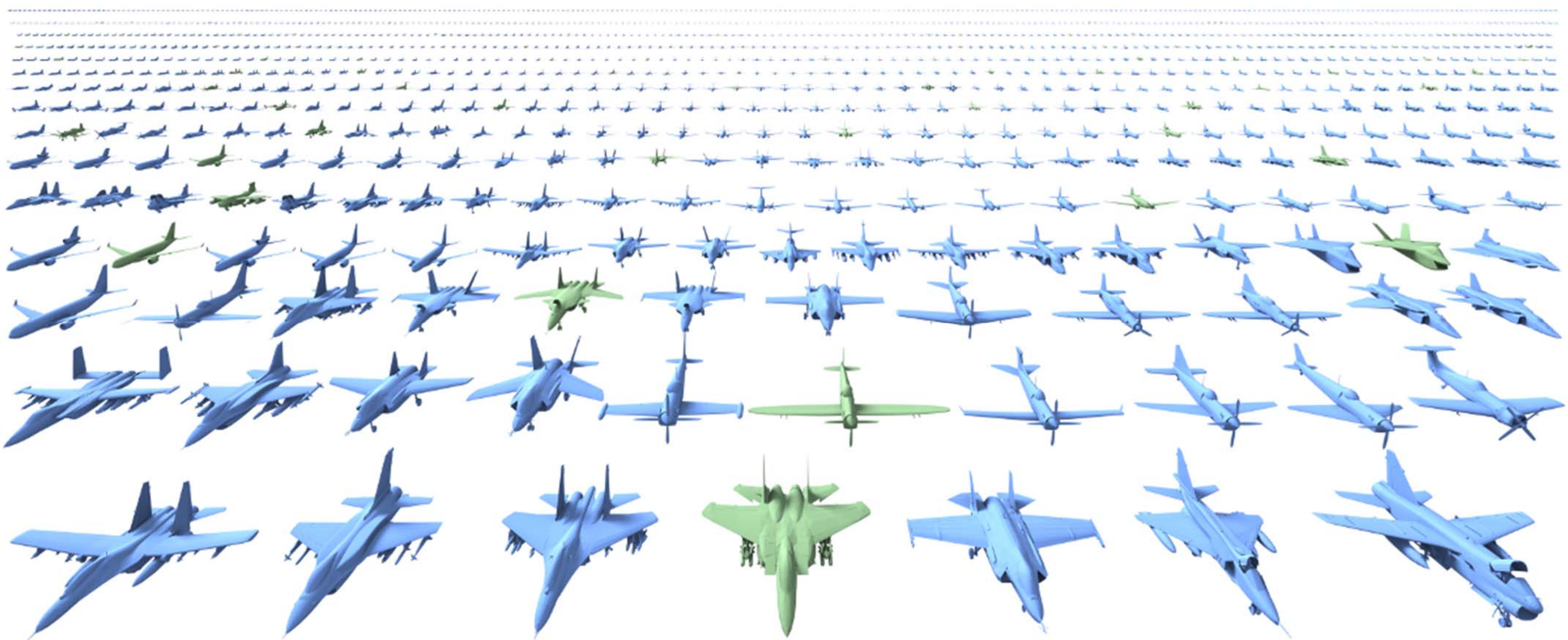
3D Generative models: GANs



Evangelos Kalogerakis –
574/674

How to generate shapes/scenes?

- Encoder-Decoders
  - ➢ Case Study: Multi-view decoder
  - ➢ Case Study: Implicit decoder
  - ➢ Case Study: Patch decoder
  - ➢ Case Study: Mesh Decoder
- **Generative Adversarial Networks**
  - – Case Study: 3D Volumetric GAN
  - – Case Study: Get3D
- Variational Autoencoders
- Autoregressive models
- Diffusion models

GAN basic idea

Learn to generate data (from scratch) based on the underlying distribution of the training set

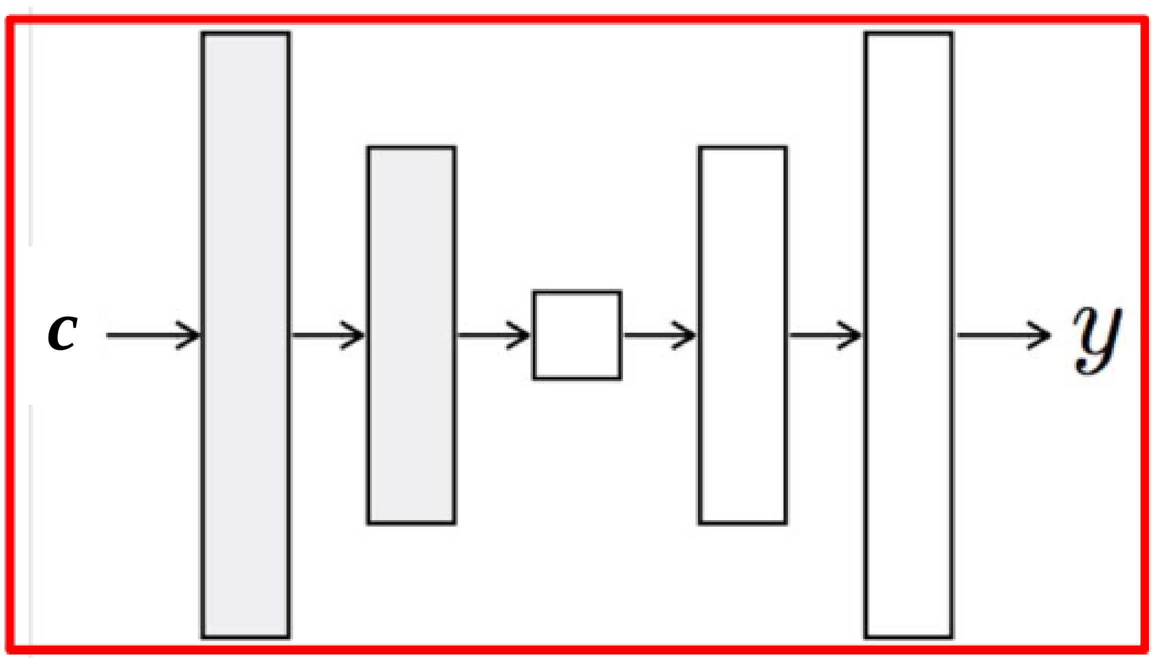
Highly successful in image synthesis

BigGAN (2018)

GAN basic idea

We need an architecture that can generate data

Previously, we have seen translation networks:

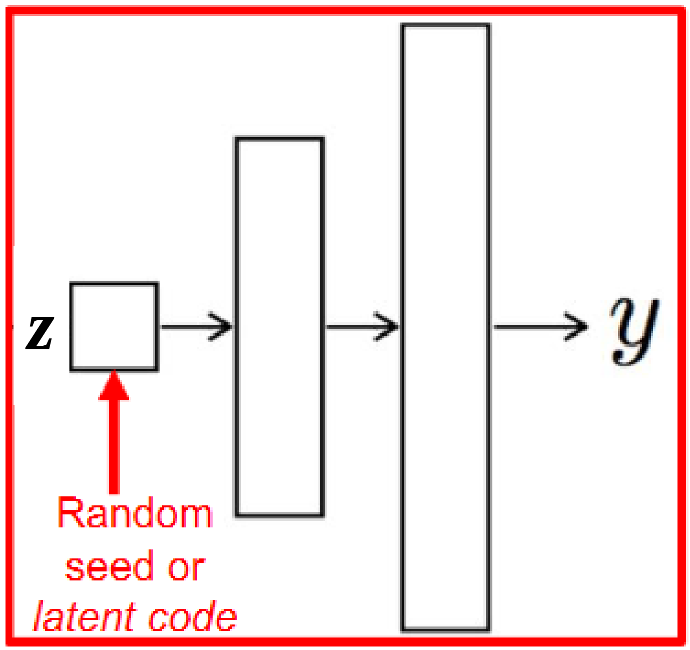


(e.g. input image c, output SDF/mesh/patches y)

GAN basic idea

Start with a random vector (noise) $\boldsymbol{z}$!

You may use any 3D decoder we discussed before!



GAN basic idea

From: https://github.com/marian42/shapegan

Training data from p_{data}

Generated samples p_{model}

We want to learn p_{model} that matches p_{data}

‘Classical’ GAN

Train two networks with **opposing objectives:**

- **Generator:** learns to generate samples
- **Discriminator:** learns to distinguish between generated and real samples

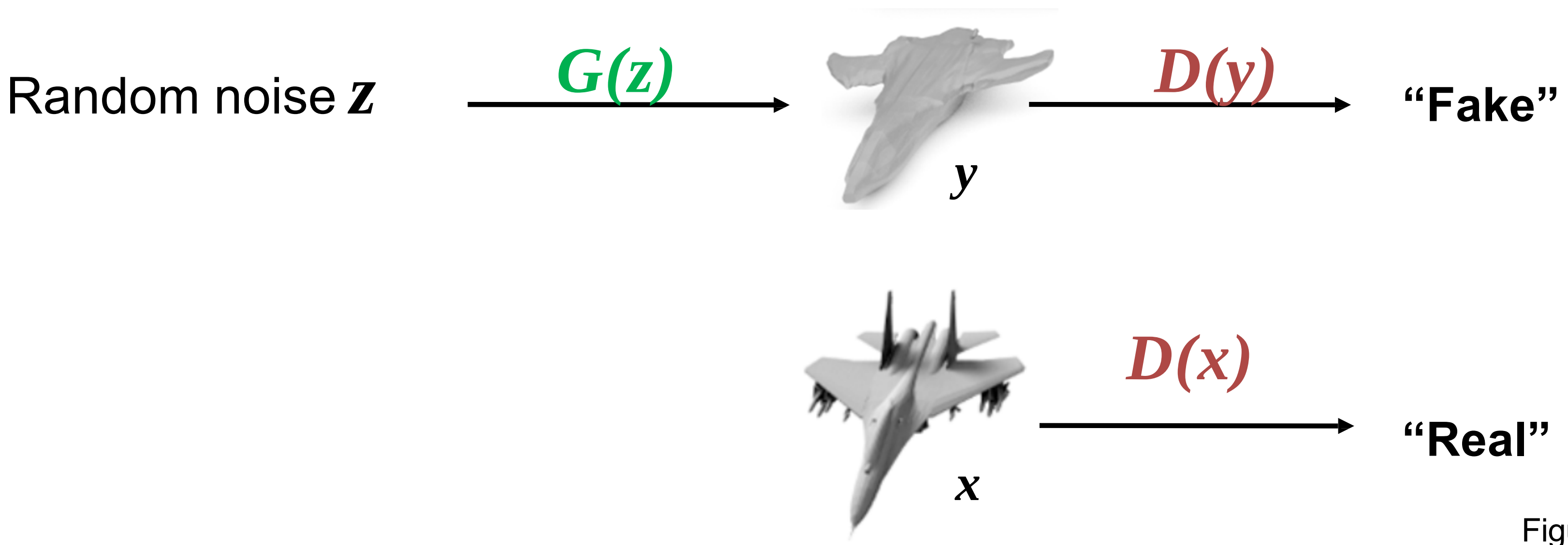


Figure adapted from [F. Fleuret]

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, NIPS 2014

GAN objective

The discriminator D should output the probability that the sample x is real i.e., we want $\mathrm{D}(\mathrm{x}) \approx 1$ for **real data** and

$$\mathrm{D}(\mathrm{y}) = \mathrm{D}(\mathrm{G}(\mathrm{z})) \approx 0$$ for **fake data**

Expected conditional log likelihood for real and generated data:

$$\mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z \sim p} \log(1 - D(G(z)))$$

We seed the generator with noise z drawn from a simple distribution p (Gaussian or uniform)

GAN objective

$$V(G, D) = \mathbb{E}_{x\sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z\sim p} \log(1 - D(G(z)))$$

- The discriminator wants to correctly distinguish real and fake samples:

$$D^* = \arg\max_D V(G, D)$$

- The generator wants to fool the discriminator:

$$G^* = \arg\min_G V(G, D)$$

- Train the generator and discriminator jointly in a *minimax game*

GAN training

$$V(G,D) = \mathbb{E}_{x\sim p_{\text{data}}} \log D(x) + \mathbb{E}_{z\sim p}\log(1 - D(G(z)))$$

Alternate between:

- *Gradient ascent* on discriminator:

$$D^* = \arg\max_D V(G,D)$$

- *Gradient descent* on generator (minimize log-probability of discriminator being right):

$$G^* = \arg\min_G V(G,D)$$
$$= \arg\min_G \mathbb{E}_{z\sim p}\log(1 - D(G(z)))$$

- In practice, do *gradient ascent* on generator (maximize log-probability of discriminator being wrong):

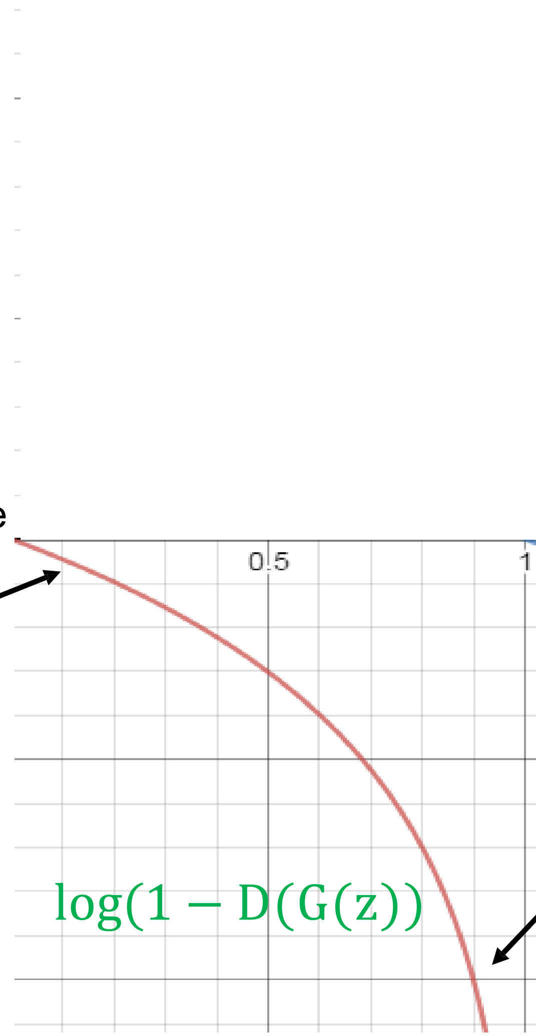
$$G^* = \arg\max_G \mathbb{E}_{z\sim p}\log(D(G(z)))$$

GAN training

$$\min_G \; \mathbb{E}_{z\sim p} \log(1 - D(G(z))) \text{ vs. } \max_G \; \mathbb{E}_{z\sim p} \log(D(G(z)))$$

Low discriminator score
(low-quality samples)

High discriminator score
(high-quality samples)



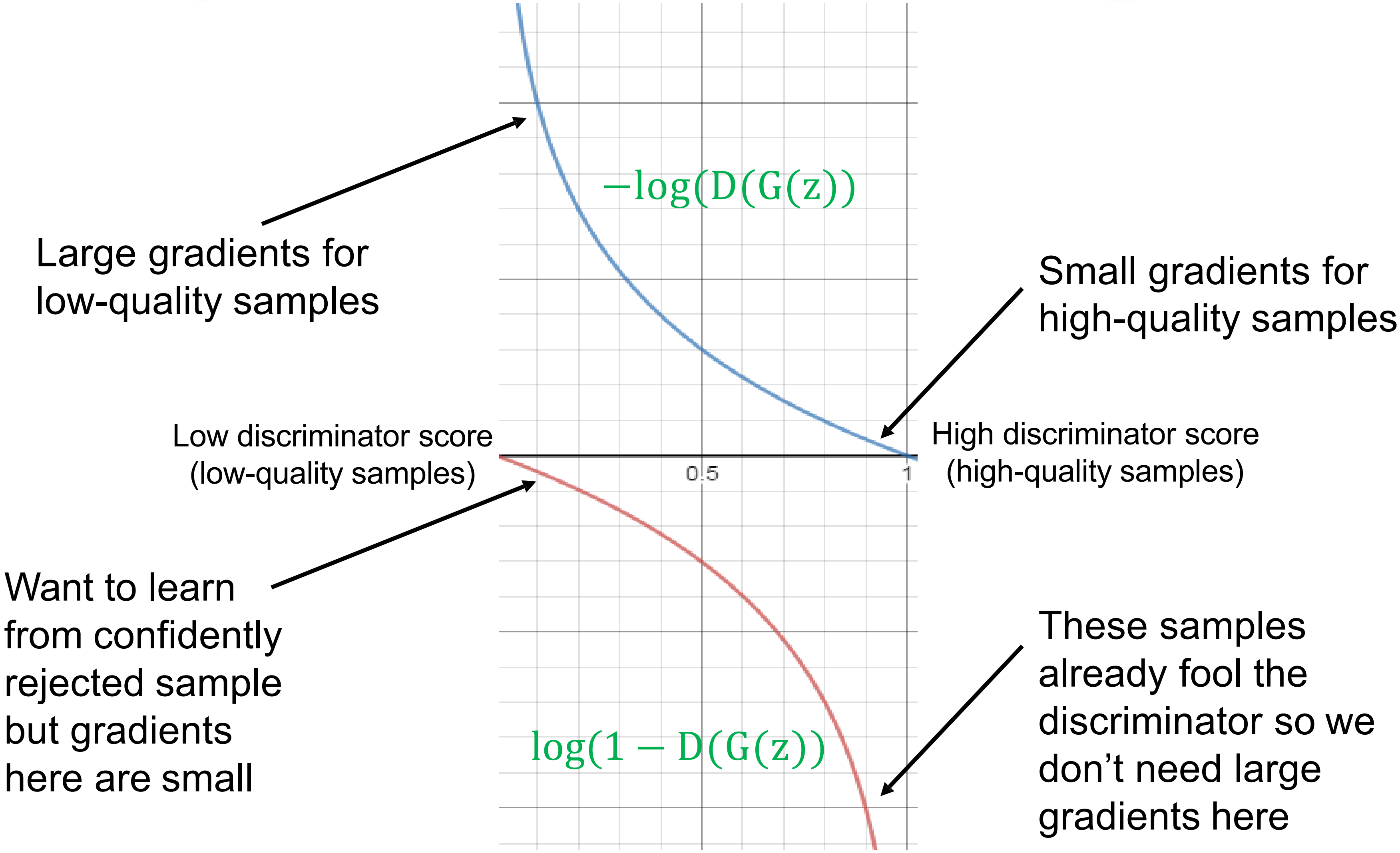
Want to learn from confidently rejected sample but gradients here are small

These samples already fool the discriminator so we don't need large gradients here

https://cs.uwaterloo.ca/%7Emli/Deep-Learning-2017-Lecture7GAN.ppt

GAN training

$\min_G \ \mathbb{E}_{z\sim p} \log(1 - D(G(z)))$ vs. $\max_G \ \mathbb{E}_{z\sim p} \log(D(G(z)))$



https://cs.uwaterloo.ca/%7Emli/Deep-Learning-2017-Lecture7GAN.ppt

GAN training

Update discriminator

- Repeat the following steps:
  - Sample mini-batch of noise samples $z_1, z_2, z_3, \ldots, z_m$ and mini-batch of real samples $x_1, x_2, x_3, \ldots, x_m$
  - Update parameters of **D** by stochastic gradient ascent:

$$\frac{1}{m}\sum_{m}[\log D(x_m) + \log(1 - D(G(z_m)))]$$

Update generator

Sample mini-batch of noise samples $z_1, z_2, z_3, \ldots, z_m$

- Update parameters of **G** by stochastic gradient ascent on

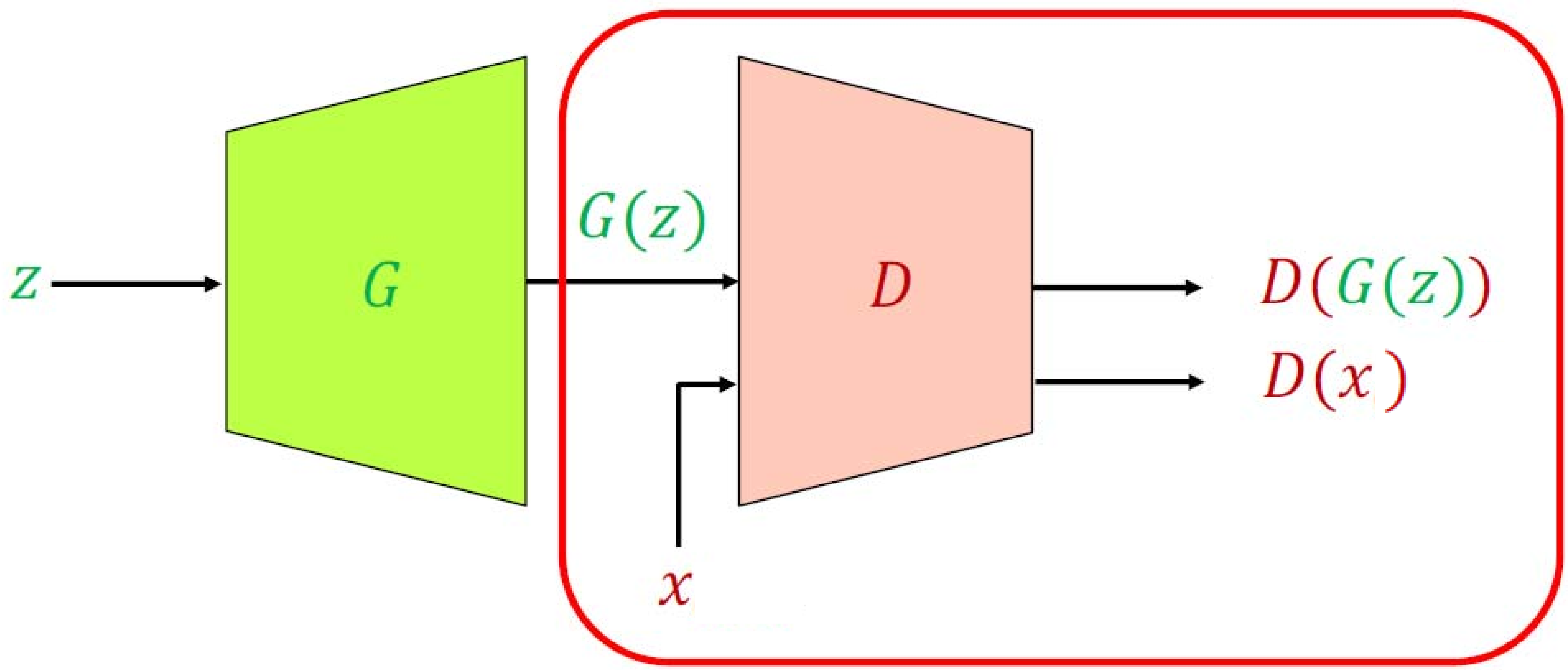
$$\frac{1}{m}\sum_{m}\log D(G(z_m))$$

... repeat discrim. & gen. training ... until happy with results

GAN conceptual picture

Update discriminator: push $\mathrm{D(x)}$ close to 1 and $\mathrm{D(G(z))}$ close to 0

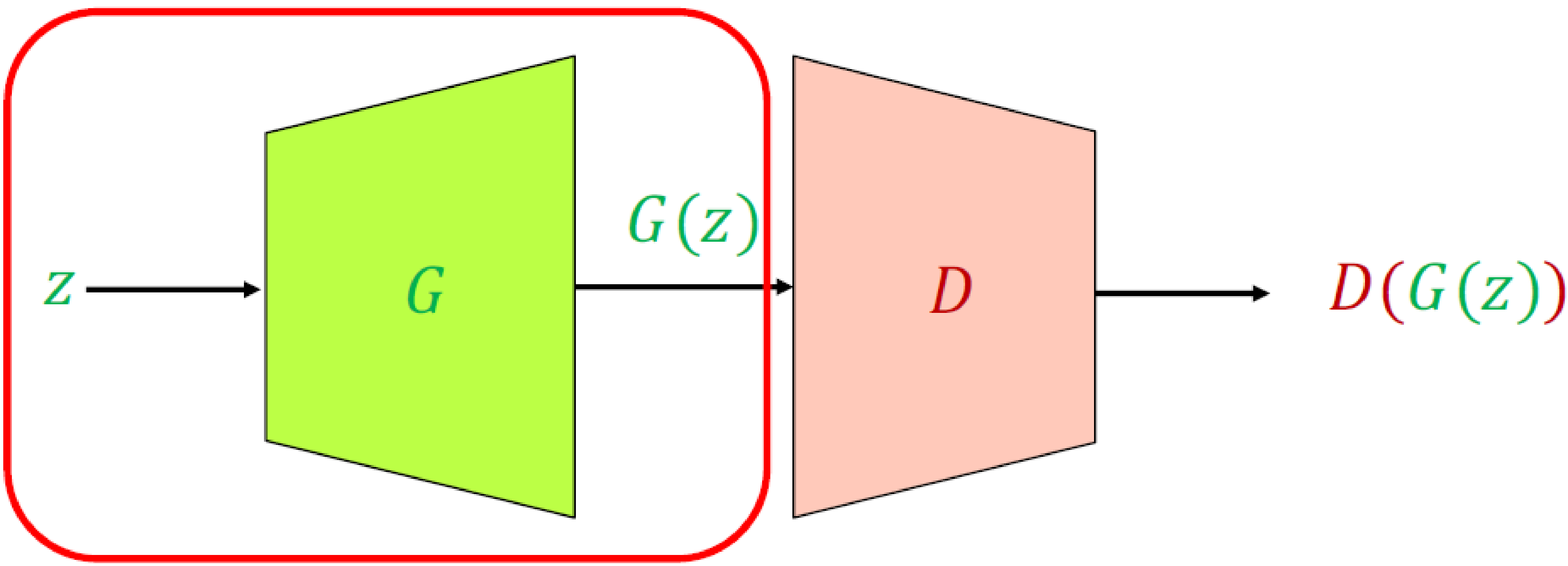
- The generator is a “black box” to the discriminator



GAN conceptual picture

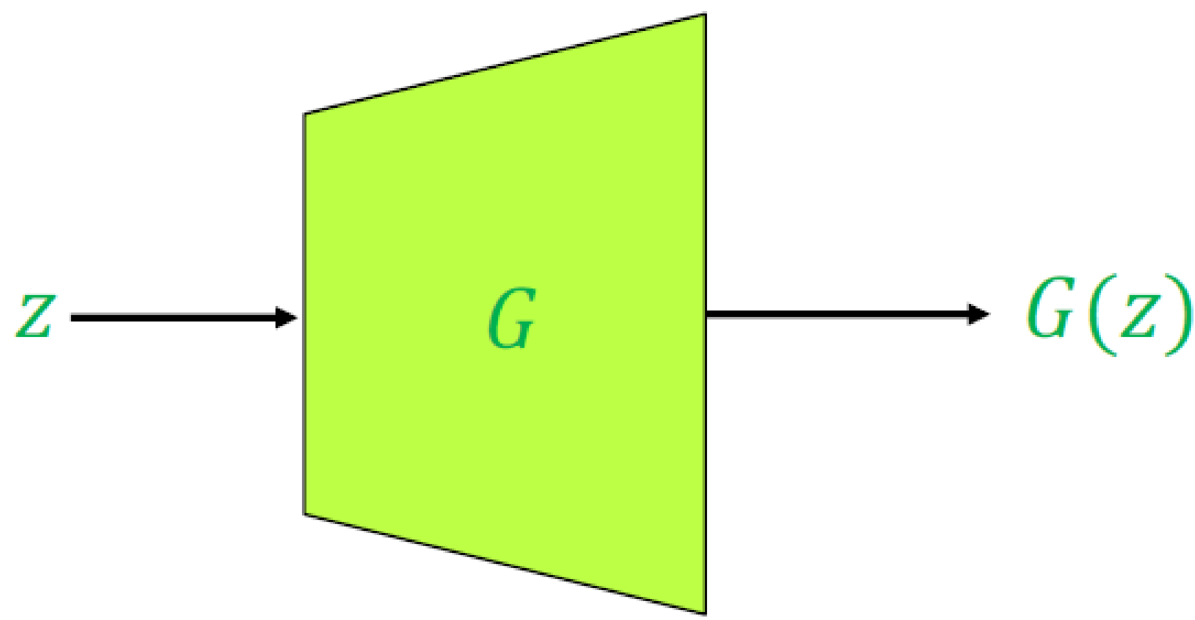
Update generator: increase $D(G(z))$

- Requires back-propagating through the composed generator-discriminator network (i.e., the discriminator cannot be a black box)
- The generator is exposed to real data only via the output of the discriminator (and its gradients)



GAN conceptual picture

Test time:



Problems with GAN training

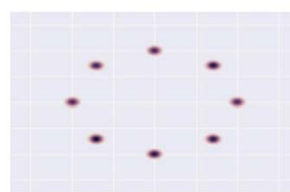
Stability

- Parameters can oscillate or diverge
- Behavior very sensitive to hyper-parameter selection

Mode collapse

- Generator ends up modeling only a small subset of the training data.

Target



Output

Metz, Luke, et al. **"Unrolled Generative Adversarial Networks."** arXiv preprint arXiv:1611.02163 (2016).

Wasserstein GAN

Motivated by *Wasserstein or Earth mover's distance*, for comparing distributions

- In practice, simply drop the sigmoid from the discriminator:

$$\min_G \max_D \left[\mathbb{E}_{x\sim p_{\text{data}}} D(x) - \mathbb{E}_{z\sim p} D(G(z))\right]$$

- Need to also clip weights to a fixed range after each gradient update to promote stability

M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, ICML 2017

How to generate shapes/scenes?

- Encoder-Decoders
  - ➢ Case Study: Multi-view decoder
  - ➢ Case Study: Implicit decoder
  - ➢ Case Study: Patch decoder
  - ➢ Case Study: Mesh Decoder

You may use any of these decoders as generators in GANs, together with a corresponding 3D network (multi-view, volumetric, point-based, graph-based) as a discriminator

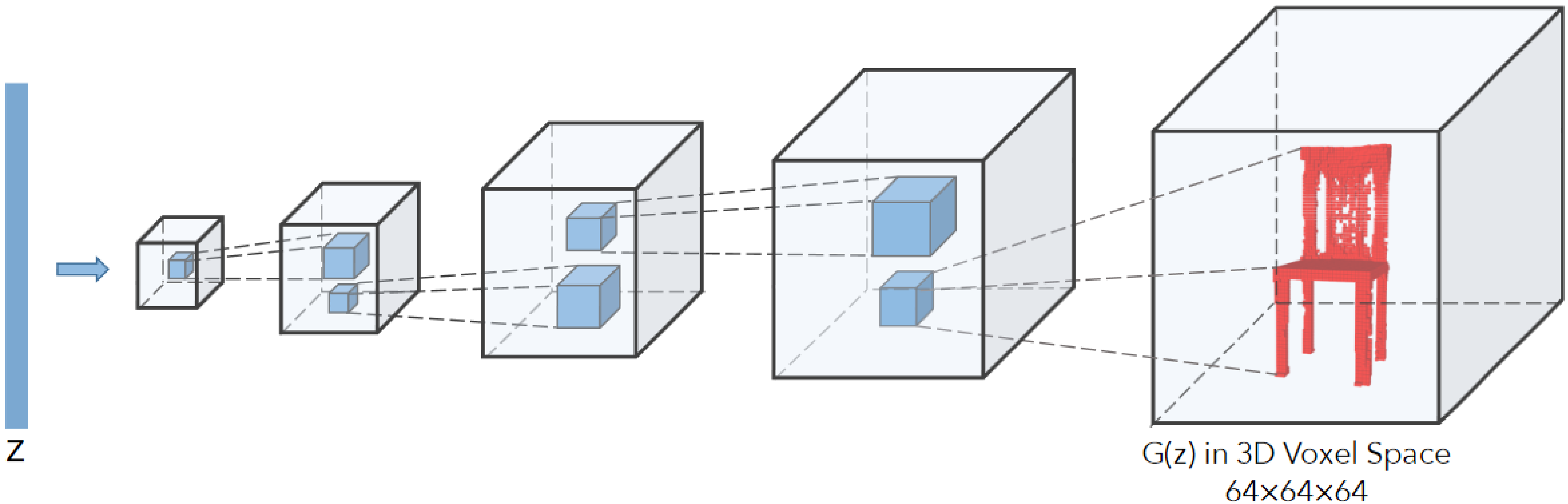
- **Generative Adversarial Networks**
  - – Case Study: 3D Volumetric GAN
  - – Case Study: Get3D
- Variational Autoencoders
- Autoregressive models
- Diffusion models

How to generate shapes/scenes?

- Encoder-Decoders
  - ➢ Case Study: Multi-view decoder
  - ➢ Case Study: Implicit decoder
  - ➢ Case Study: Patch decoder
  - ➢ Case Study: Mesh Decoder
- Generative Adversarial Networks
  - – **Case Study: 3D Volumetric GAN**
  - – Case Study: Get3D
- Variational Autoencoders
- Autoregressive models
- Diffusion models

3D GANs

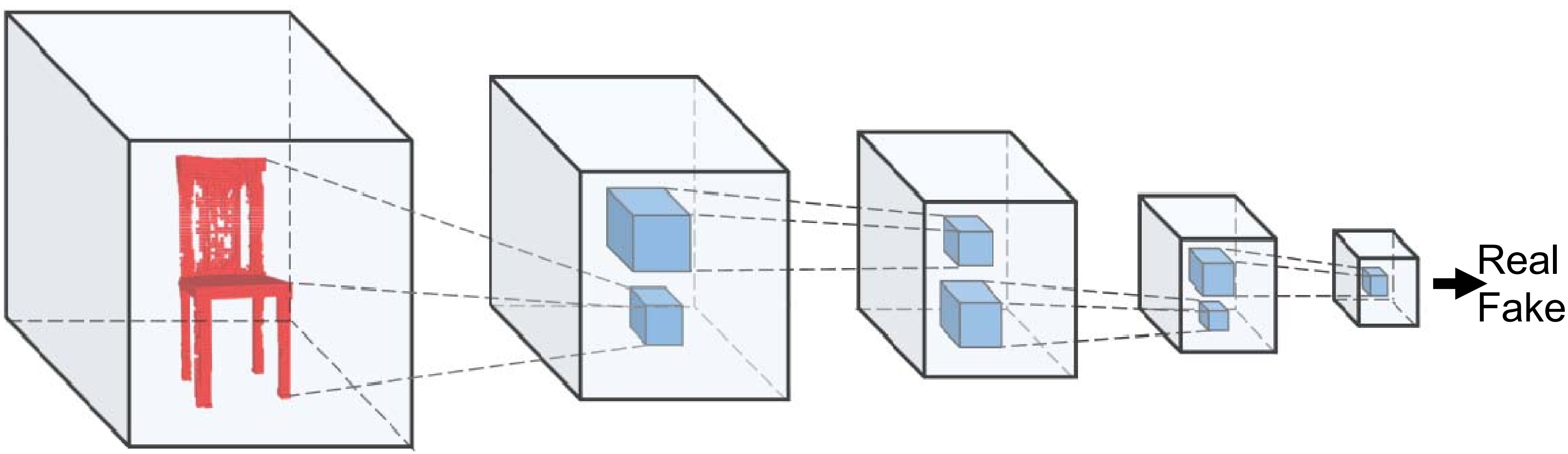
Uses a volumetric decoder to produce a dense grid with a series of transpose 3D convolution layers



Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling, Wu et al. 2016

3D GANs

The discriminator largely mirrors the generator (with a real/fake prediction in the end)

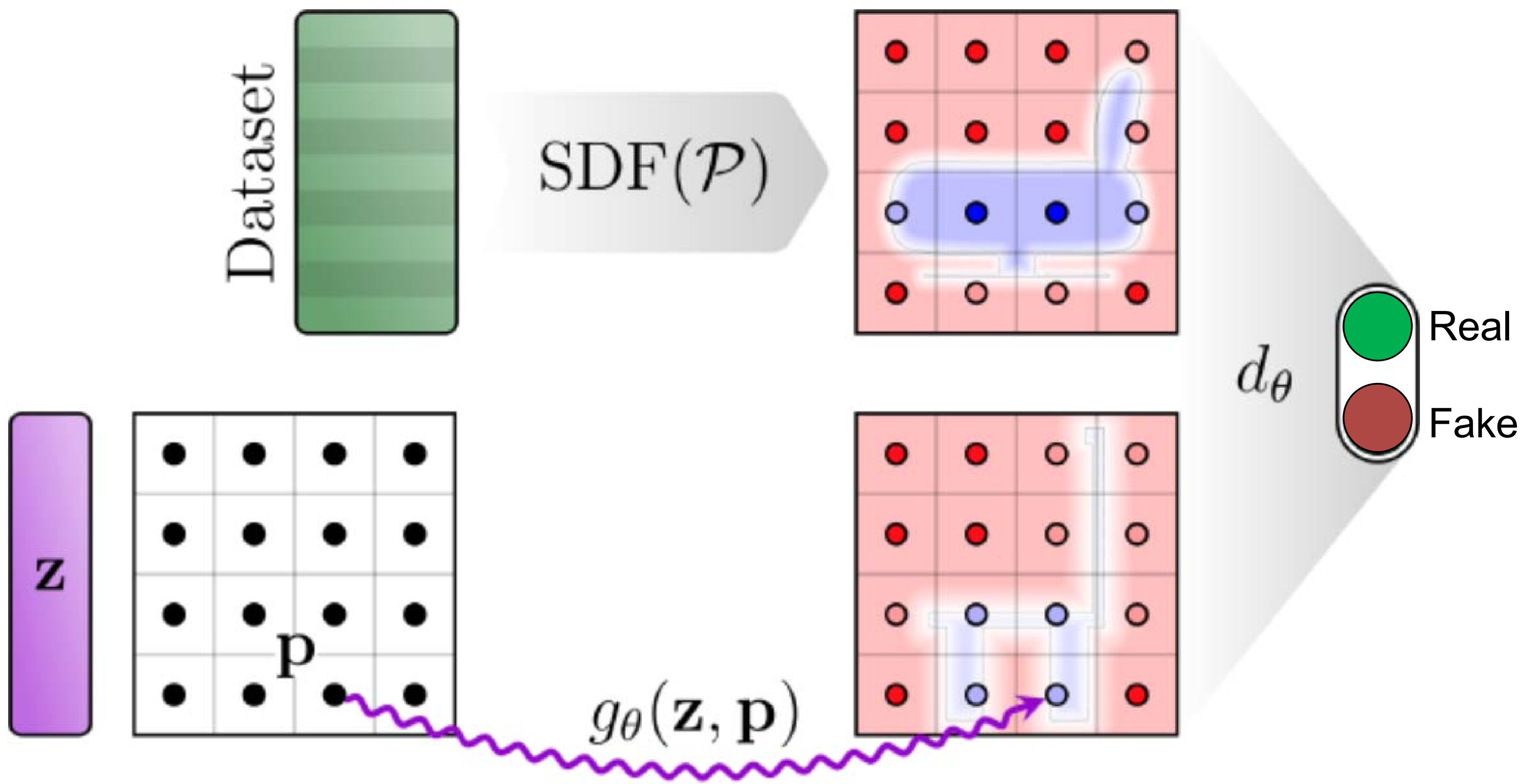


Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling, Wu et al. 2016

Results

Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling, Wu et al. 2016

Extensions



Adversarial Generation of Continuous Implicit Shape Representations, Kleineberg et al. 2020

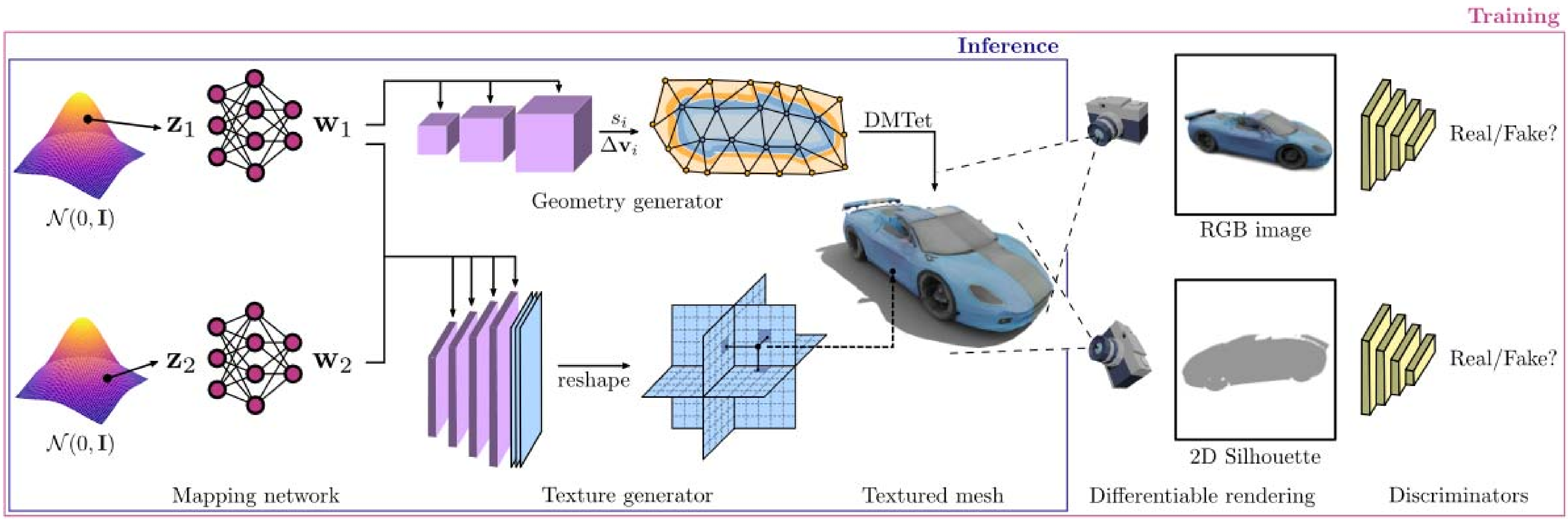
Results

Adversarial Generation of Continuous Implicit Shape Representations, Kleineberg et al. 2020

How to generate shapes/scenes?

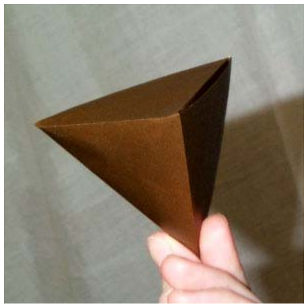
- Encoder-Decoders
  - ➢ Case Study: Multi-view decoder
  - ➢ Case Study: Implicit decoder
  - ➢ Case Study: Patch decoder
  - ➢ Case Study: Mesh Decoder
- Generative Adversarial Networks
  - – Case Study: 3D Volumetric GAN
  - – **Case Study: Get3D**
- Variational Autoencoders
- Autoregressive models
- Diffusion models

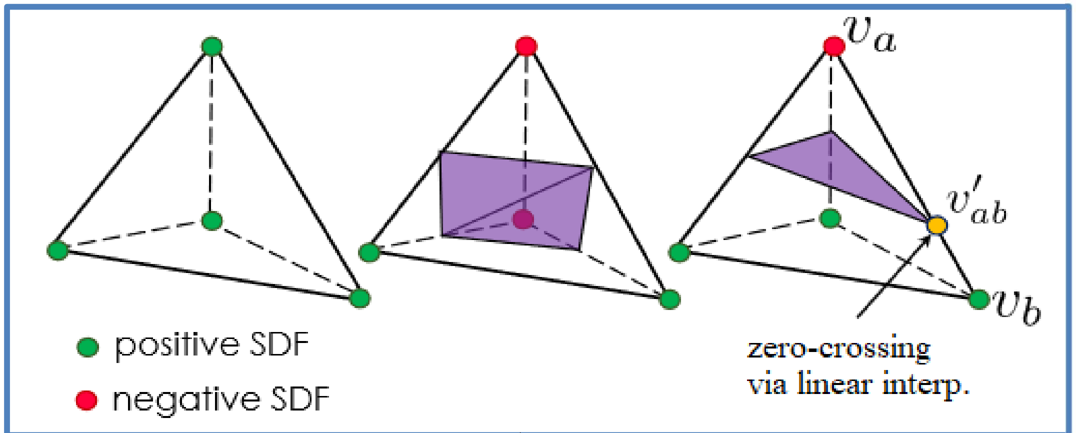
Get3D

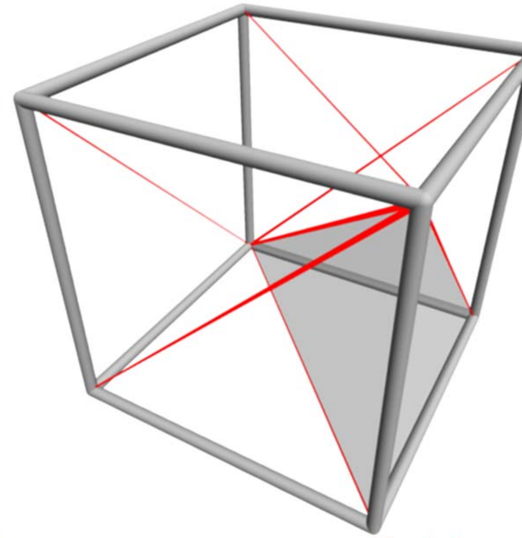


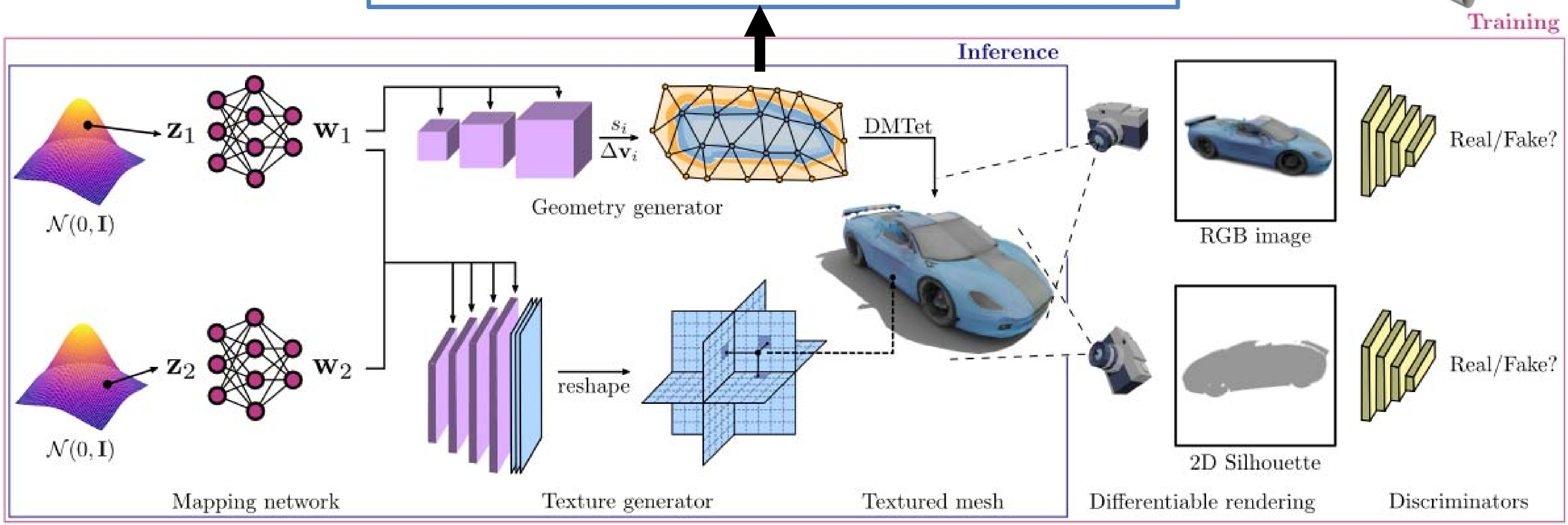
GET3D: A Generative Model of High-Quality 3D Textured Shapes Learned from Images, Gao et al. 2021

Get3D

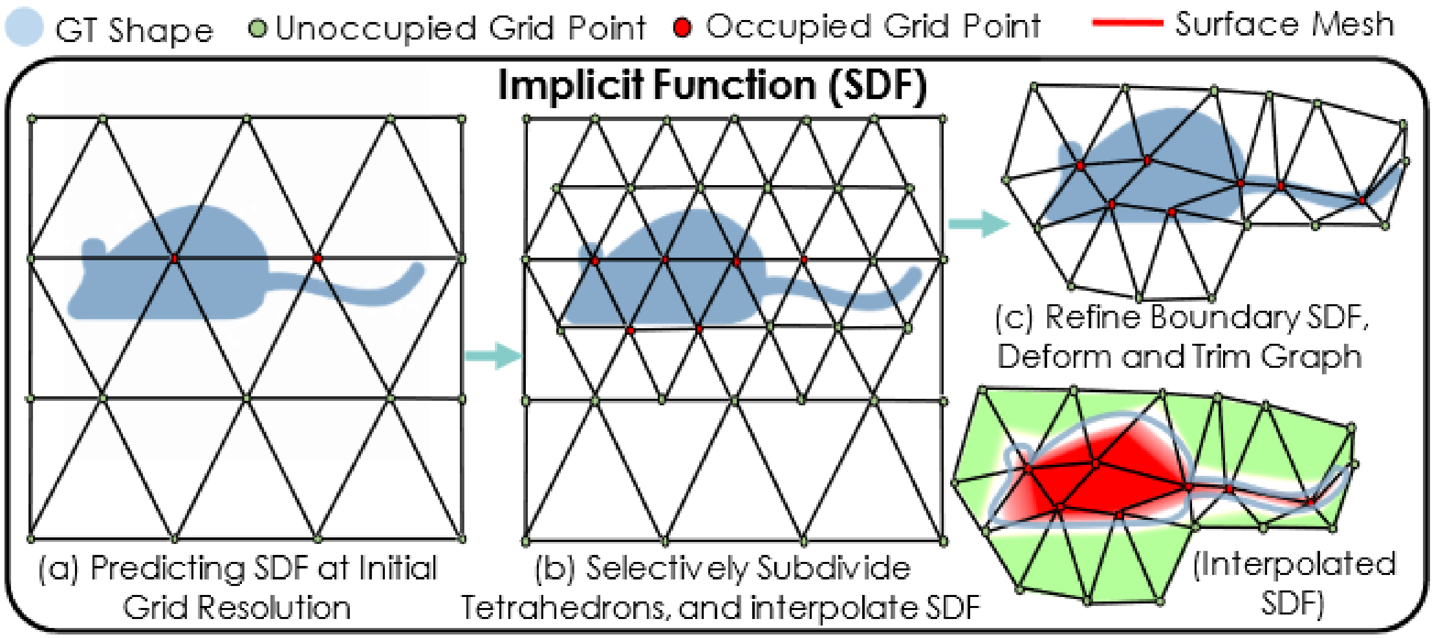




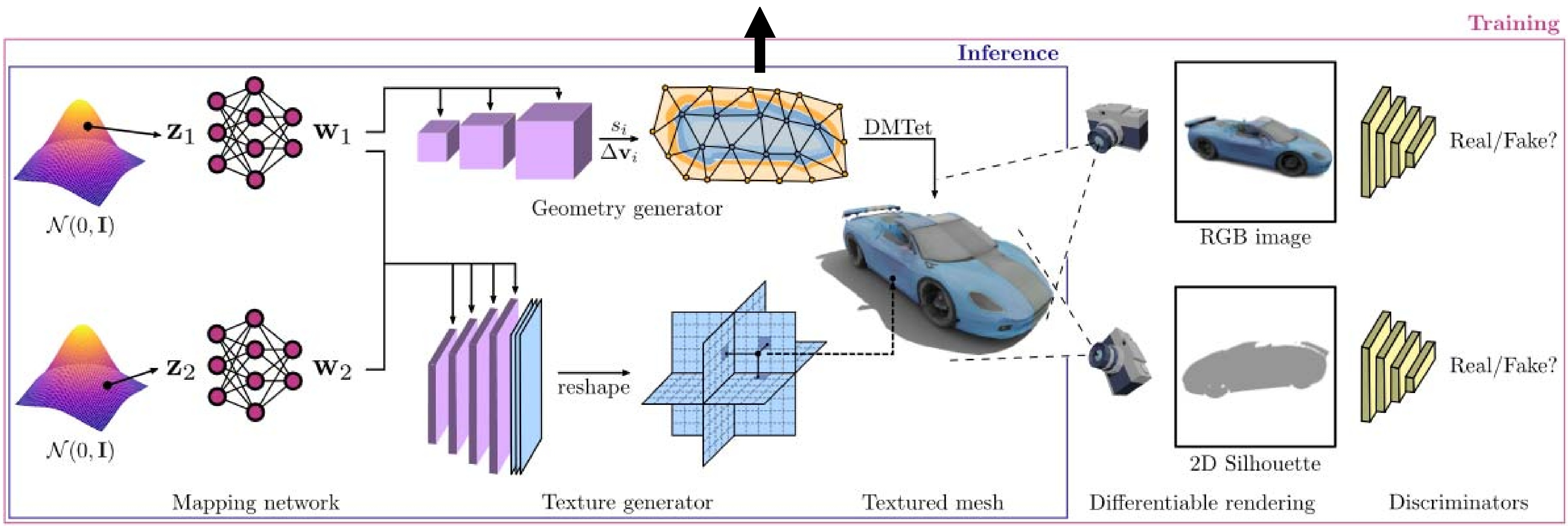




GET3D: A Generative Model of High-Quality 3D Textured Shapes Learned from Images, Gao et al. 2021



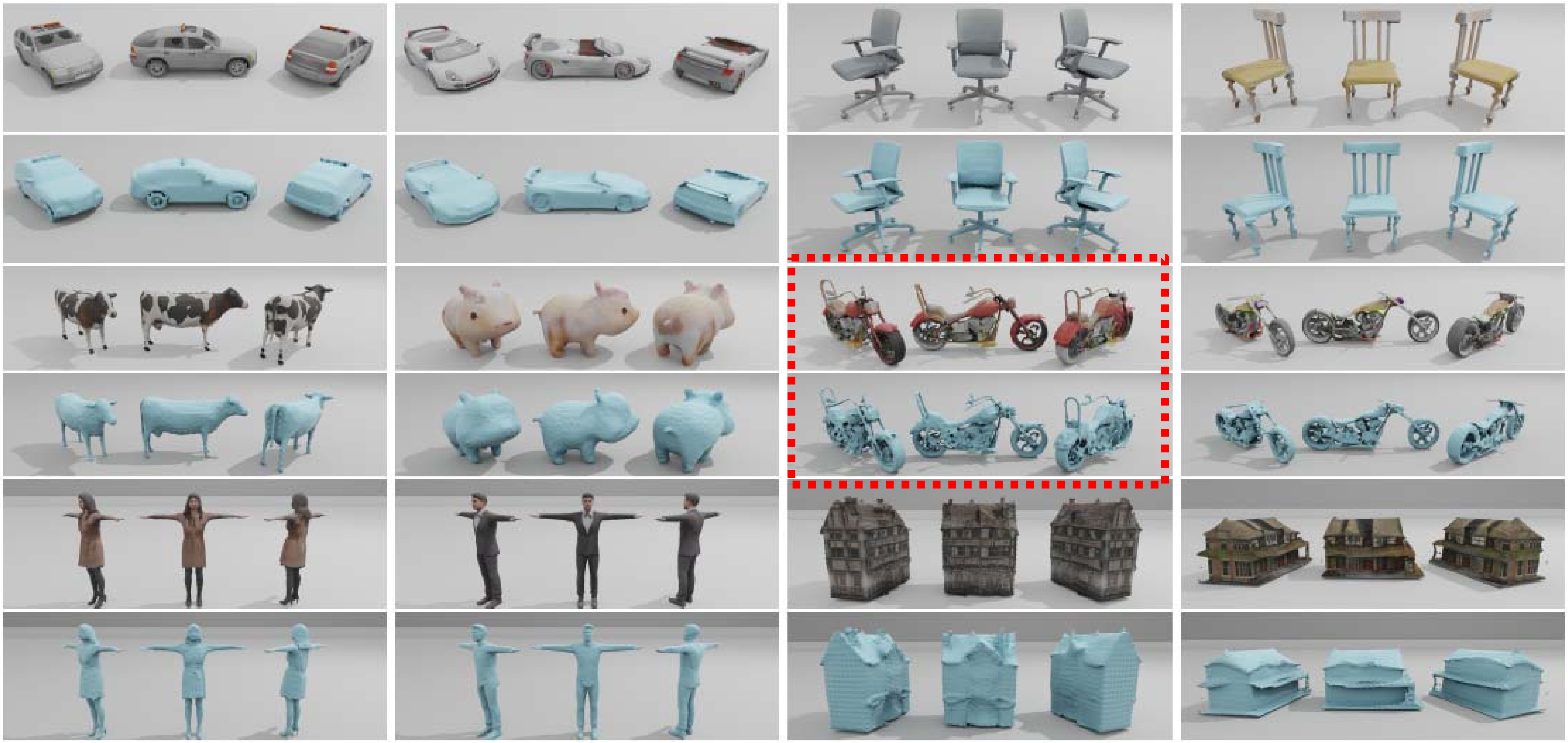
Deep Marching Tetrahedra Shen et al 2021



GET3D: A Generative Model of High-Quality 3D Textured Shapes Learned from Images, Gao et al. 2021

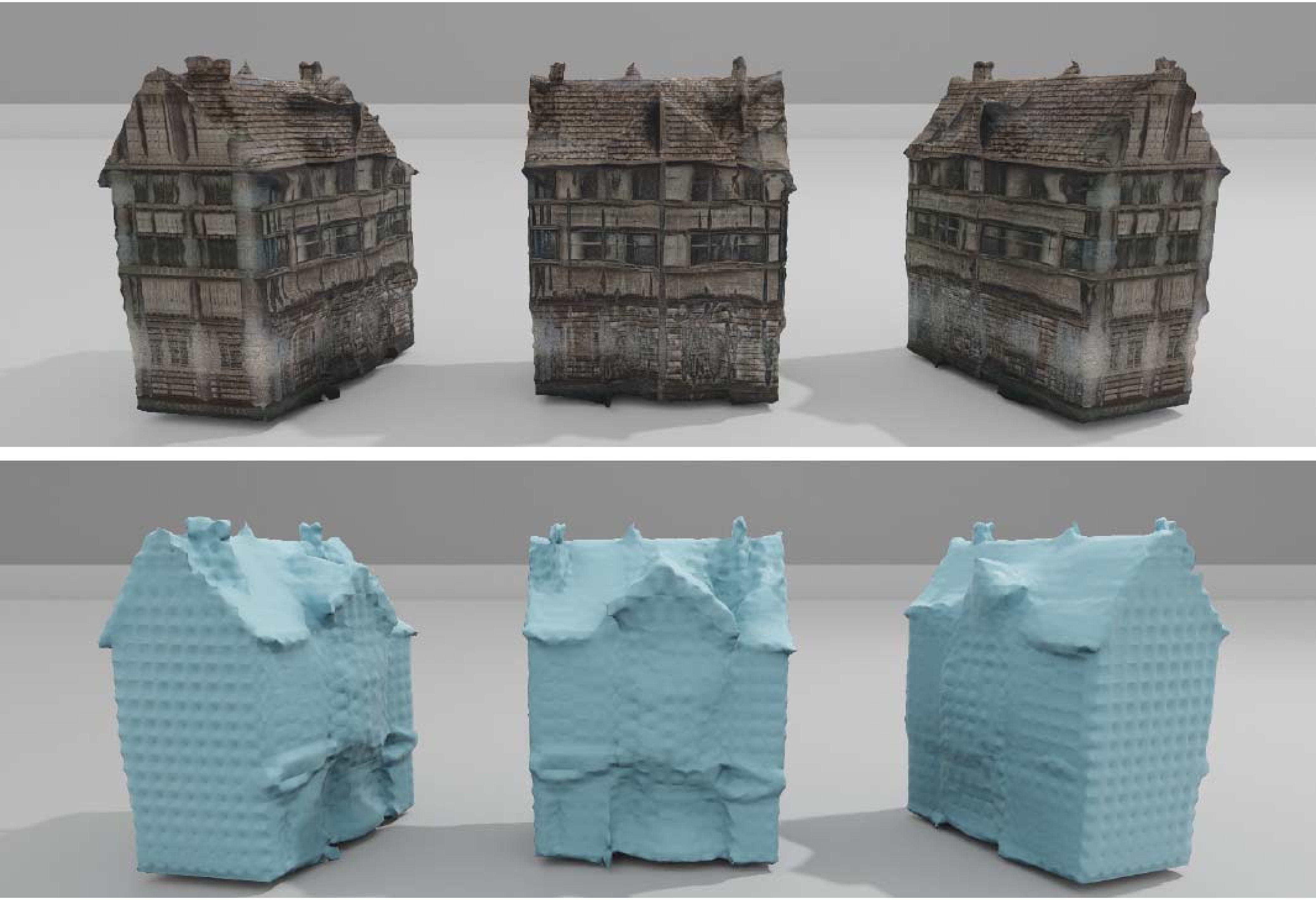
Results

Results

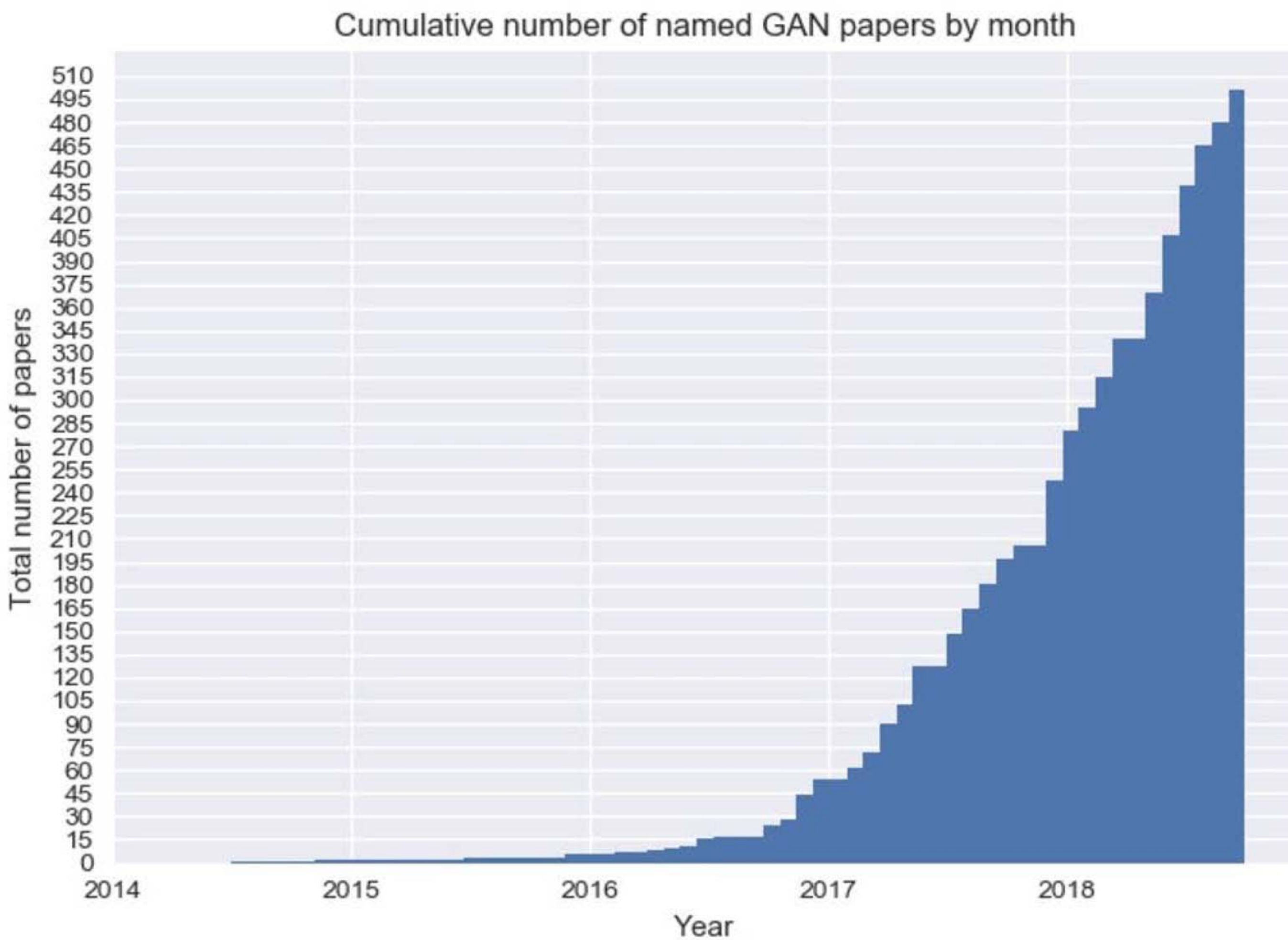


Results

Results



GAN explosion!



https://github.com/hindupuravinash/the-gan-zoo

Good GAN papers to read

Resources

- T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs, NIPS 2016
- S. Chintala, E. Denton, M. Arjovsky, M. Mathieu, How to train a GAN? Tips and tricks to make GANs work, 2016
- I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A. Courville, Improved training of Wasserstein GANs, NIPS 2017
- M. Lucic, K. Kurach, M. Michalski, O. Bousquet, S. Gelly, Are GANs created equal? A large-scale study, NIPS 2018
- P. Isola et al. “Image-to-image translation with conditional adversarial networks.” CVPR 2017.
- J.-Y. Zhu et al. “Unpaired image-to-image translation using cycle-consistent adversarial networks.” ICCV. 2017.
- Zhang, Han, et al. “Self-attention generative adversarial networks”, 2018
- T. Karras, S. Laine, T. Aila, A Style-Based Generator Architecture for Generative Adversarial Networks, CVPR 2019