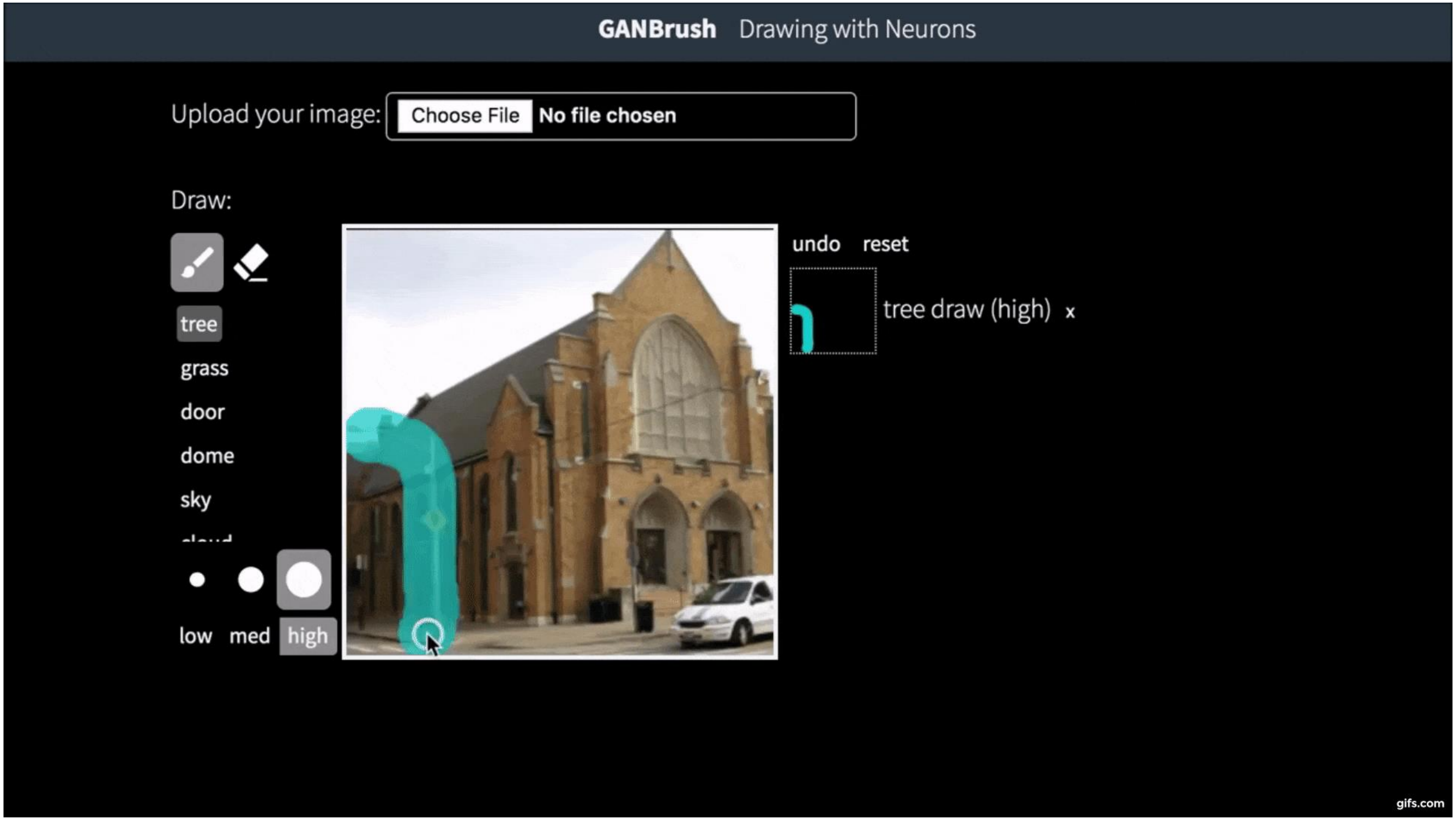
Semantic Photo Manipulation with a Generative Image Prior

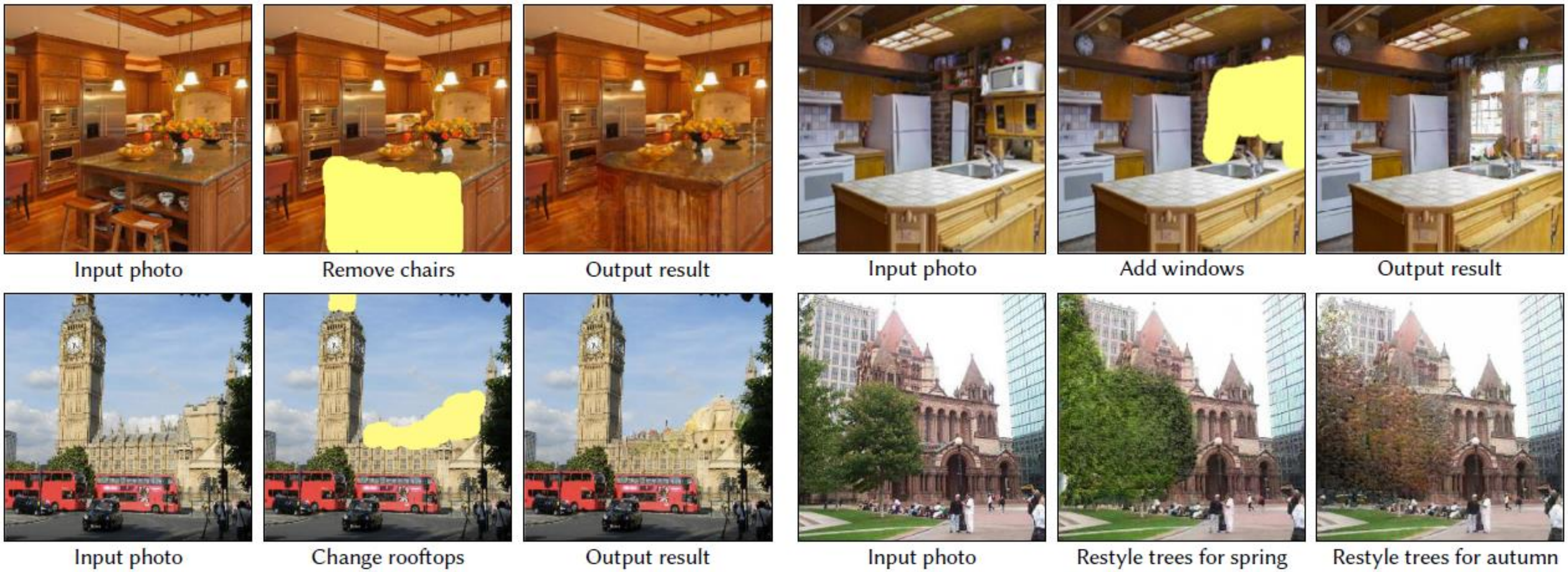
DAVID BAU, MIT CSAIL and MIT-IBM Watson AI Lab
HENDRIK STROBELT, IBM Research and MIT-IBM Watson AI Lab
WILLIAM PEEBLES, MIT CSAIL
JONAS WULFF, MIT CSAIL
BOLEI ZHOU, The Chinese University of Hong Kong
JUN-YAN ZHU, MIT CSAIL
ANTONIO TORRALBA, MIT CSAIL and MIT-IBM Watson AI Lab

要曙丽 2020/05/14

1. Introduction



1. Introduction



1. Introduction

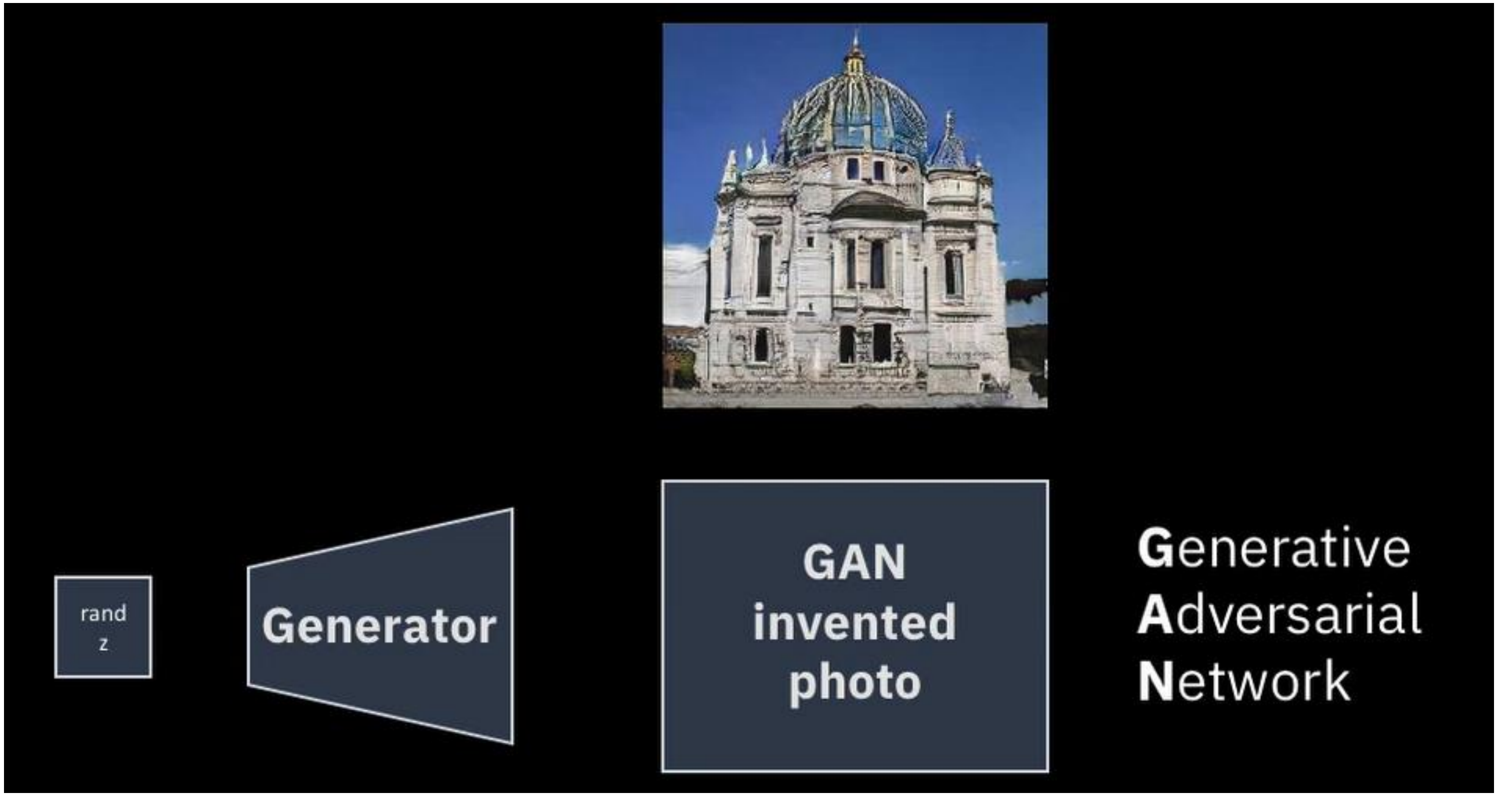
- Deep generative models
  - Provide latent semantic representations
  - Preserve image realism
  - Allow users to manipulate a photograph with abstract concepts
- Two technical challenges
  - It is hard for GANs to precisely reproduce an input image
  - The newly synthesized pixels often do not fit the original image after manipulation
- Present an image-specific adaptation method
  - Learn an image-specific generative model $G' \approx G$
  - G' produces new visual content, consistent with the original photo while reflecting semantic manipulations

2. Related Work

- Generative Adversarial Networks
  - Goodfellow et al. 2014, Karras et al. 2018, Bau et al. 2019…
  - Little work has used GANs for interactively manipulating an existing natural photograph
  - Some work manipulate a photo using GANs but only work with a single object at low resolutions (64x64) and often involve post-processing steps
- Interactive Photo Manipulation
  - An and Pellacini 2008, Tao et al. 2010, Zhang et al. 2016a…
  - Manual annotations of the object geometry and scene layout, choice of an appropriate object or RGBD data
- Deep Image Manipulation
  - Iizuka et al. 2017, Li et al. 2018, Kim and Park 2018…
  - Achieve high-quality results, but the editing task is fixed at training time and requires specific training data

3. Method

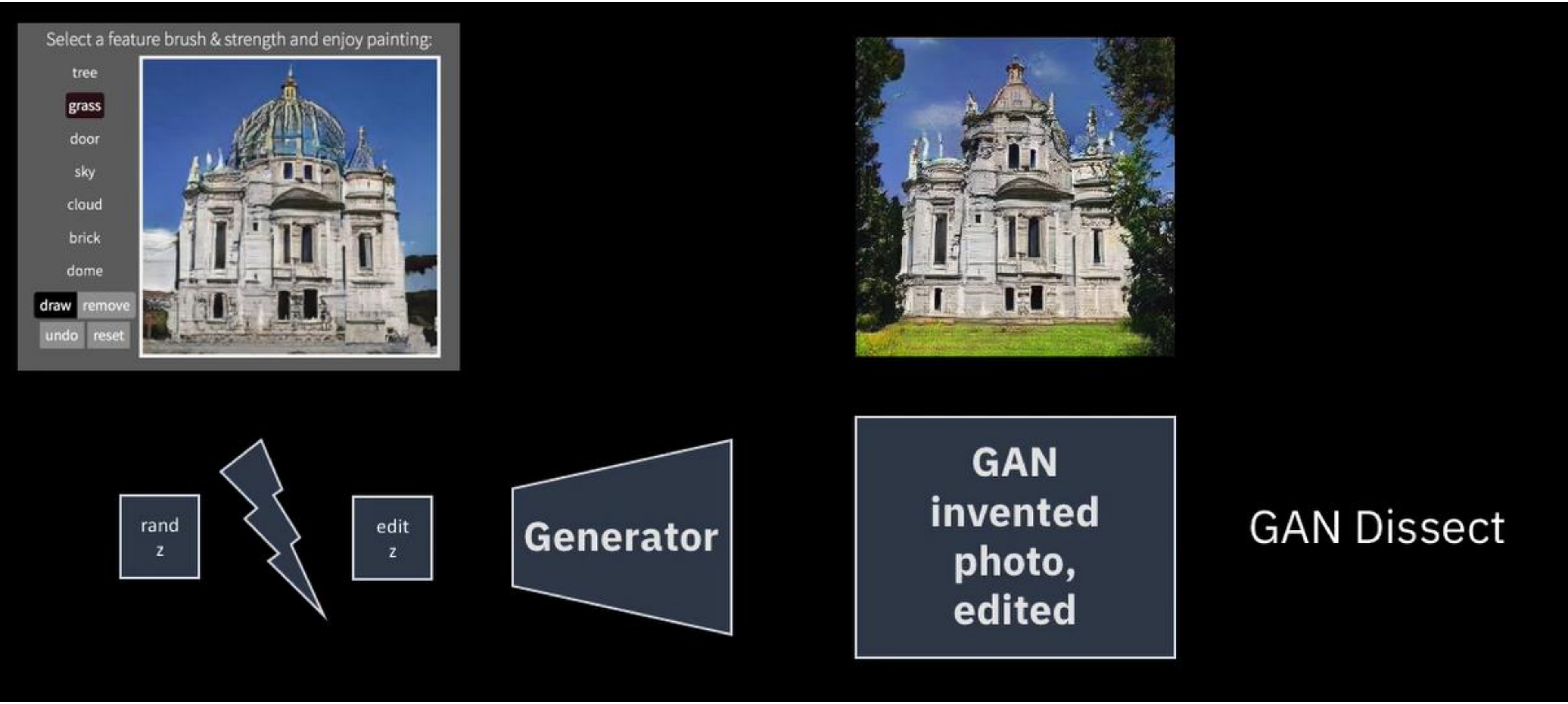
➢ Controllable Image Synthesis with GANs



$$G : z \rightarrow x$$

3. Method

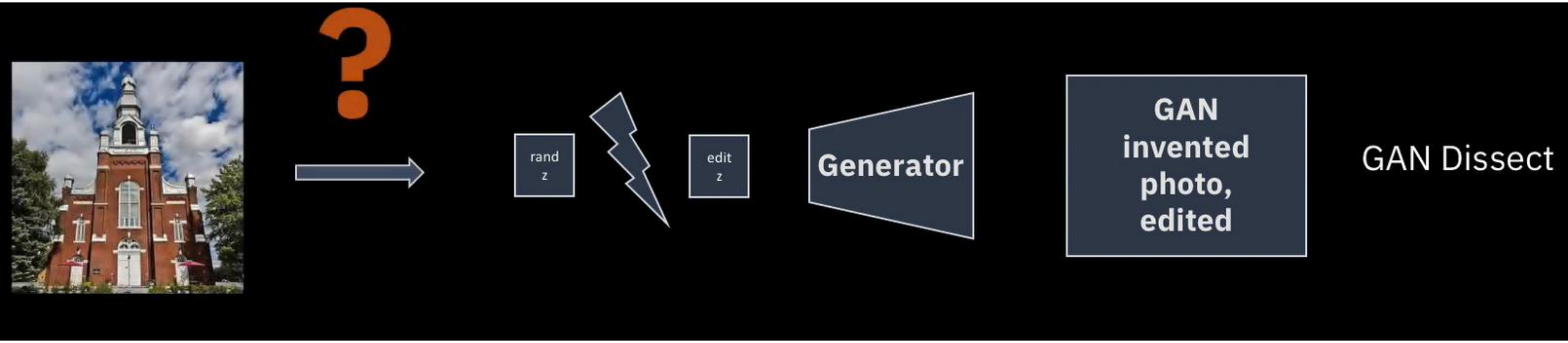
➢ Controllable Image Synthesis with GANs



$$z_e = \mathrm{edit}(z)$$

3. Method

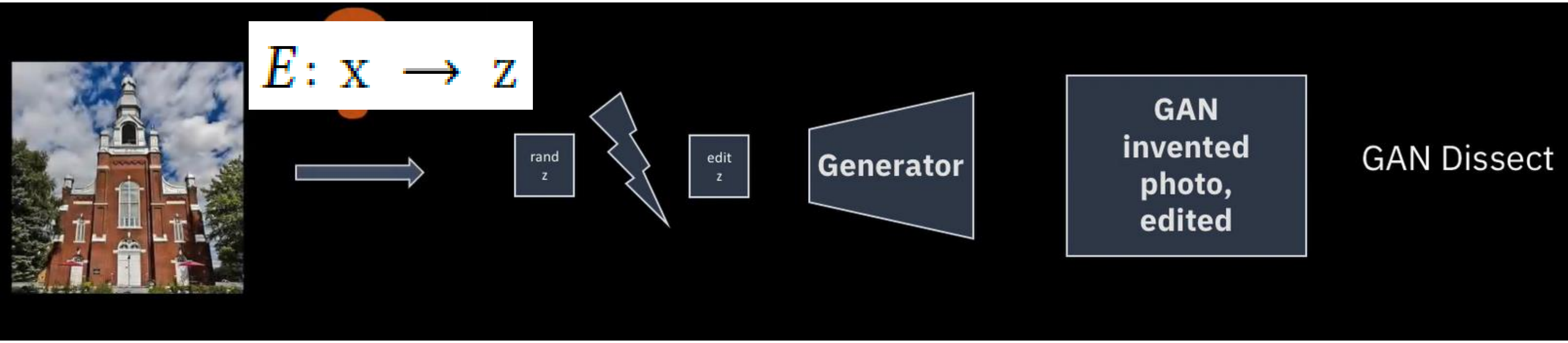
- Reproducing a Natural Image with a Generator



$$\mathcal{L}_r(\mathrm{x}, G(\mathrm{z})) = ||\mathrm{x} - G(\mathrm{z})||_1 + \lambda_{\mathrm{VGG}} \sum_{i=1}^{N} \frac{1}{M_i} ||F^{(i)}(\mathrm{x}) - F^{(i)}(G(\mathrm{z}))||_1$$

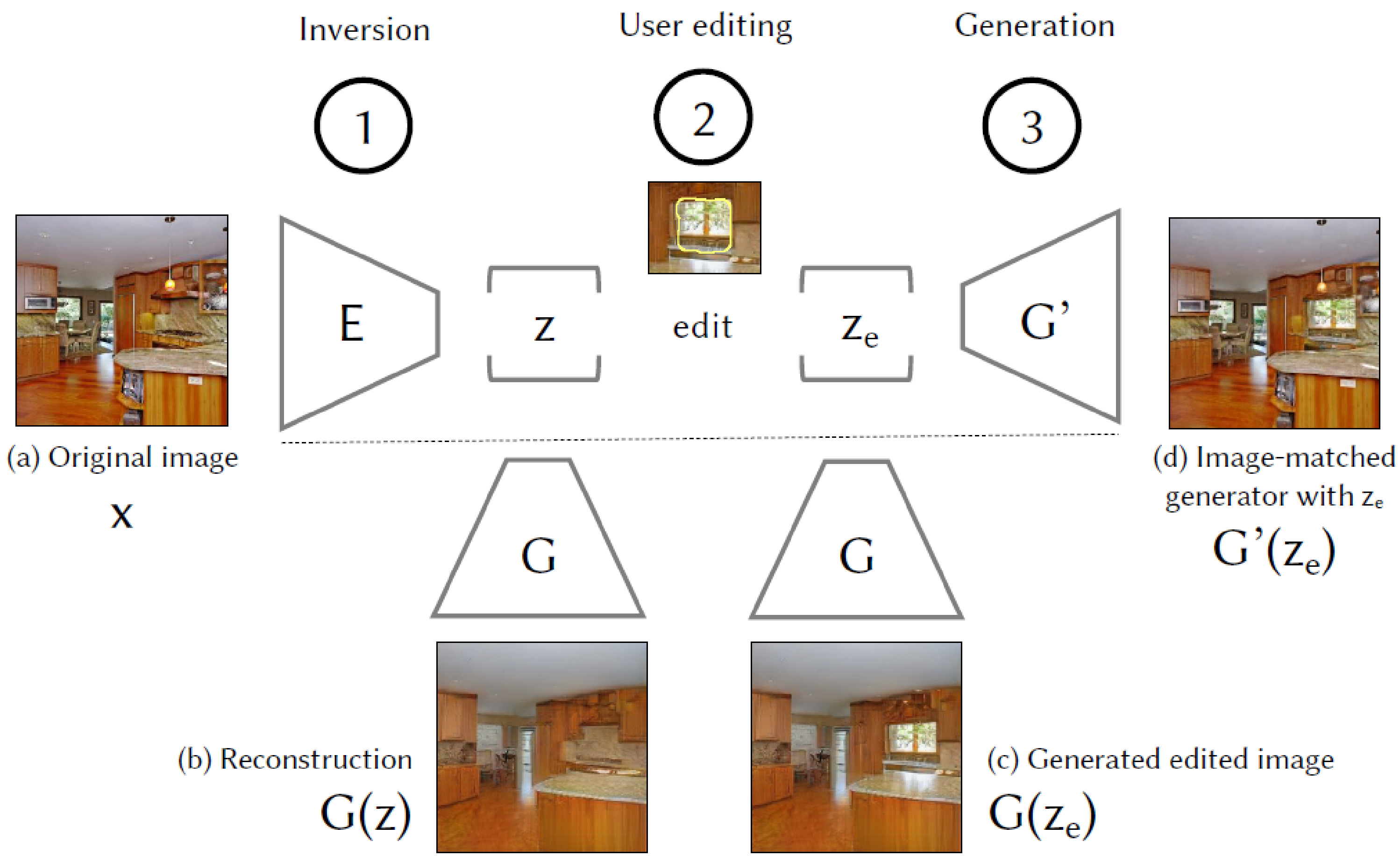
3. Method

➢ Reproducing a Natural Image with a Generator



$$\min_E \mathbb{E}_{\mathrm{x} \sim p_{data}(\mathrm{x})} \mathcal{L}_r(\mathrm{x}, G(E(\mathrm{x})))$$

3. Method


Inversion
User editing
Generation
1
2
3
E
z
edit
z_e
G'
(a) Original image
x
(d) Image-matched generator with z_e
$G'(z_e)$
G
G
(b) Reconstruction
$G(z)$
(c) Generated edited image
$G(z_e)$

3. Method

- Image-Specific Adaptation : Image-Specific Generator G′
  - G′ can produce a near-exact match for our input image x
  - G′ should be close to G so that they share an underlying semantic representation
- G′ can preserve the visual details of the original photo during semantic manipulations
- Given a user stroke binary mask:

$$\mathtt{mask}_e = \begin{cases} 1 & \text{where the stroke is present} \\ 0 & \text{outside the stroke} \end{cases}$$

- Minimizing a simple difference between the input image x and those generated by G′(z_e), summed over the image regions outside of the strokes

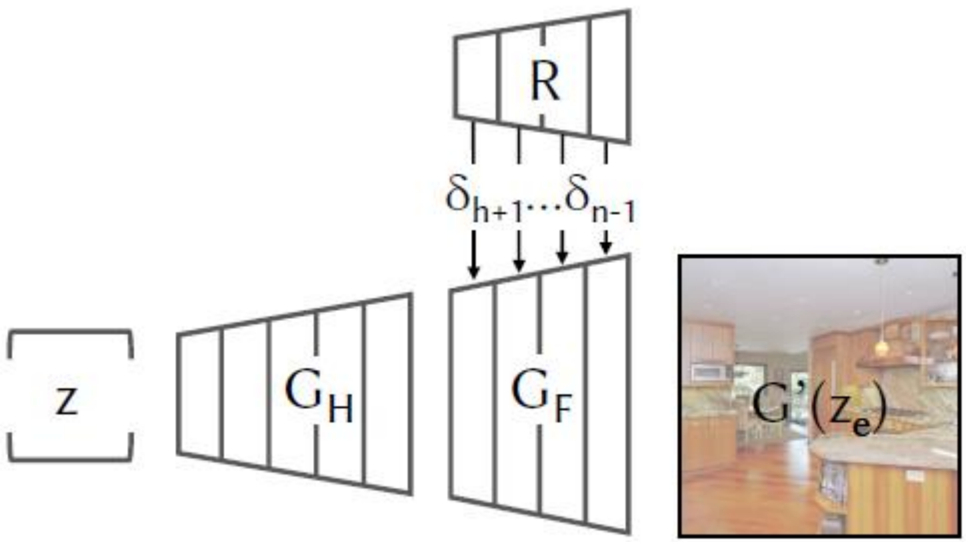
$$\mathcal{L}_{\text{match}} \equiv ||(G'(z_e) - x) \odot (1 - \mathtt{mask}_e)||_1$$

3. Method

➢ Preserving Semantic Representation

$$G(z) = g_n(g_{n-1}(\cdots(g_1(z)).\cdots))$$

$$z_h \equiv G_H(z) \equiv g_h(g_{h-1}(\cdots g_1(z)\cdots))$$
$$G_F(z_h) \equiv g_n(g_{n-1}(\cdots(g_{h+1}(z_h)\cdots))$$



$$G'_F(z_h) \equiv g_n((1+\delta_{n-1}) \odot g_{n-1}(\cdots((1+\delta_{h+1}) \odot g_{h+1}(z_h)\cdots)))$$
$$G'(z) \equiv G'_F(G_H(z)). \quad (6)$$

3. Method

- To further prevent overfitting, we add a regularization term to penalize large perturbations:

$$\mathcal{L}_{\text{reg}} \equiv \sum_{i=h+1}^{n-1} ||\delta_i||^2$$

- Overall optimization:

$$\mathcal{L} = \mathcal{L}_{\text{match}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}.$$

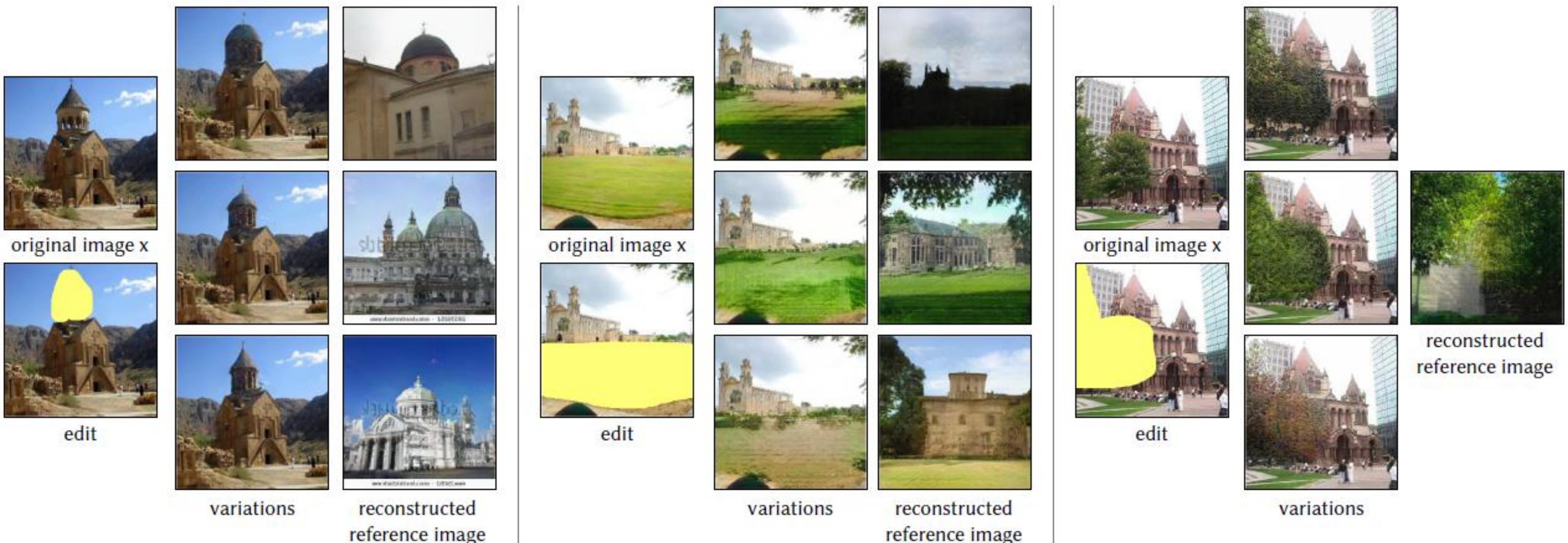
3. Method

- Semantic Editing Operations: GANPaint
  - Adding and removing objects $\alpha_c = (i_c \otimes U) \in \mathbb{R}^{8\times8\times512}$

$$z_e := \underbrace{(1-\alpha_c)\odot z}_{\text{activations retained from } z} + \underbrace{\alpha_c \odot (s\,p_c)}_{\text{edited activations}}$$

3. Method

- Semantic Editing Operations: GANPaint
  - Changing the appearance of objects



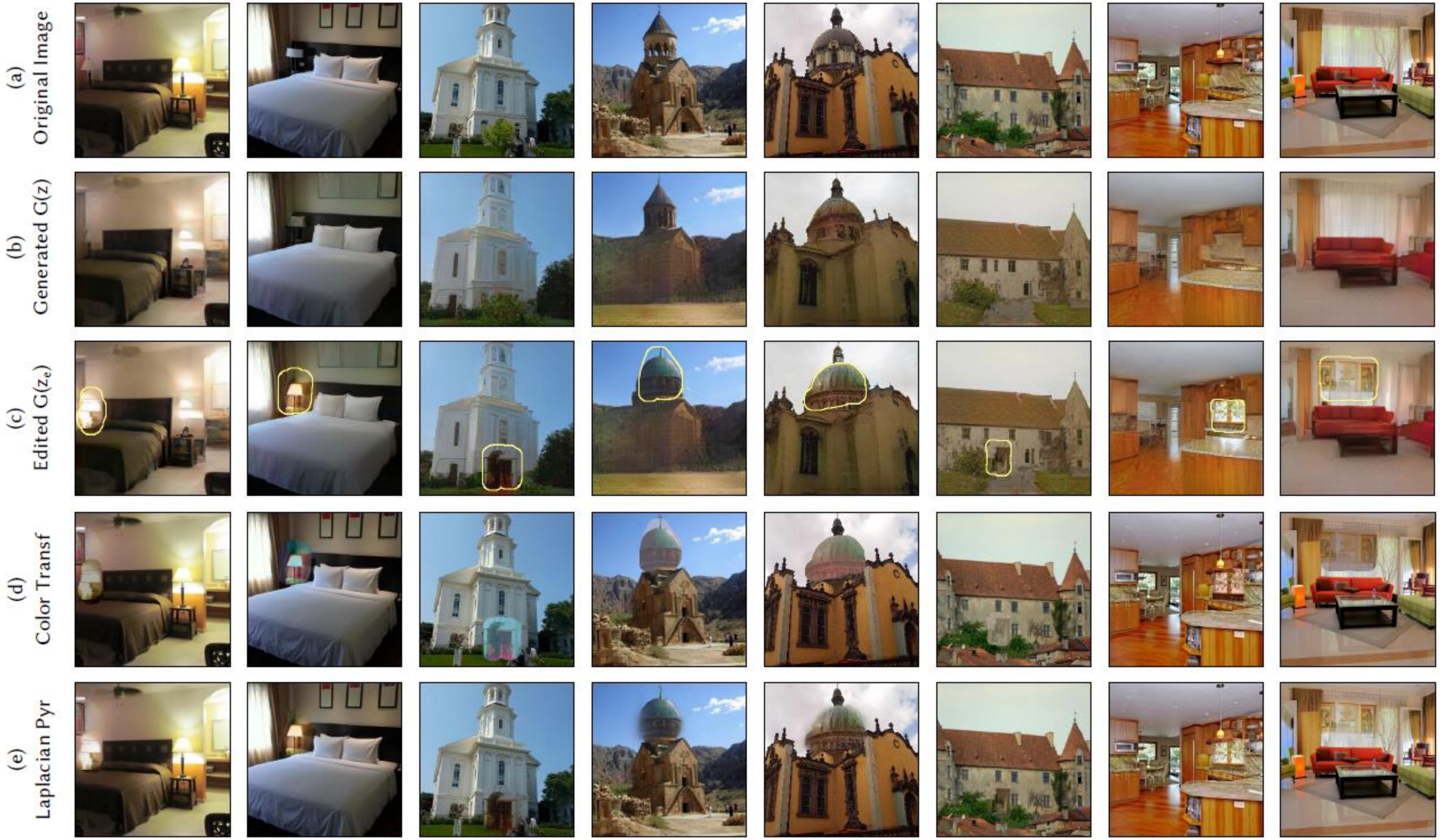
(a) Dome Appearance Changing

(b) Grass Appearance Changing

(c) Tree Appearance Changing varying only strength

4. Experiments

➢ Comparing Image-Specific Adaptation to Compositing



4. Experiments

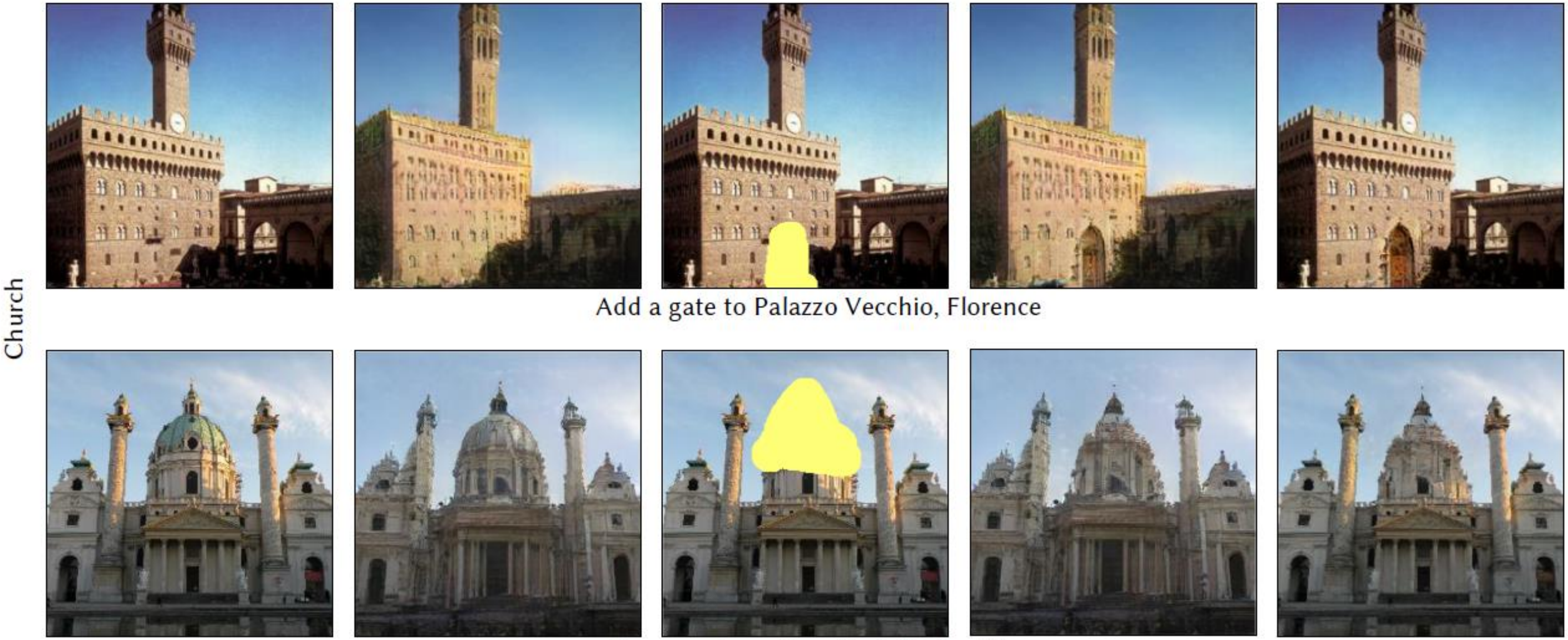
➢ Ablation Studies

Table 1. AMT evaluation of compositing methods compared to our method: we report the percentage of users that prefer various other methods over ours. Our method is also compared to the unadapted generator G as well as a directly adapted generator G'_w in which the weights have been fitted so $G'_w(z) \approx x$.

Method	% prefer vs ours
Color transfer [Reinhard et al. 2001]	16.8%
Error-tolerant image compos. [Tao et al. 2010]	43.6%
Poisson blending [Pérez et al. 2003]	44.2%
Laplacian pyramid blending	47.2%
Our method	50.0%
$G(z_e)$ without adaptation	37.4%
$G'_w(z_e)$, weights are fitted so $G'_w(z) \approx x$	33.1%

4. Experiments

➢ Qualitative Results

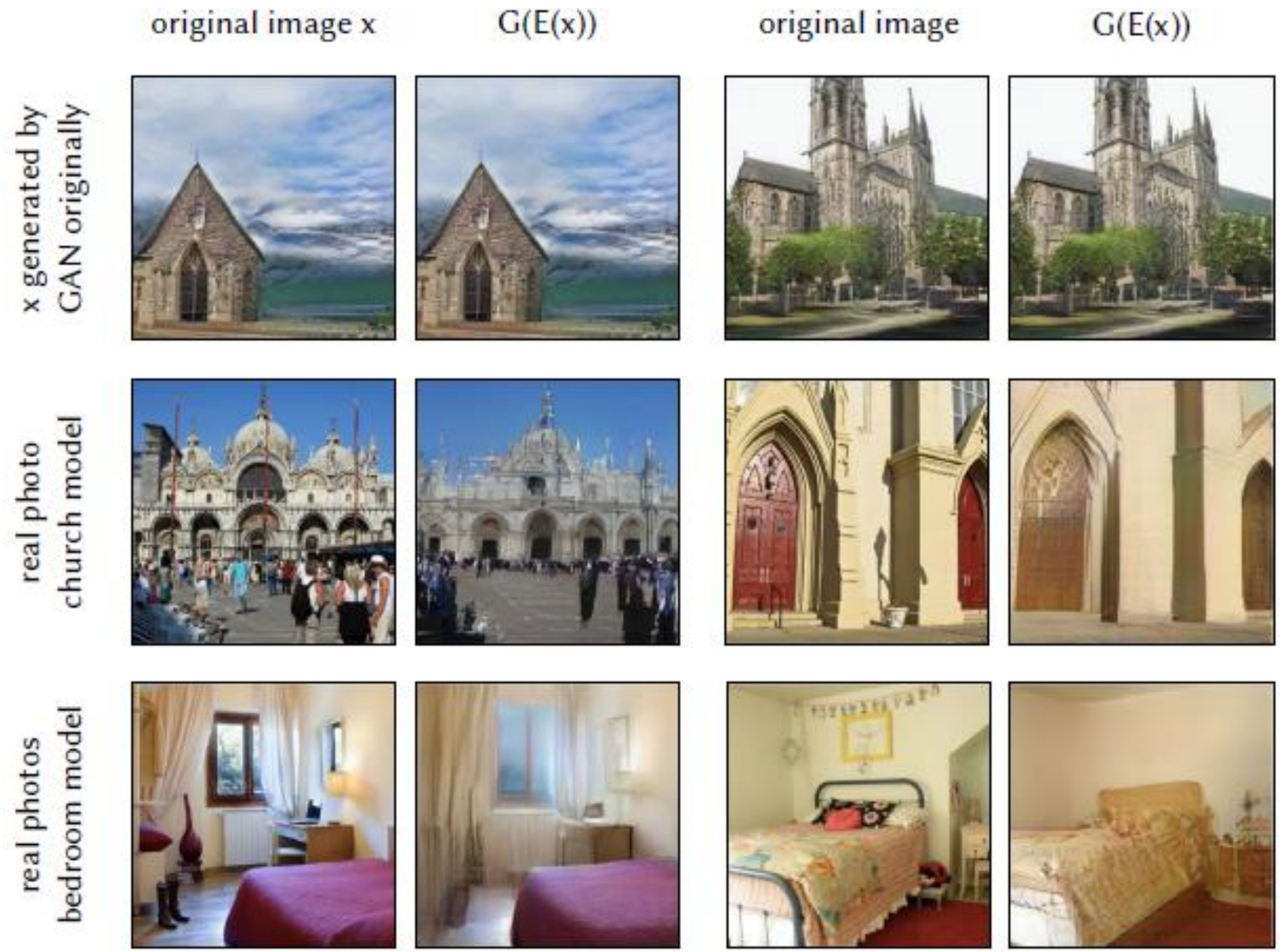


Add a gate to Palazzo Vecchio, Florence

Remove dome from Karlskirche, Vienna

4. Experiments

➢ Recovering the Latent Vector z



5. Conclusion

- Require an optimization be run after each edit, which takes about 30 seconds on a modern GPU
- Latent spaces learned by deep neural networks are not fully disentangled
- The quality and resolution of our current results are still limited

Thanks!