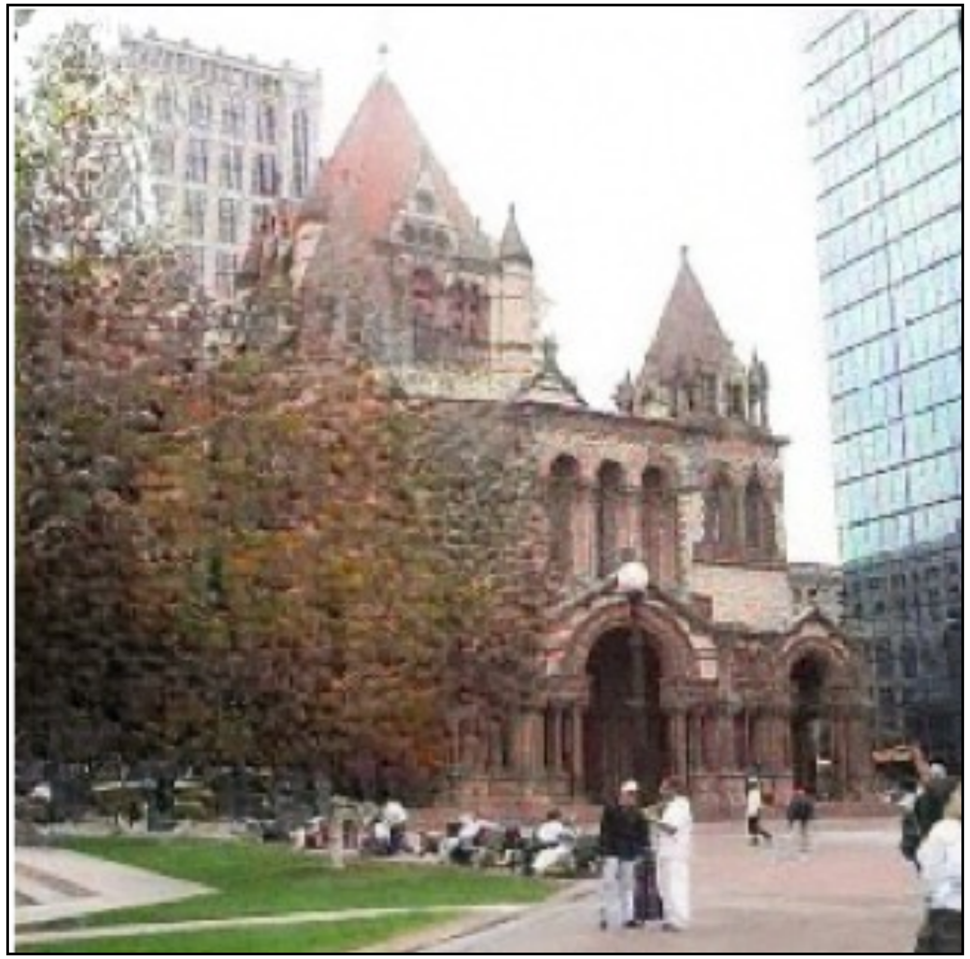
Semantic Photo Manipulation [Bau, Strobelt, Peebles, Wulff, Zhou, Zhu, Torralba, **SIGGRAPH** 2019]

Manipulating a Real Photo via GAN Dissection

Input image



Restyle trees for spring



Restyle trees for autumn

Semantic Photo Manipulation [Bau, Strobelt, Peebles, Wulff, Zhou, Zhu, Torralba, **SIGGRAPH** 2019]

GANBrush Drawing with Neurons

Upload your image: Choose File No file chosen

Draw:

undo reset

tree

grass

door

dome

sky

low med high