INVERSE GRAPHICS

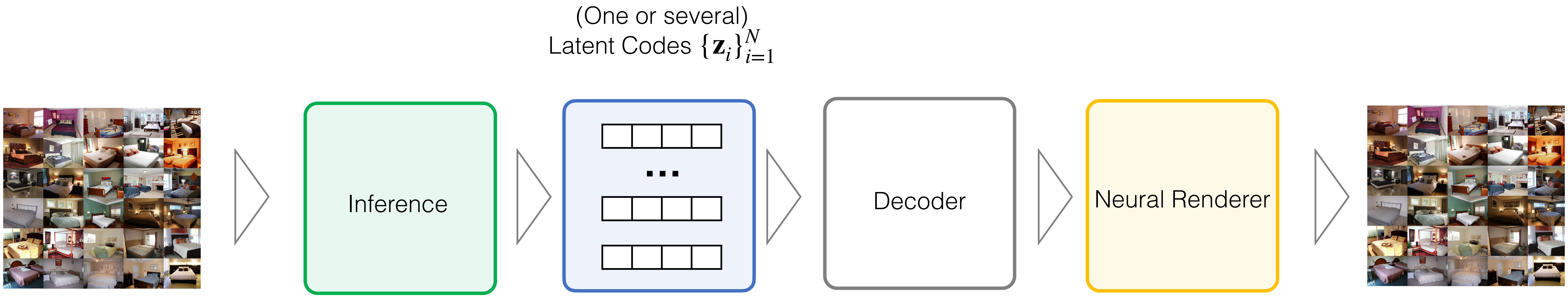
MODULE 3:

Representation Learning, Latent Variable Models, and Auto-encoding

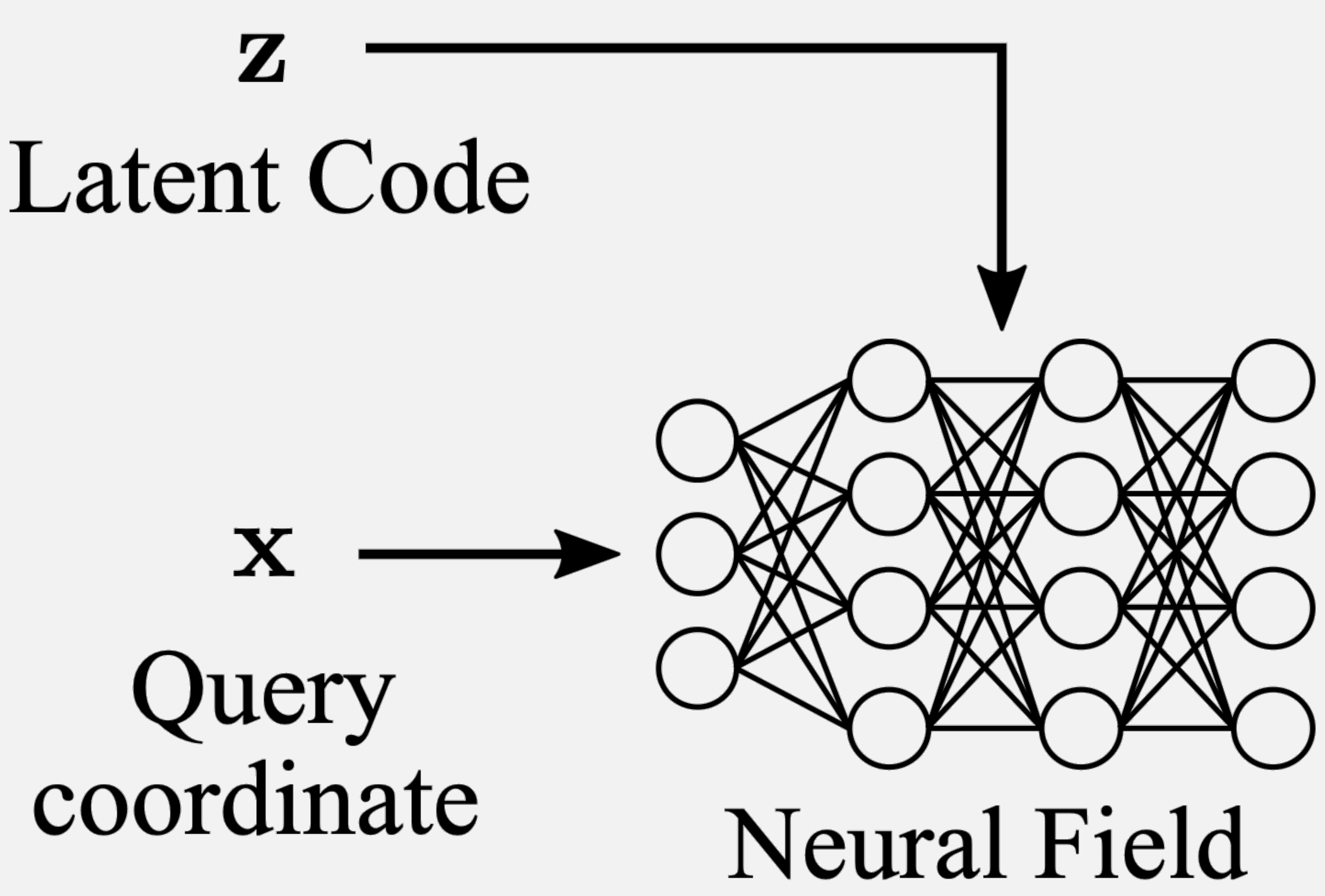
UNCONDITIONAL GENERATIVE MODELING OF 3D SCENES

6.S980

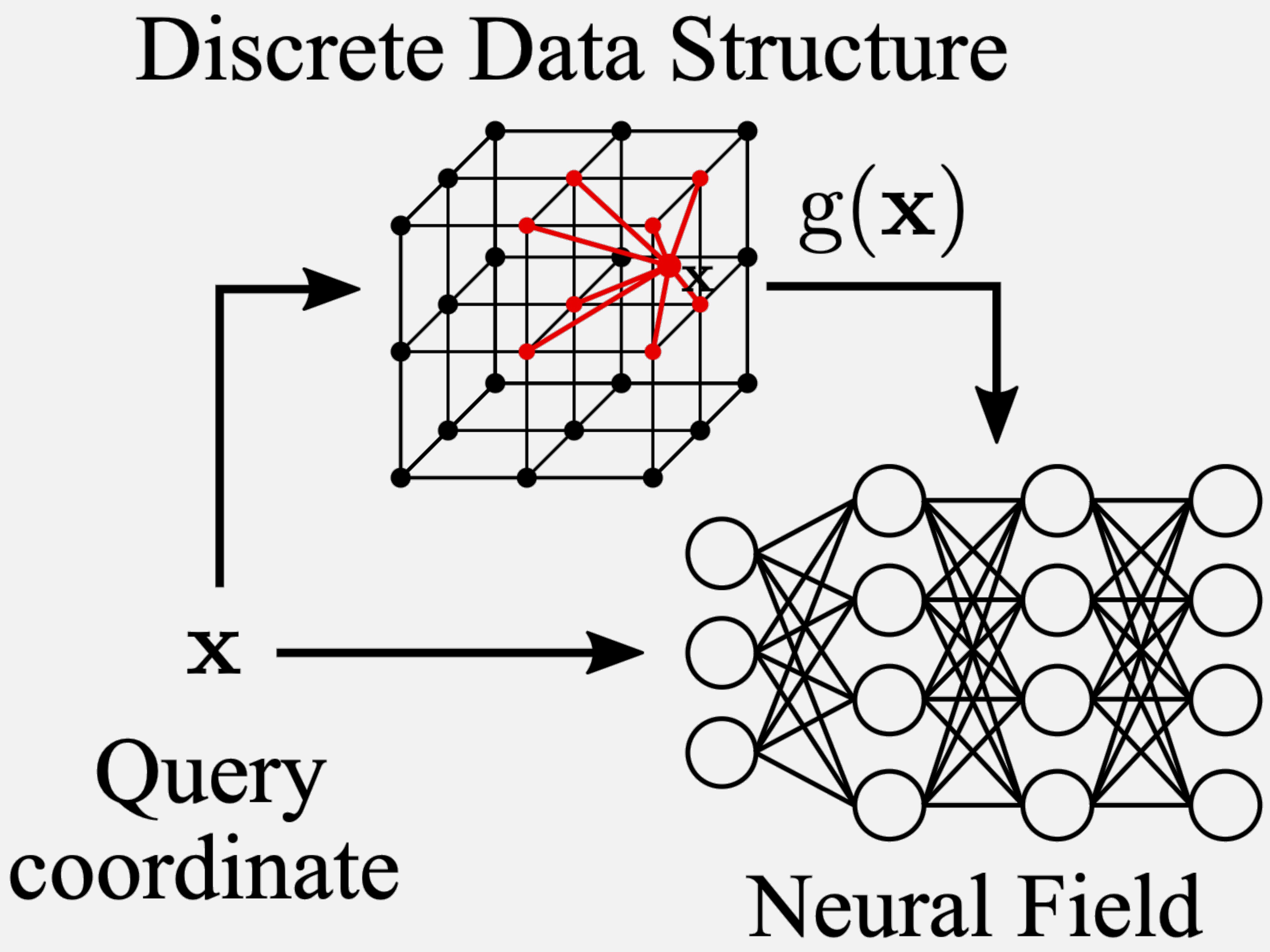
Recap: General Framework for auto-encoding based Scene Representation Learning



Recap: Global Conditioning and Local Conditioning



Global Conditioning



Local Conditioning

Neural Fields in Visual Computing and Beyond, Xie et al., EG STAR 2022

Global Conditioning: One-dimensional latent per scene

embedding z_0

embedding z_1

+

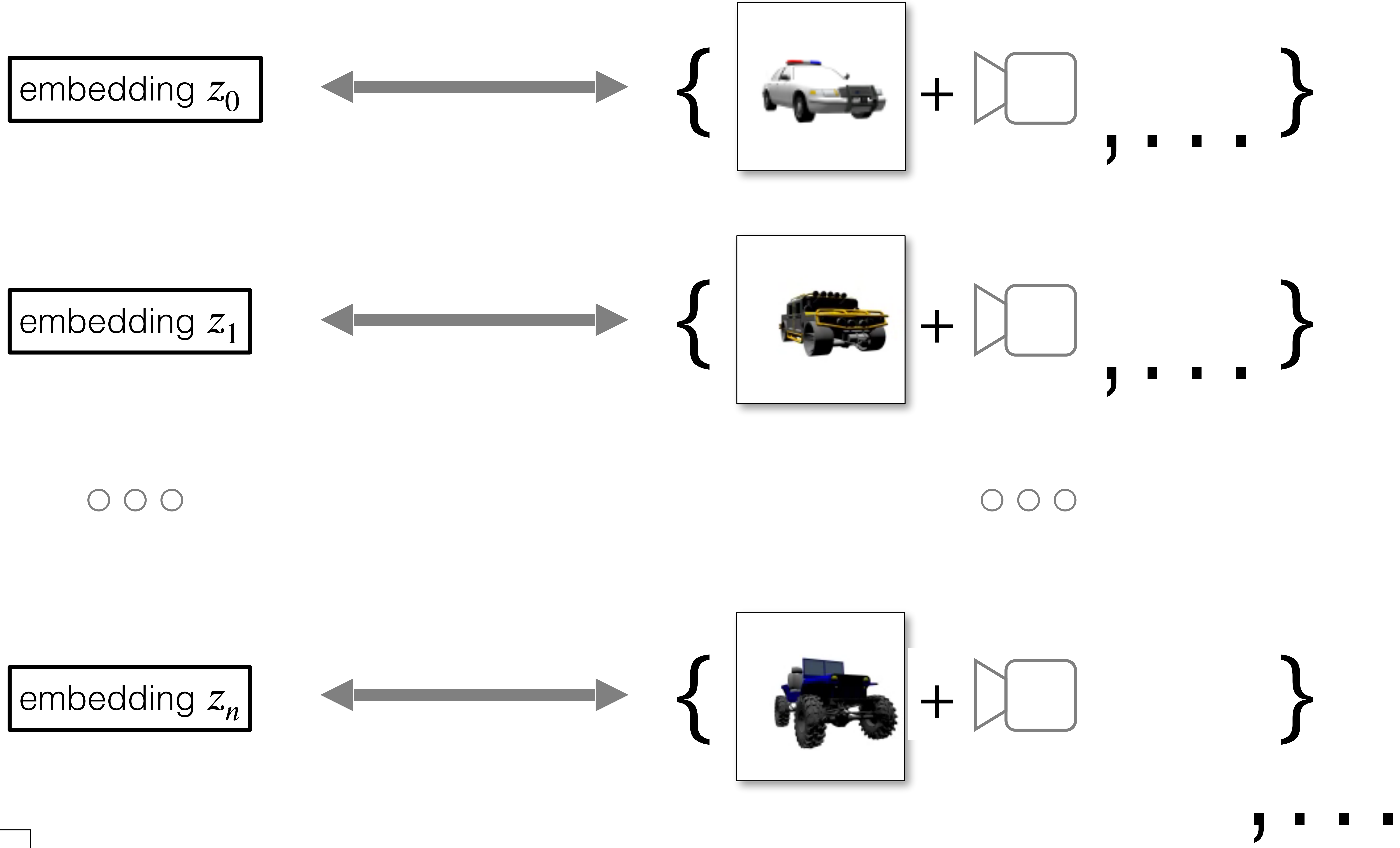
○ ○ ○ ○ ○ ○

embedding z_n

+

$z_{j=0,\ldots,n} \sim \mathcal{N}(0,\sigma^2)$

Global Conditioning: One-dimensional latent per scene

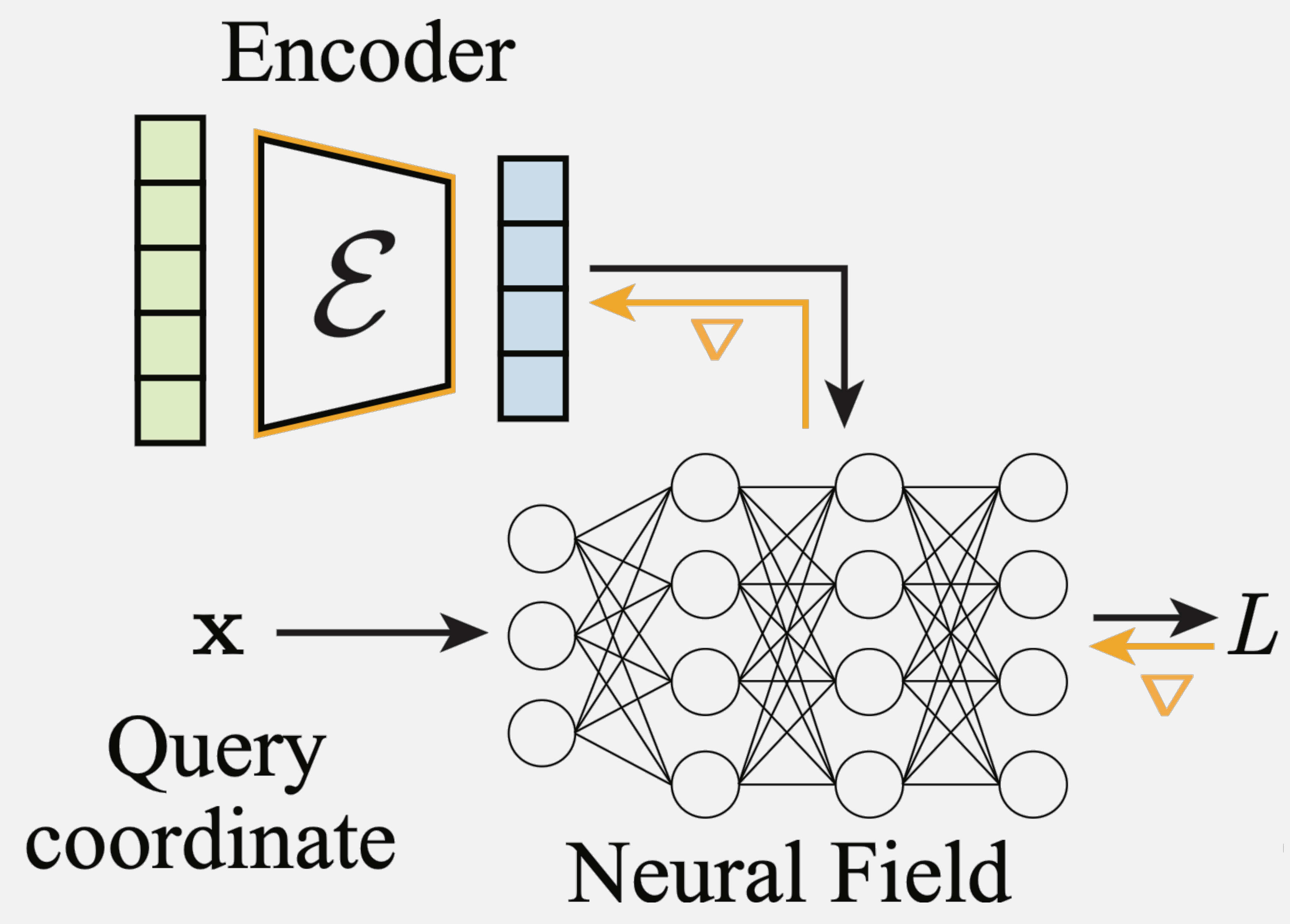


$$z_{j=0,\ldots,n} \sim \mathcal{N}(0, \sigma^2)$$

Local Conditioning: Pixel-Aligned Features.

PiFU, Saito et al., ICCV 2019.
PixelNeRF, Yu et al., CVPR 2021
Grf: Learning a general radiance field…, Trevithick et al.

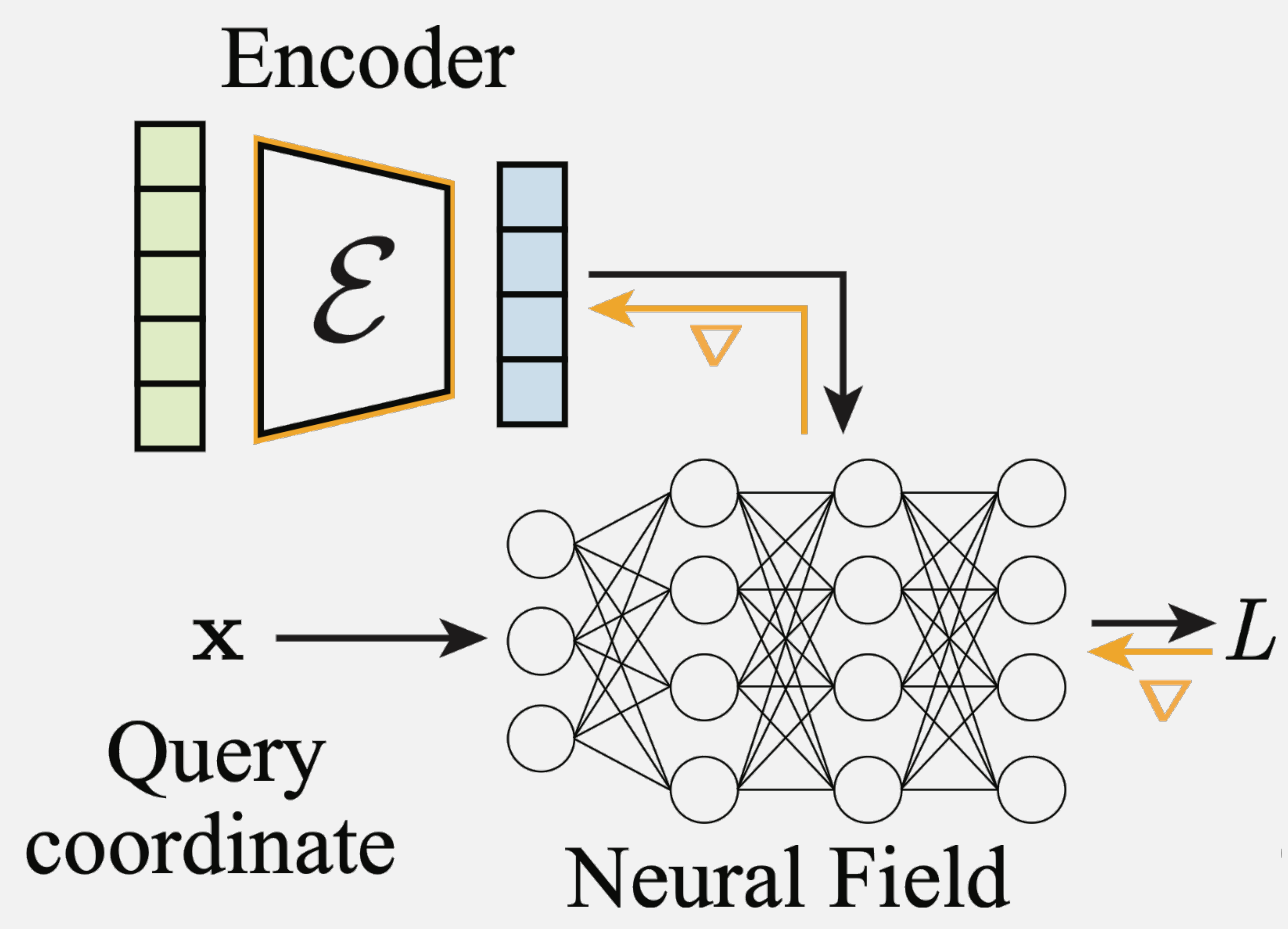
Recap: Encoding & Auto-Decoding



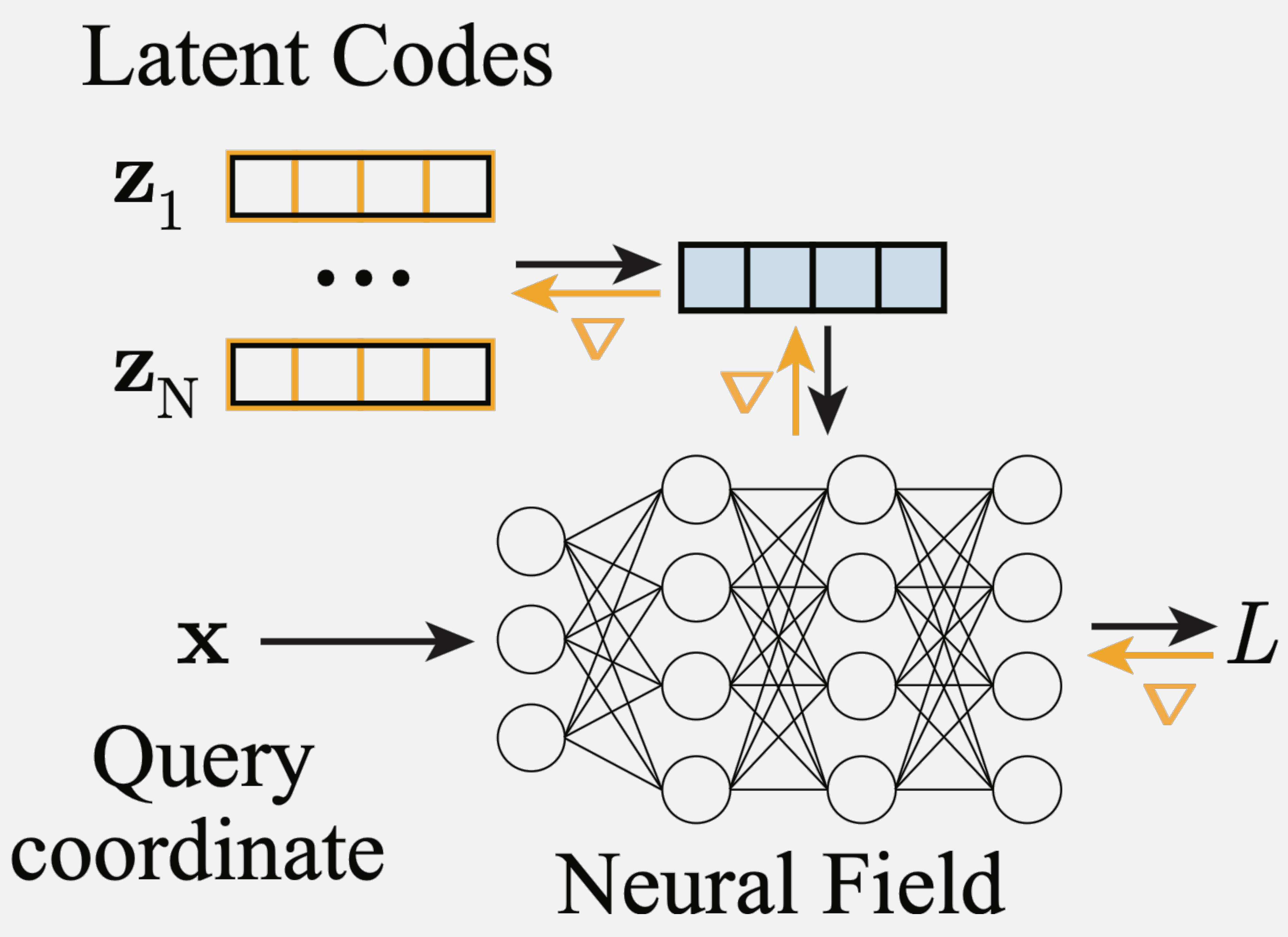
Encoding

Neural Fields in Visual Computing and Beyond, Xie et al., EG STAR 2022

Recap: Encoding & Auto-Decoding



Encoding



Auto-Decoding

Neural Fields in Visual Computing and Beyond, Xie et al., EG STAR 2022

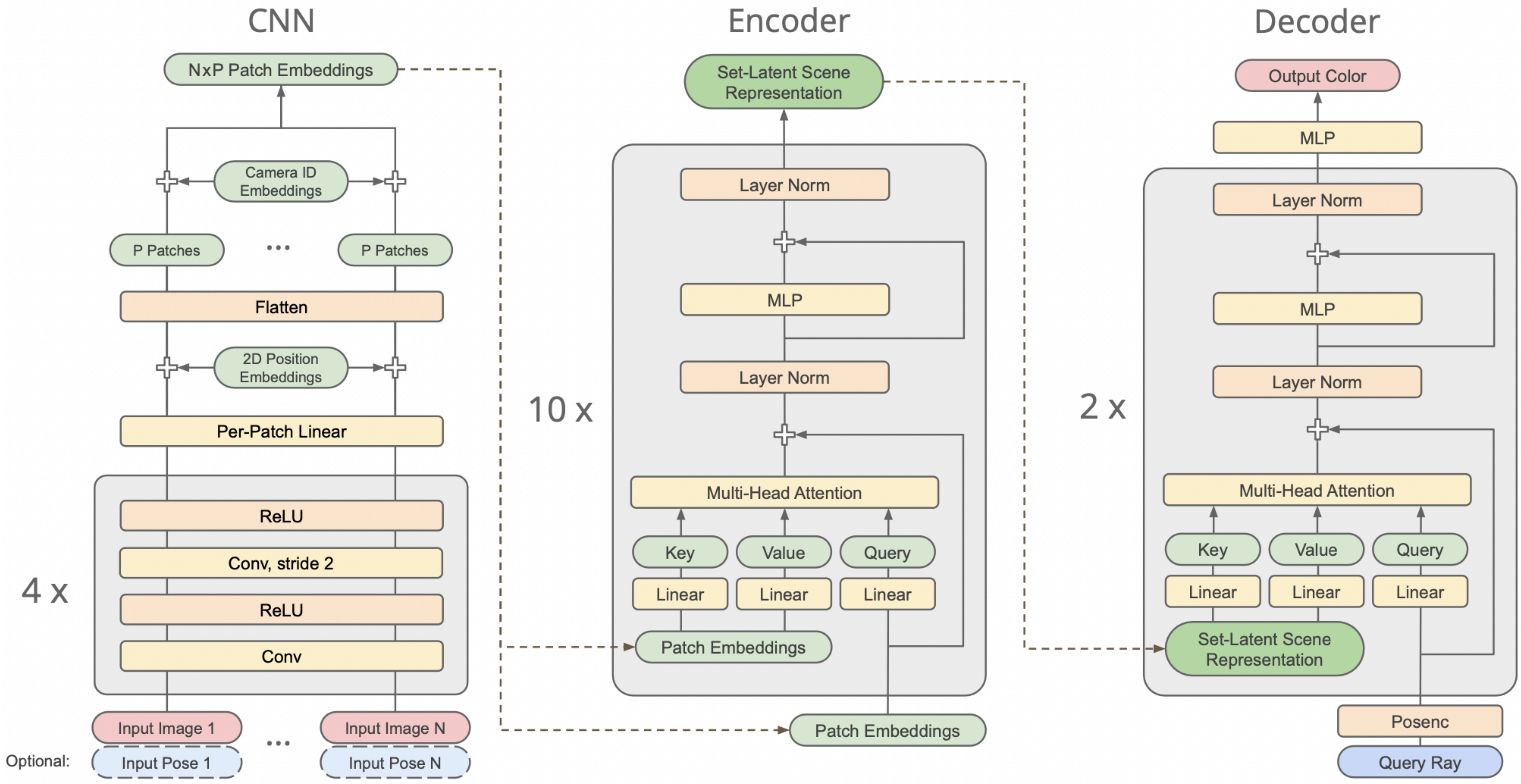
Auto-Decoding: Embedding is not predicted, it's optimized for via gradient descent (test-time optimization!)

$$\arg \min_{\{z_j\}_{j=1}^{M}, \psi, \theta} \sum_j \sum_i \left\| \mathrm{Render}_{\theta}(\mathrm{Srn}_{\phi=HN_{\psi}(z_j)}, \xi_i) - \mathcal{I}_i^j \right\|$$

Encoding: Latents are output of encoder

PiFU, Saito et al., ICCV 2019.
PixelNeRF, Yu et al., CVPR 2021
Grf: Learning a general radiance field…, Trevithick et al.

Encoding: Latents are output of encoder



Many Slides adapted from…

CMU 16-889: Learning for 3D Vision

Prof. Shubham Tulsiani

Today: 3D from scratch (Unconditional Generation)

Slide Credit: Shubham Tulsiani

Today: 3D from scratch (Unconditional Generation)

Learn the space of (possibly textured) shapes

Today: 3D from scratch (Unconditional Generation)

Learn the space of (possibly textured) shapes

Should be able to **sample** from the learned shape space

Slide Credit: Shubham Tulsiani

Today: 3D from scratch (Unconditional Generation)

Why is this an important task?

Slide Credit: Shubham Tulsiani

Today: 3D from scratch (Unconditional Generation)

Why is this an important task?

Can re-use the learned shape space for downstream conditional inference

Today: 3D from scratch (Unconditional Generation)

Why is this an important task?

Can re-use the learned shape space for downstream conditional inference

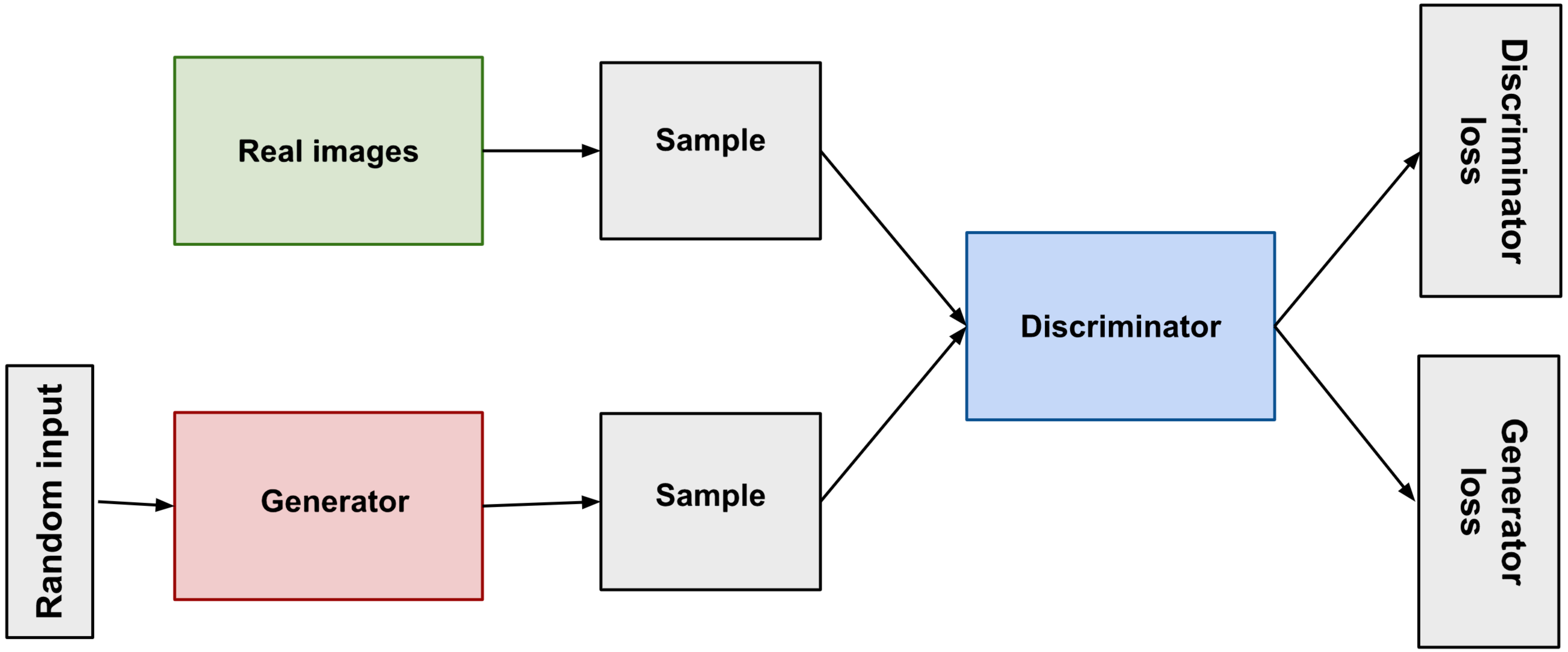
Useful for artists exploring

Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks

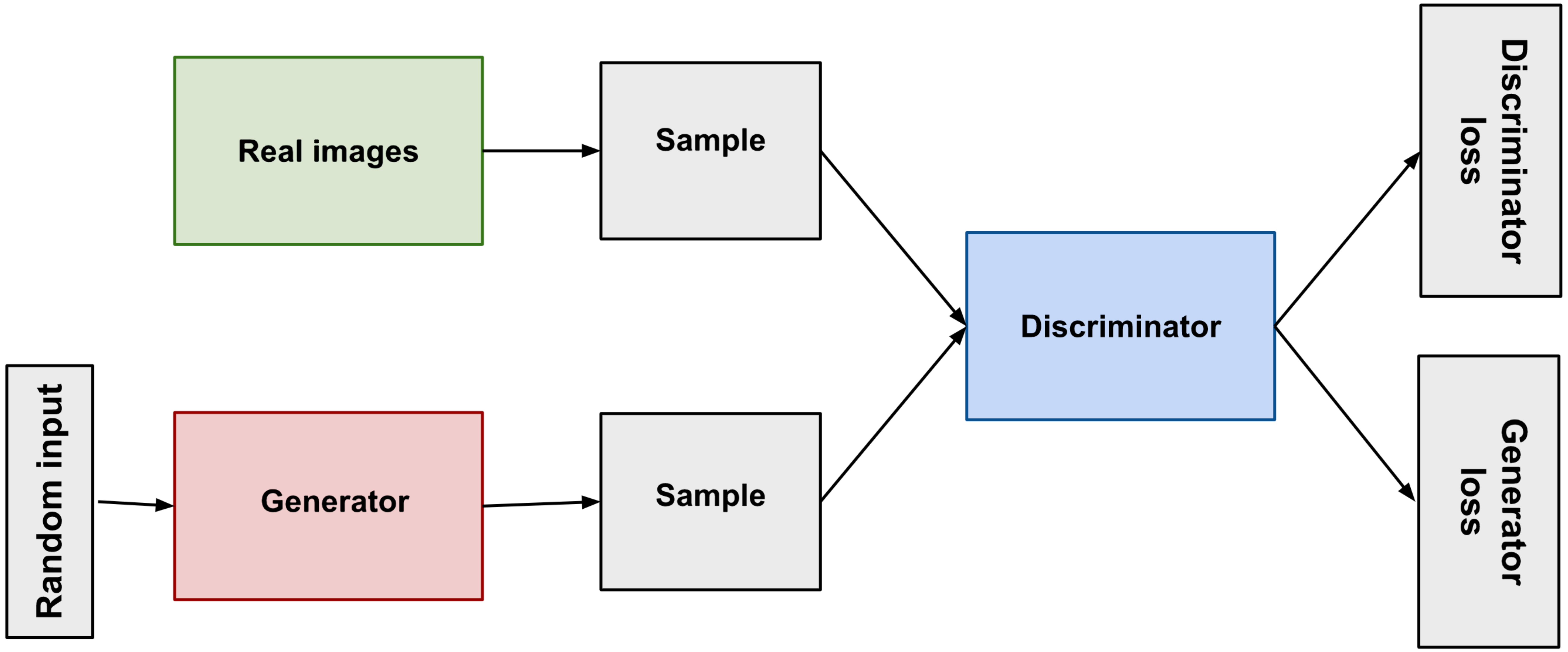
Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks



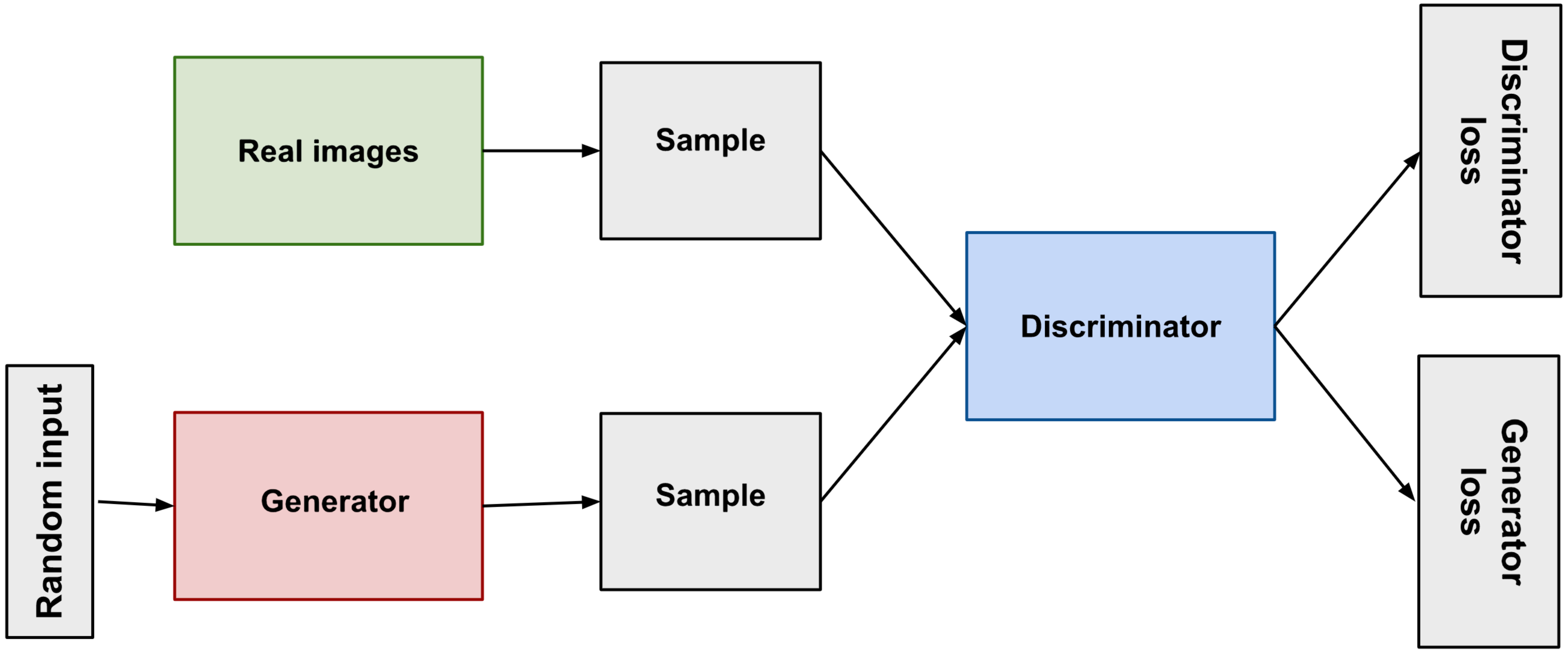
Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks



$$L_{GAN}(G, D) = E_y[log\ D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Background: Generative Adversarial Networks



$$L_{GAN}(G, D) = E_y[log\ D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Learning Objective: Generate output that is indistinguishable from a 'real' example

Slide Credit: Shubham Tulsiani

StyleGAN2 | StyleGAN3 (Ours)

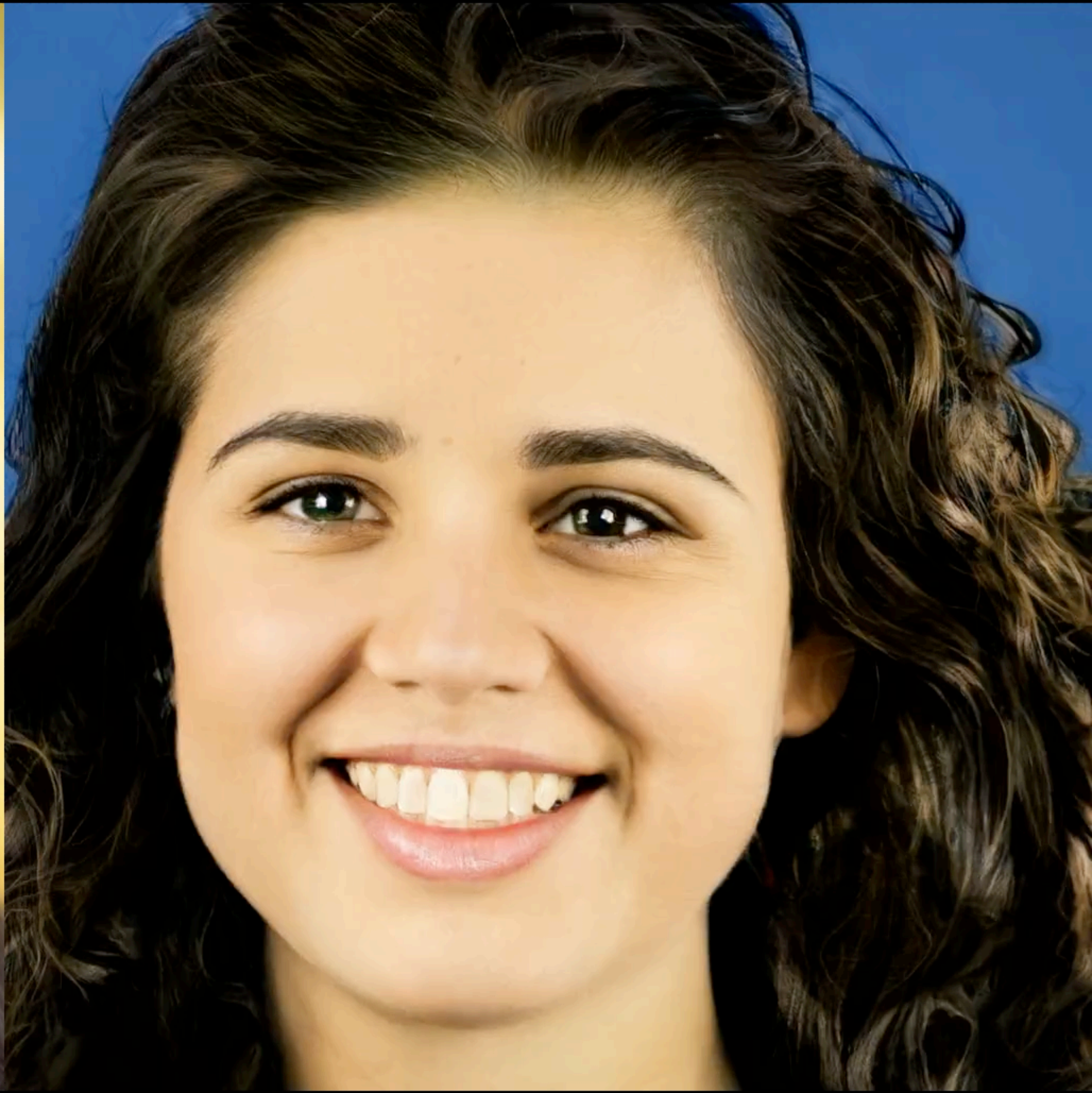
Random latent walk using directions from StyleCLIP, GANSpace, and SeFa.

StyleGAN2 | StyleGAN3 (Ours)

Random latent walk using directions from StyleCLIP, GANSpace, and SeFa.

StyleGAN2

StyleGAN3 (Ours)



Random latent walk using directions from StyleCLIP, GANSpace, and SeFa.

StyleGAN2 | StyleGAN3 (Ours)

Random latent walk using directions from StyleCLIP, GANSpace, and SeFa.


StyleGAN2

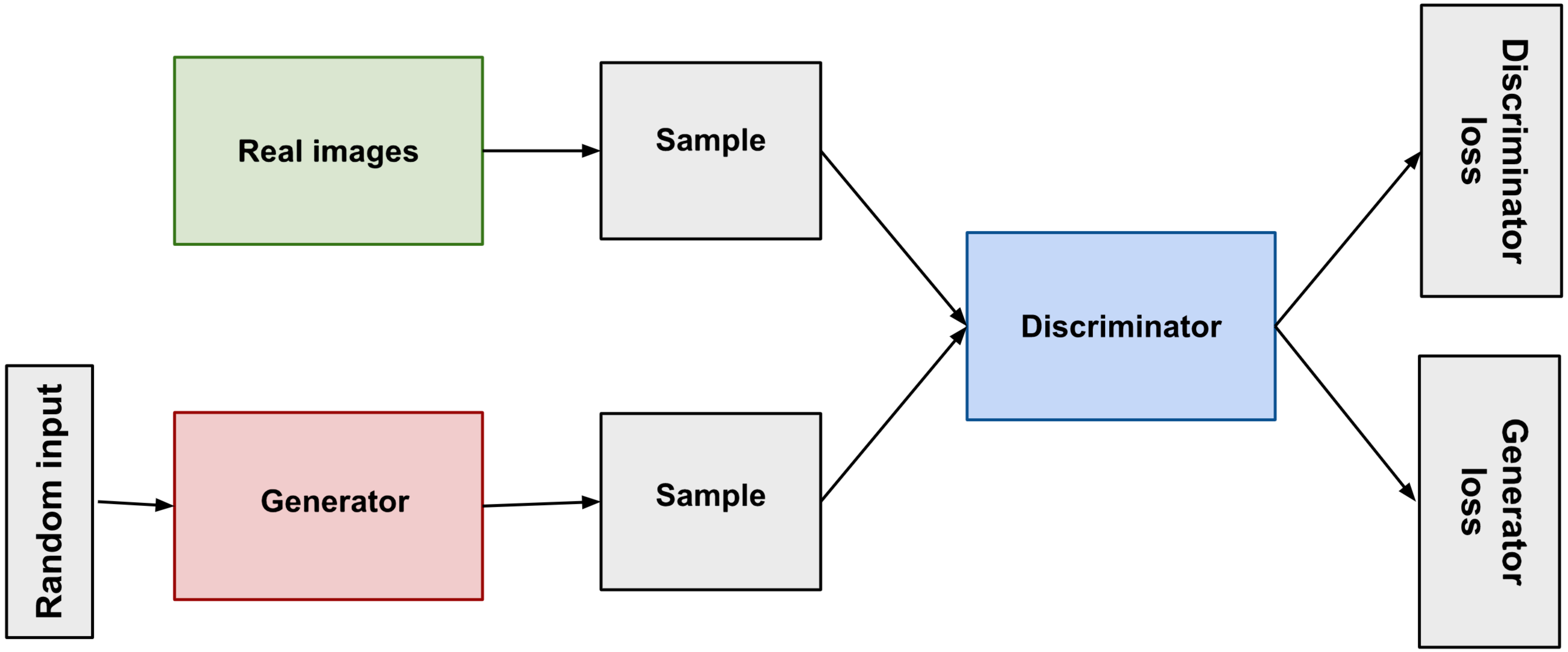
StyleGAN3 (Ours)


StyleGAN2
StyleGAN3 (Ours)


StyleGAN2
StyleGAN3 (Ours)


StyleGAN2
StyleGAN3 (Ours)

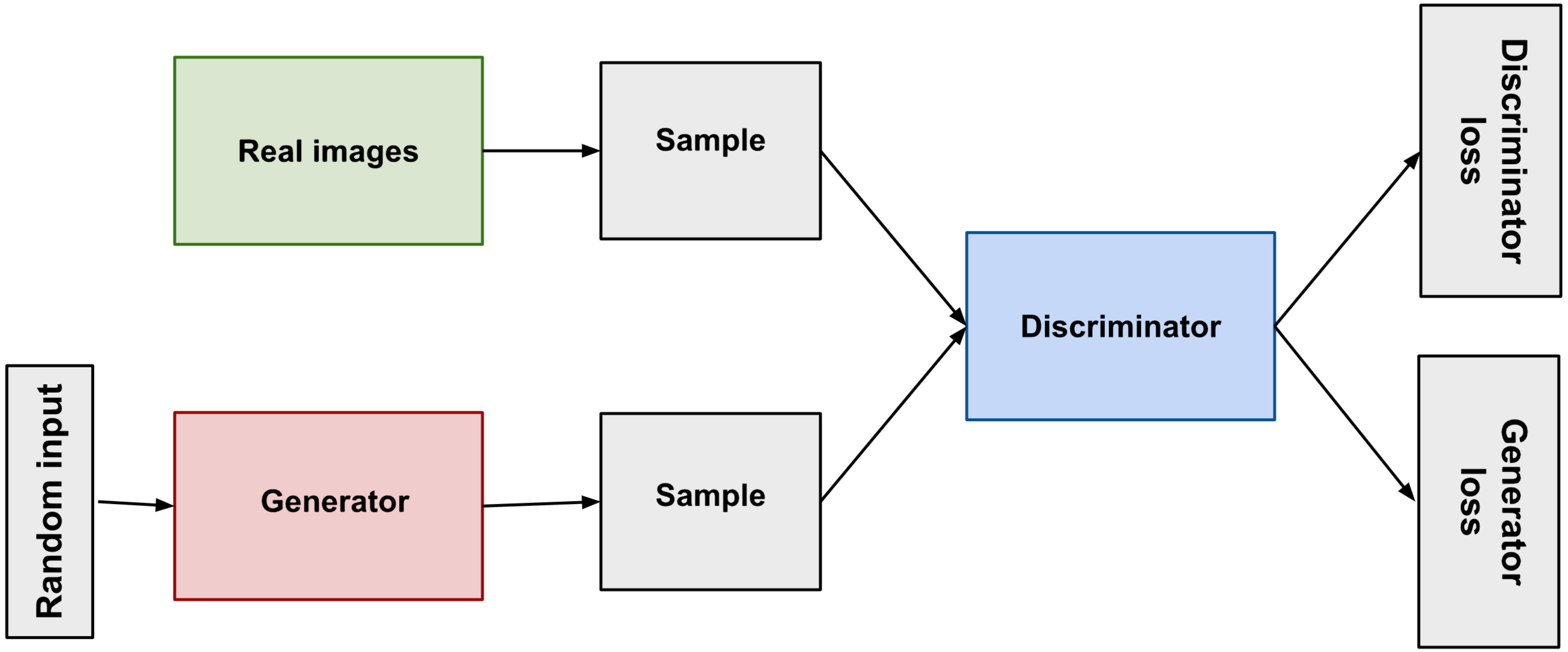
Background: Generative Adversarial Networks



$$L_{GAN}(G, D) = E_y[log\ D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks



$$L_{GAN}(G, D) = E_y[log\, D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Can we similarly learn to generate 3D?

Slide Credit: Shubham Tulsiani

3D GANs for Geometry

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

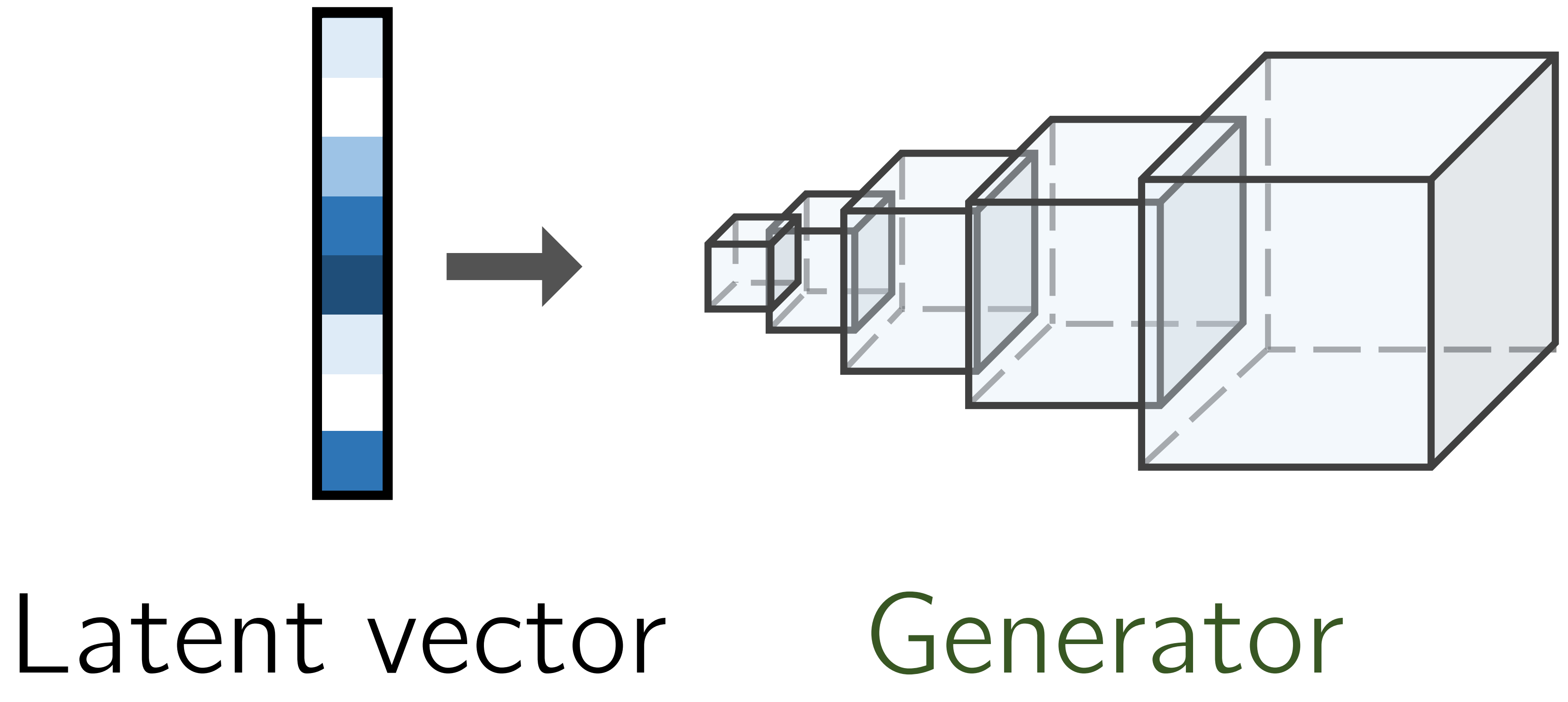
Slide Credit: Shubham Tulsiani

3D GANs for Geometry

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

Slide Credit: Shubham Tulsiani

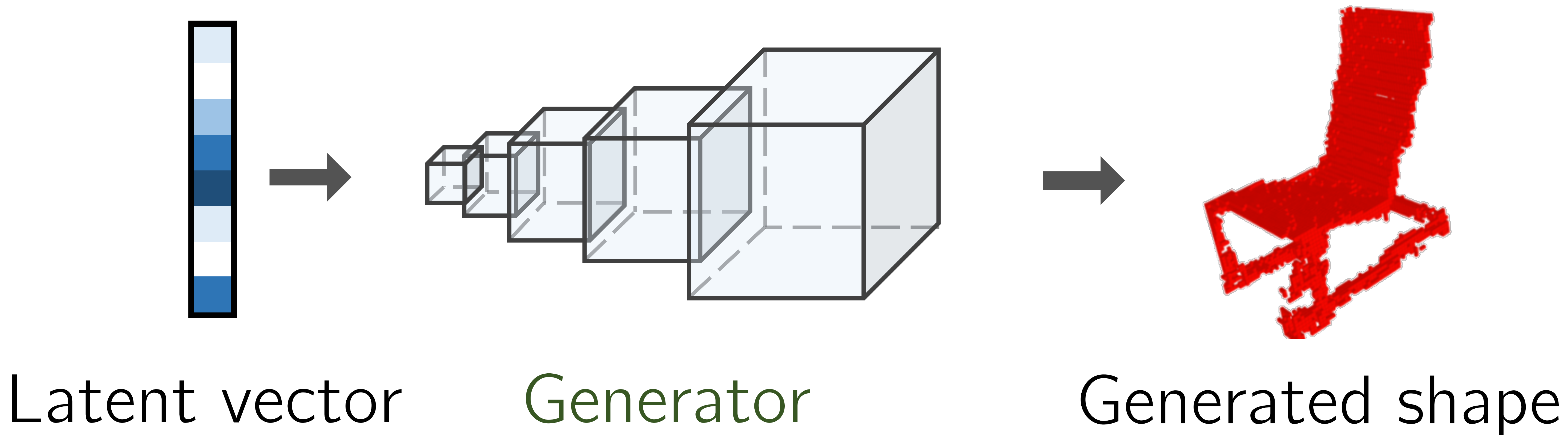
3D GANs for Geometry



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

Slide Credit: Shubham Tulsiani

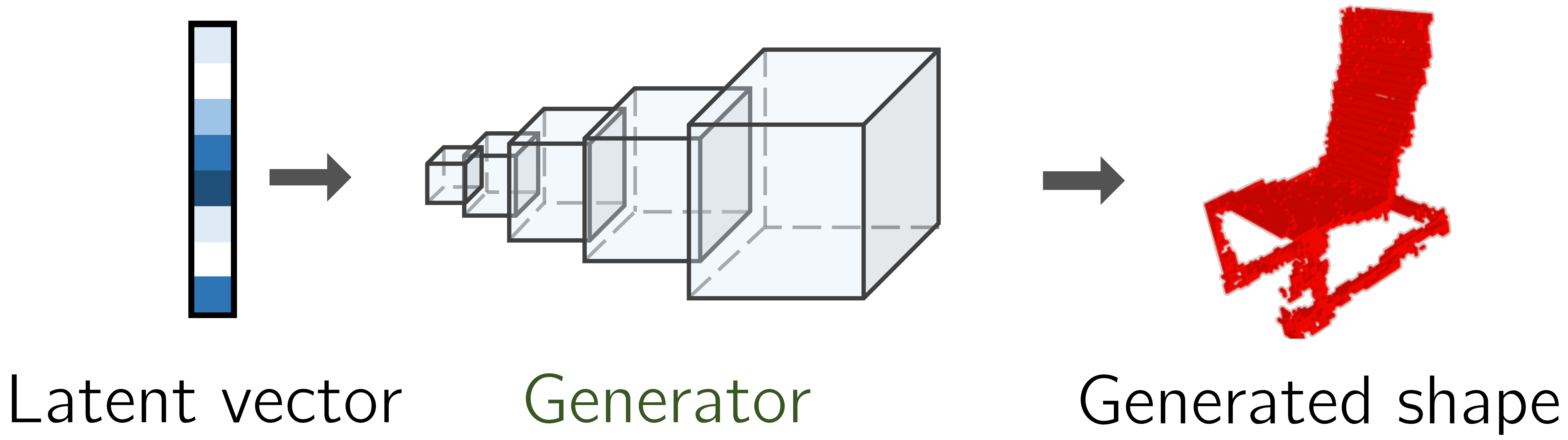
3D GANs for Geometry



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

Slide Credit: Shubham Tulsiani

3D GANs for Geometry



(sampled from ShapeNet)

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

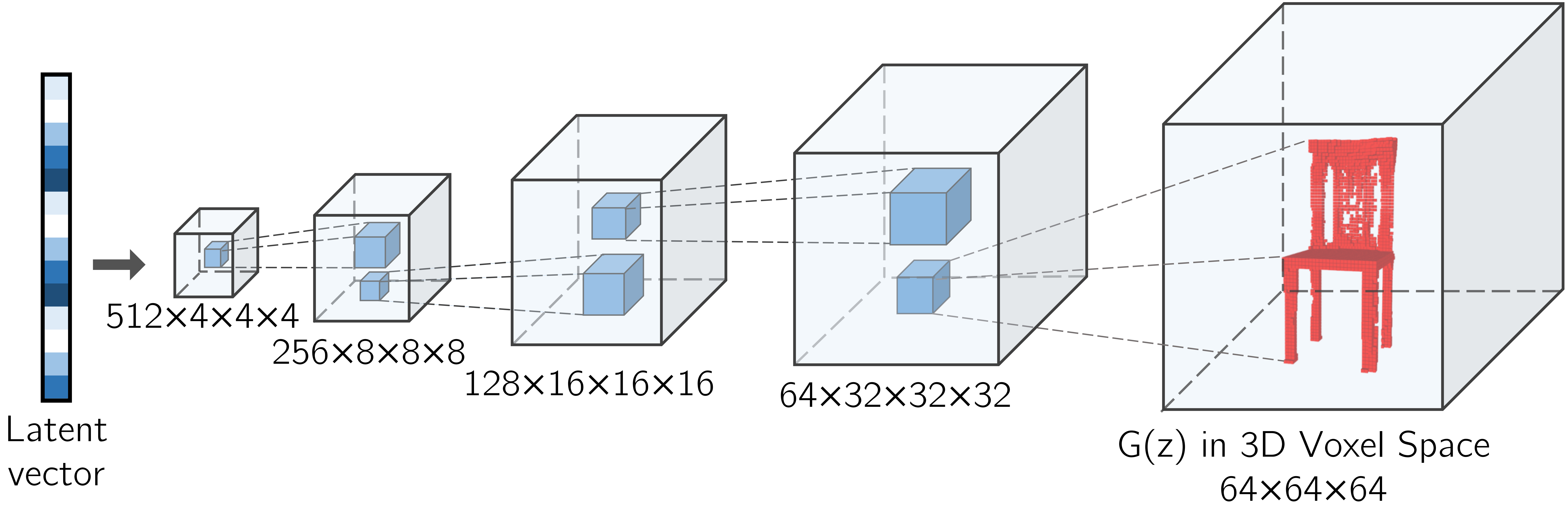
Slide Credit: Shubham Tulsiani

3D GANs for Geometry

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

Slide Credit: Shubham Tulsiani

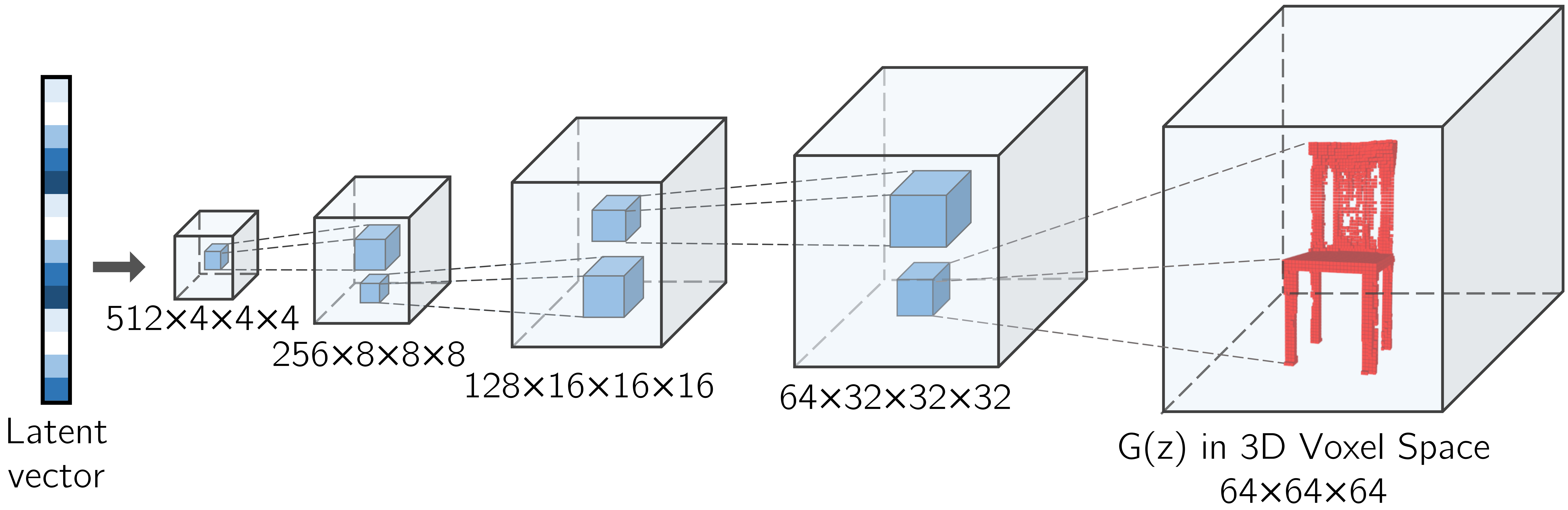
3D GAN: Generator



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

Slide Credit: Shubham Tulsiani

3D GAN: Generator

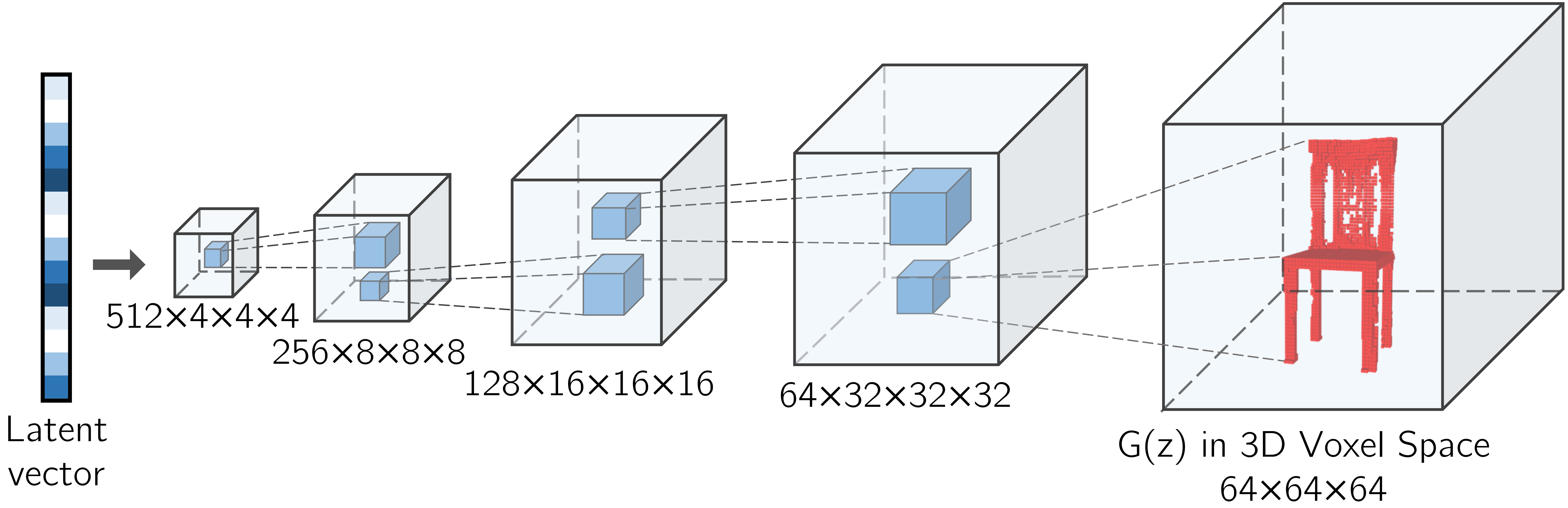


Incrementally increase resolution via convolutions and upsampling layers

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

Slide Credit: Shubham Tulsiani

3D GAN: Generator



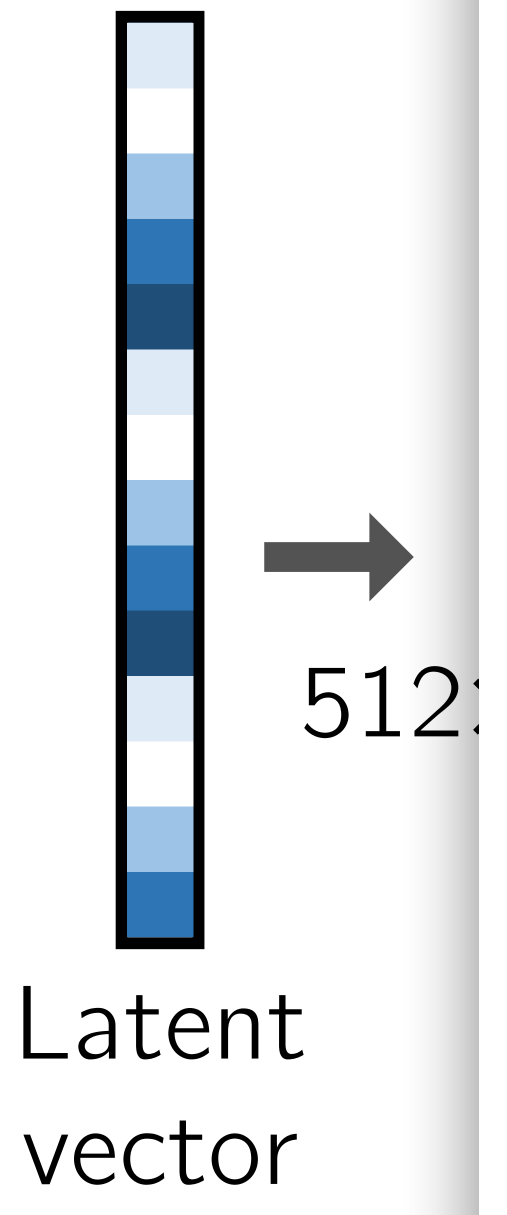
Incrementally increase resolution via convolutions and upsampling layers

Will implement that in Assignment 3!

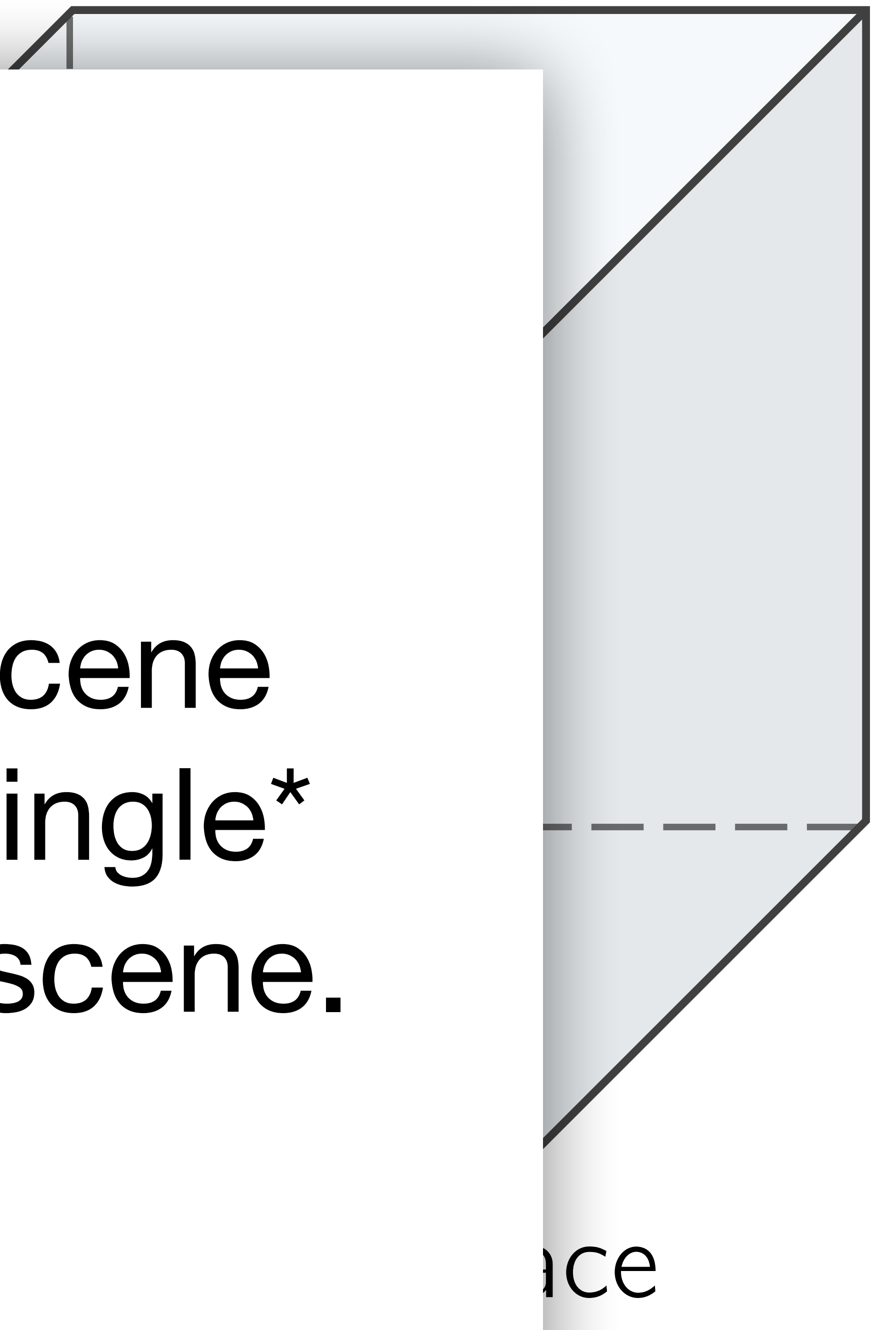
Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. *Wu et. al.*

Slide Credit: Shubham Tulsiani

3D GAN: Generator



Note: GANs are a form of *globally conditioned* scene representation! We condition the generator on a *single* latent code that encodes all information about the scene.



Incrementally increase resolution via convolutions and upsampling layers

Will implement that in Assignment 3!

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

3D GAN: Sampled Shapes

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

Slide Credit: Shubham Tulsiani

3D GAN: Sampled Shapes

Chairs

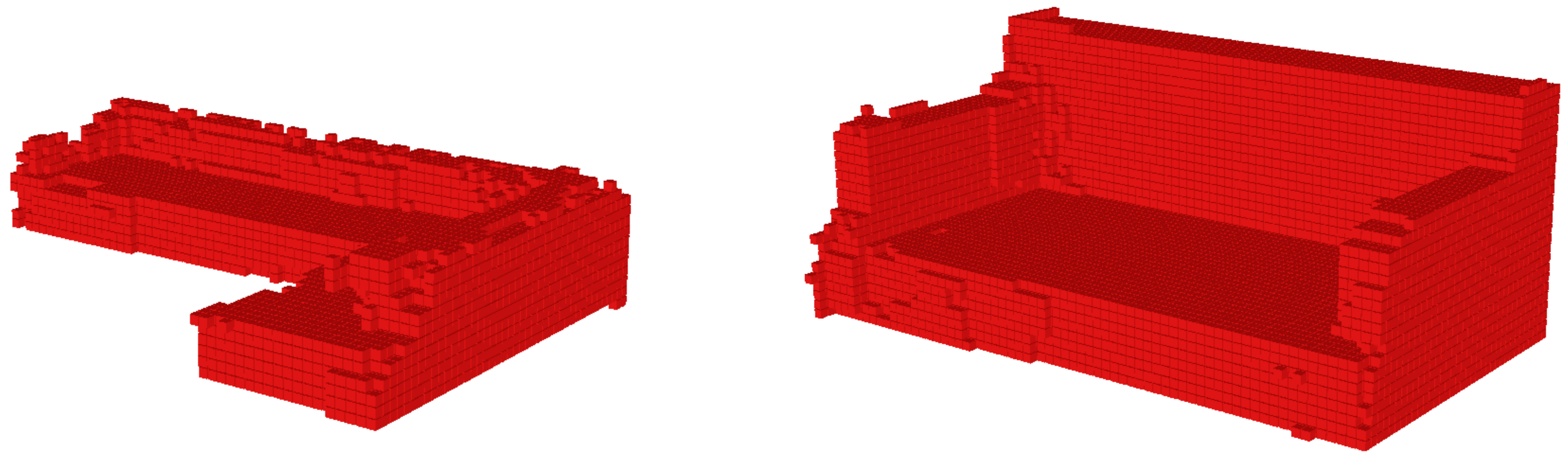
Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. *Wu et. al.*

Slide Credit: Shubham Tulsiani

3D GAN: Sampled Shapes

Chairs

Sofas



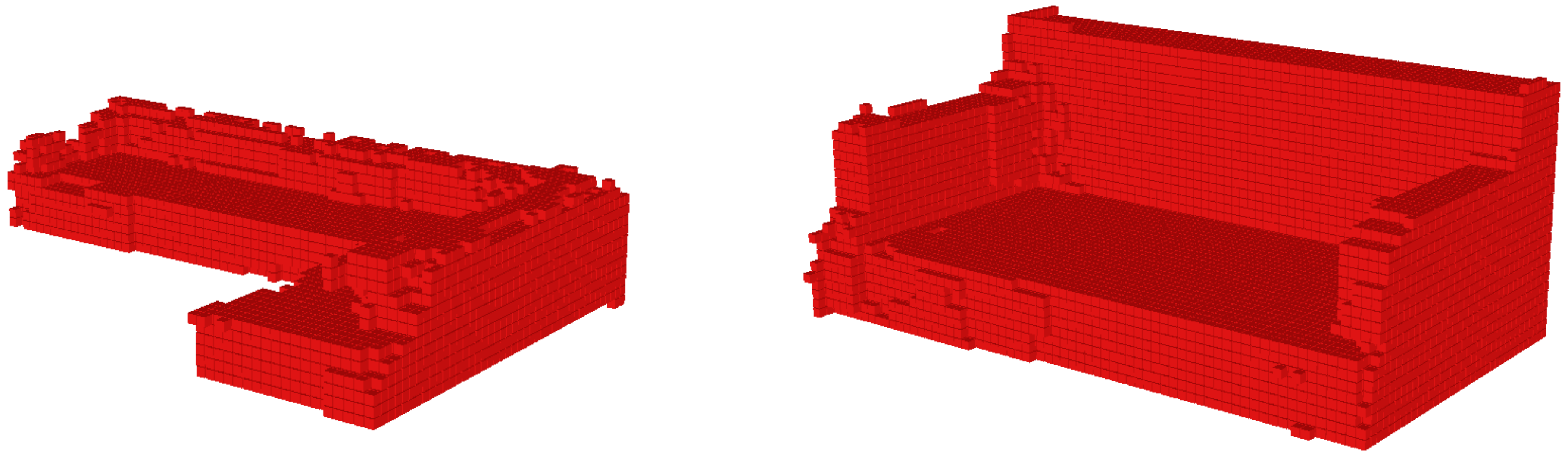
Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. *Wu et. al.*

Slide Credit: Shubham Tulsiani

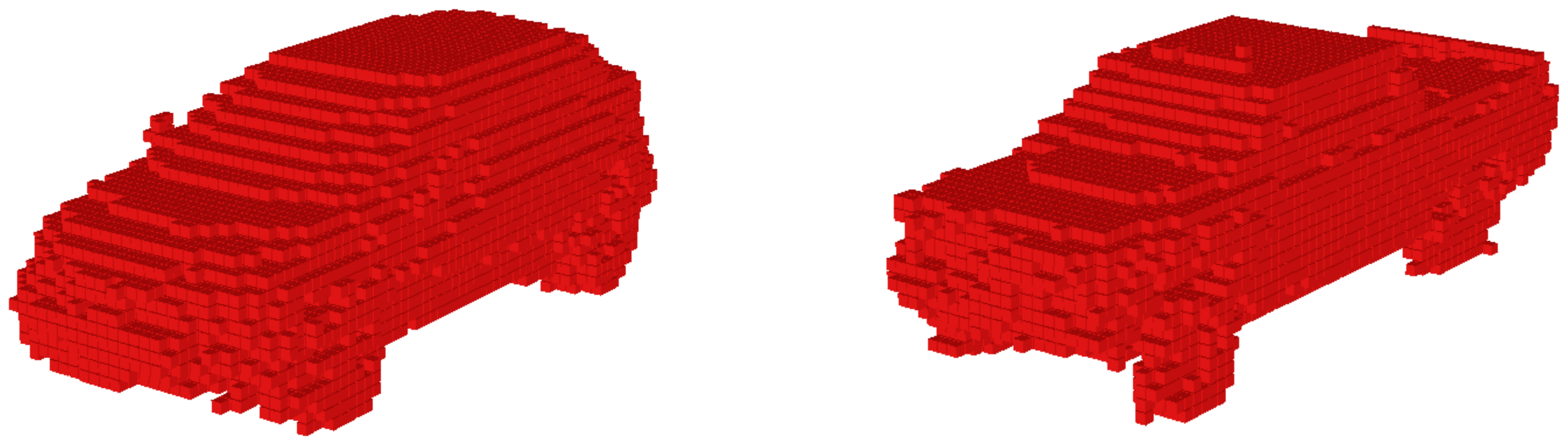
3D GAN: Sampled Shapes

Chairs

Sofas



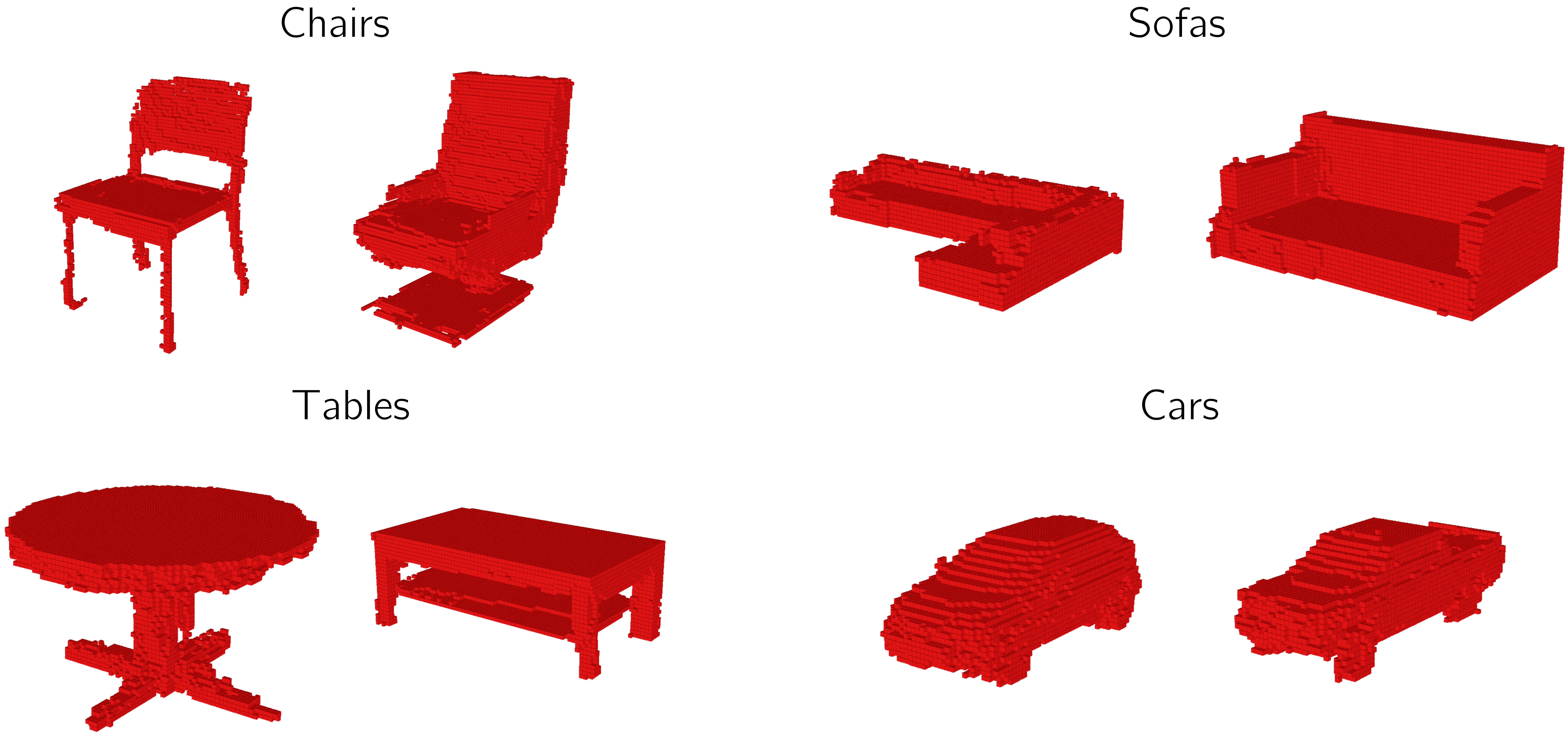
Cars



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

Slide Credit: Shubham Tulsiani

3D GAN: Sampled Shapes



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

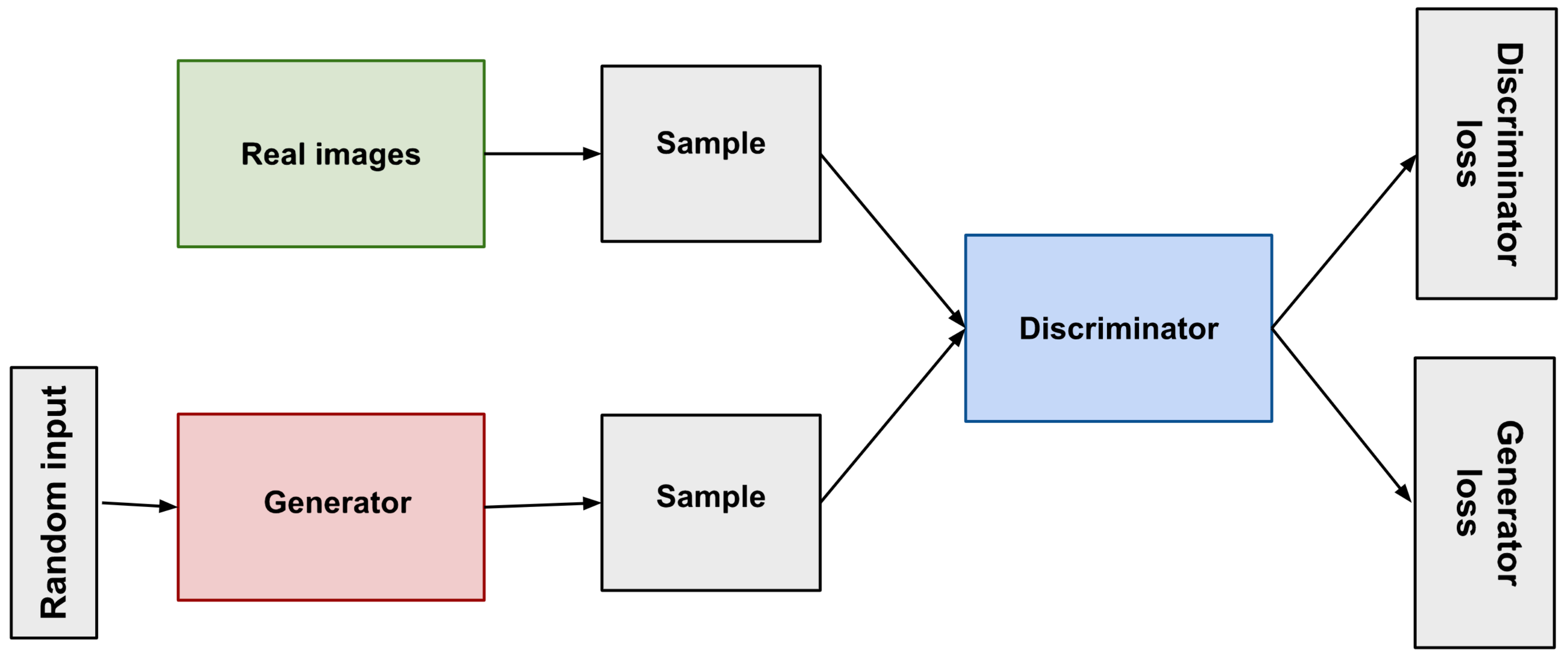
Slide Credit: Shubham Tulsiani

3D GAN: Sampled Shapes

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

Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks

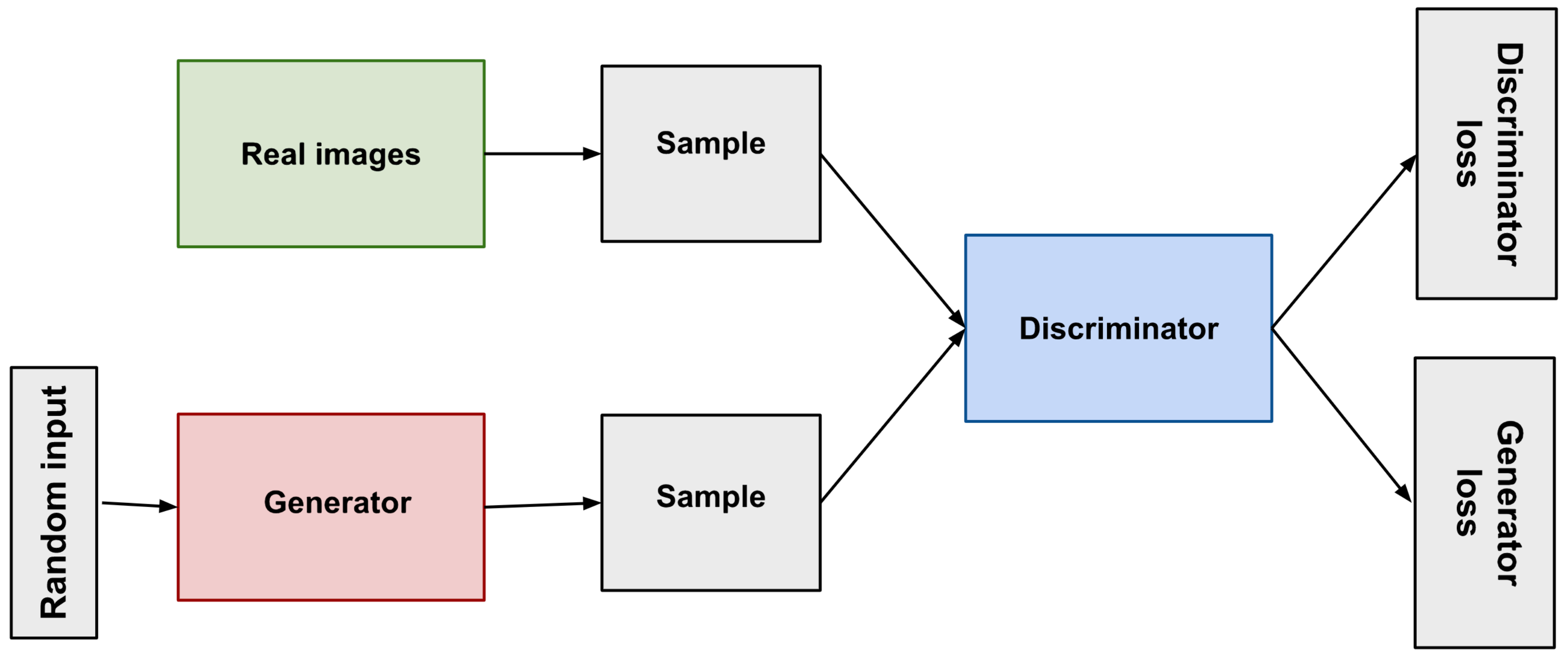


$$L_{GAN}(G, D) = E_y[log\ D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Image courtesy: Overcoming Mode Collapse and the Curse of Dimensionality. *Ke Li*

Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks

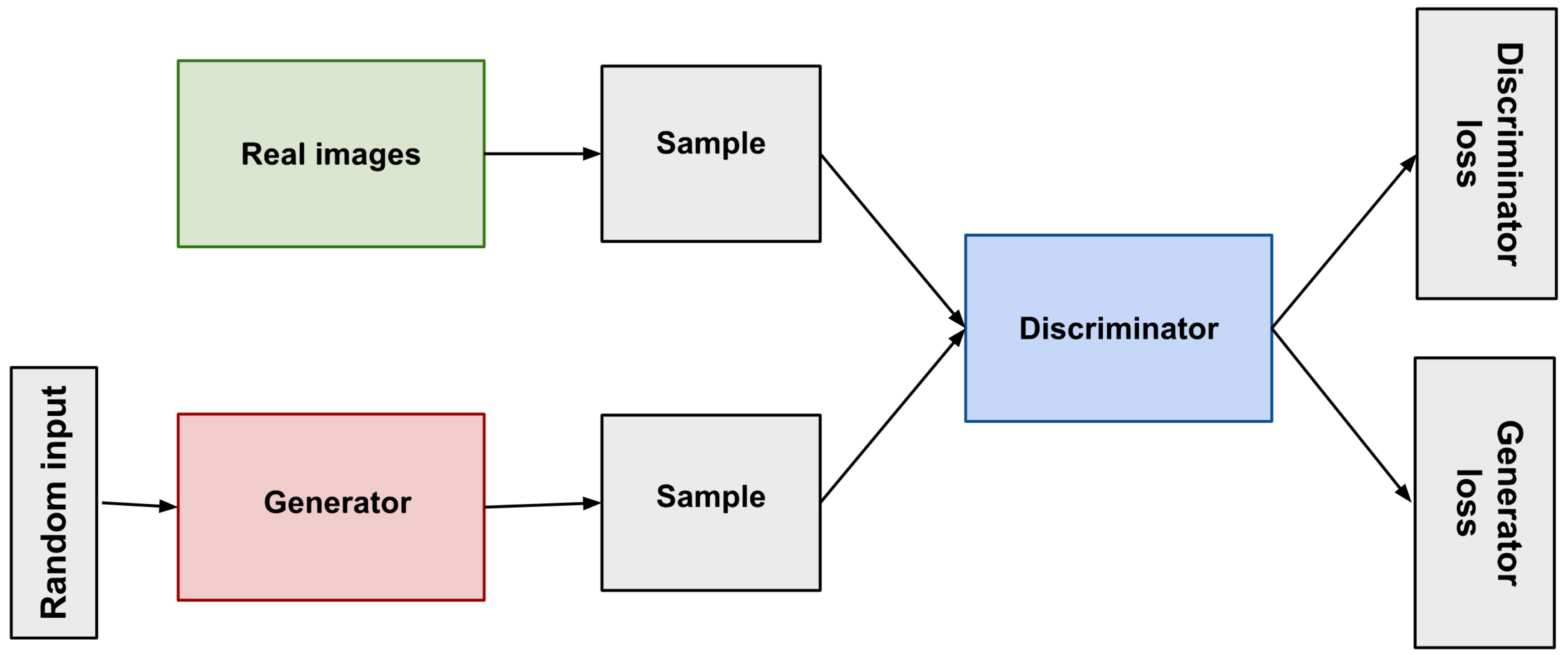


$$L_{GAN}(G, D) = E_y[log\ D(y) + E_{x,z}[log(1 - D(G(x,z))]$$

Common issue: **mode collapse**

Image courtesy: Overcoming Mode Collapse and the Curse of Dimensionality. *Ke Li*

Background: Generative Adversarial Networks



$$L_{GAN}(G, D) = E_y[log\, D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Common issue: **mode collapse**

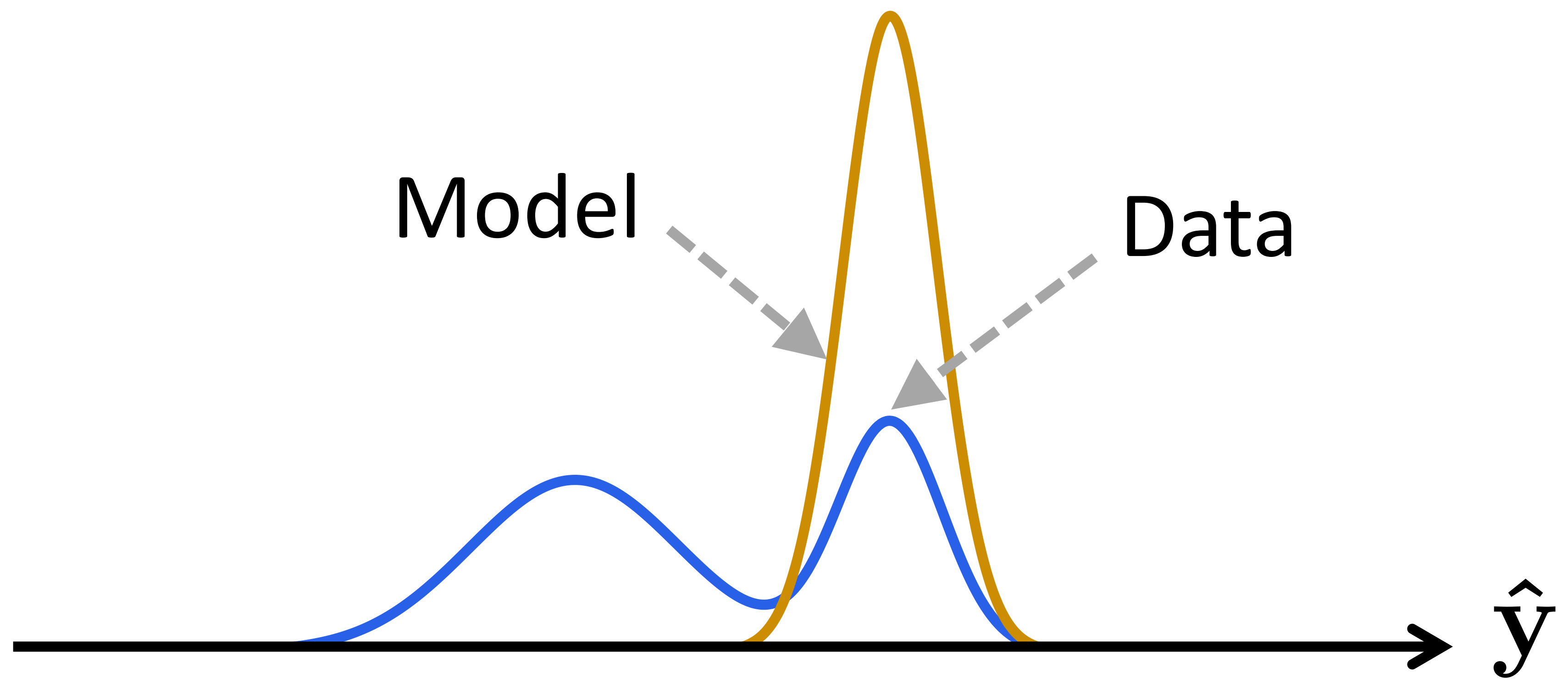
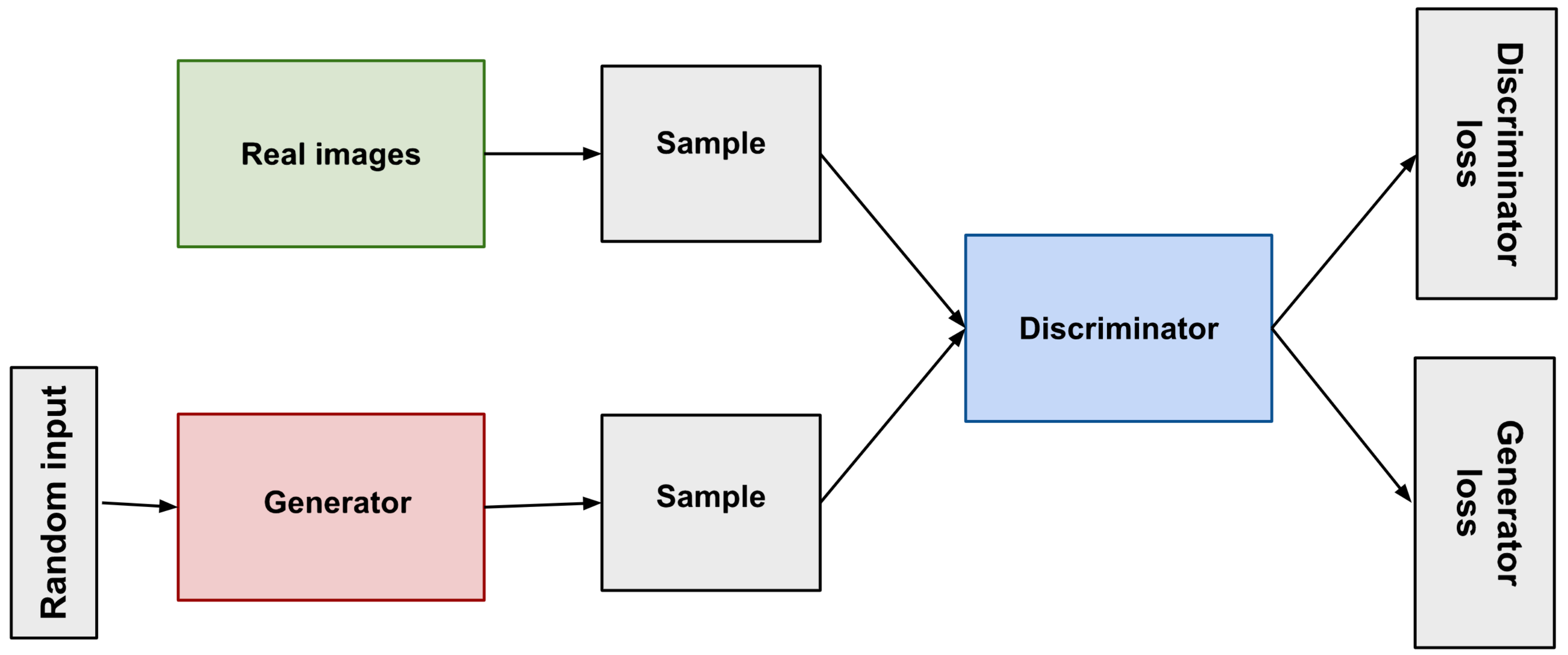


Image courtesy: Overcoming Mode Collapse and the Curse of Dimensionality. *Ke Li*

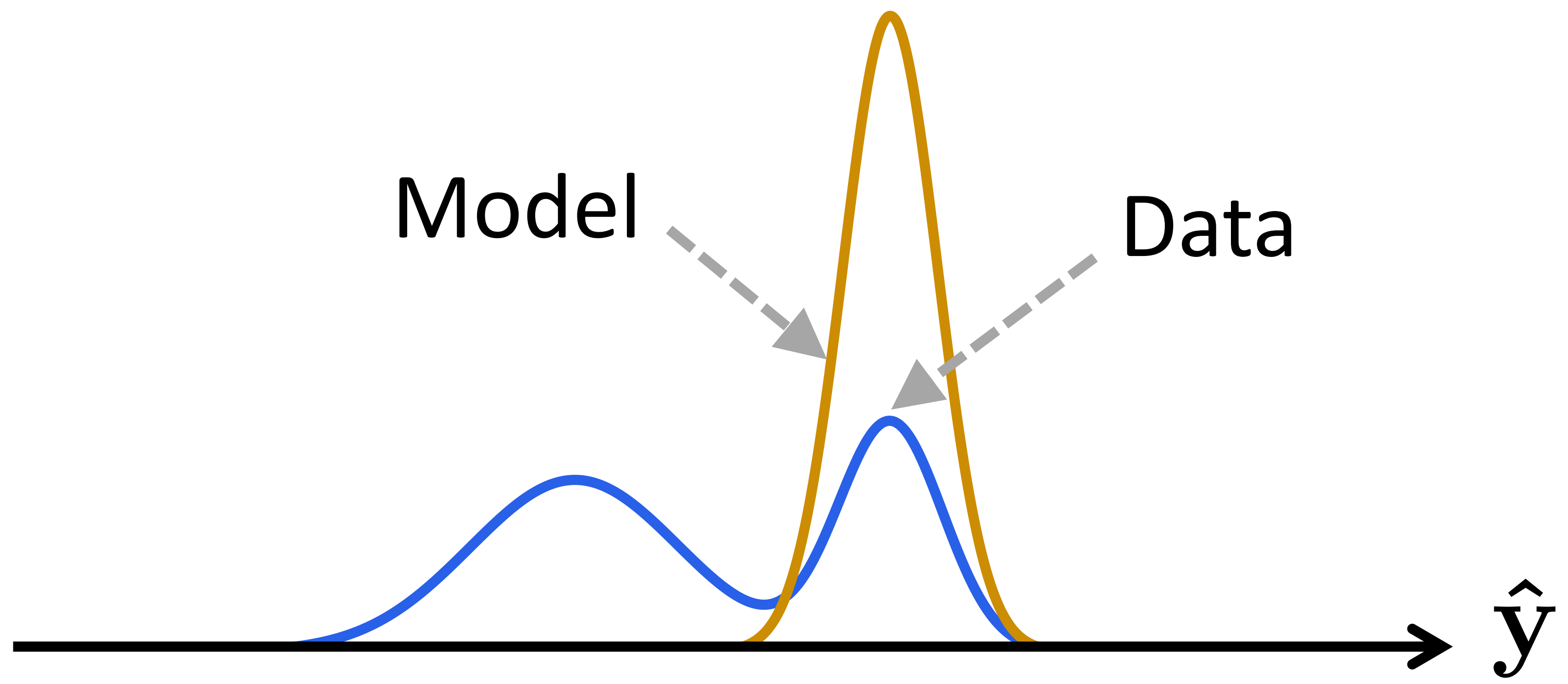
Slide Credit: Shubham Tulsiani

Background: Generative Adversarial Networks



$$L_{GAN}(G, D) = E_y[log\, D(y) + E_{x,z}[log(1 - D(G(x, z))]$$

Common issue: **mode collapse**



GANs don’t maximize likelihood of data

Image courtesy: Overcoming Mode Collapse and the Curse of Dimensionality. *Ke Li*

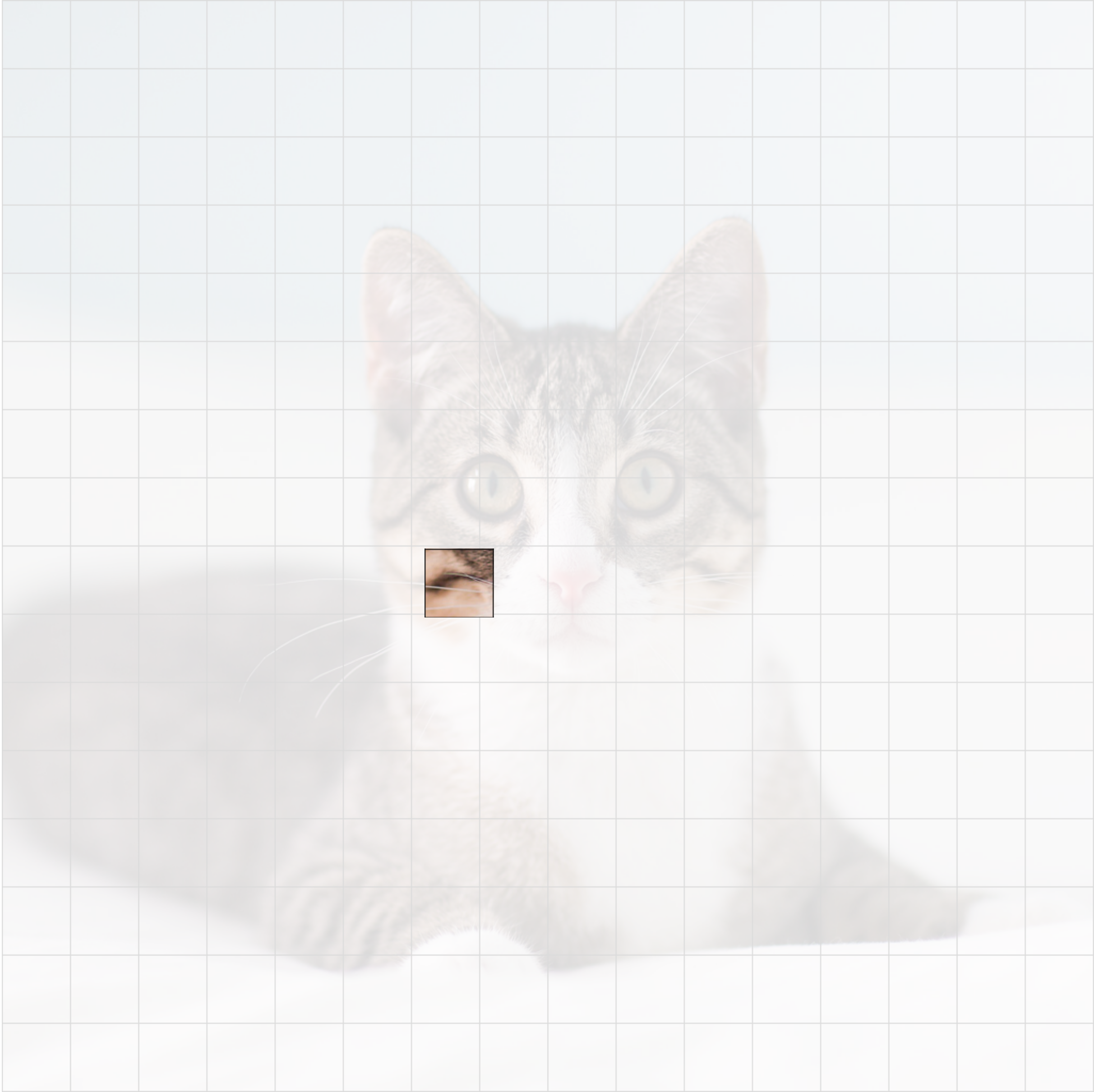
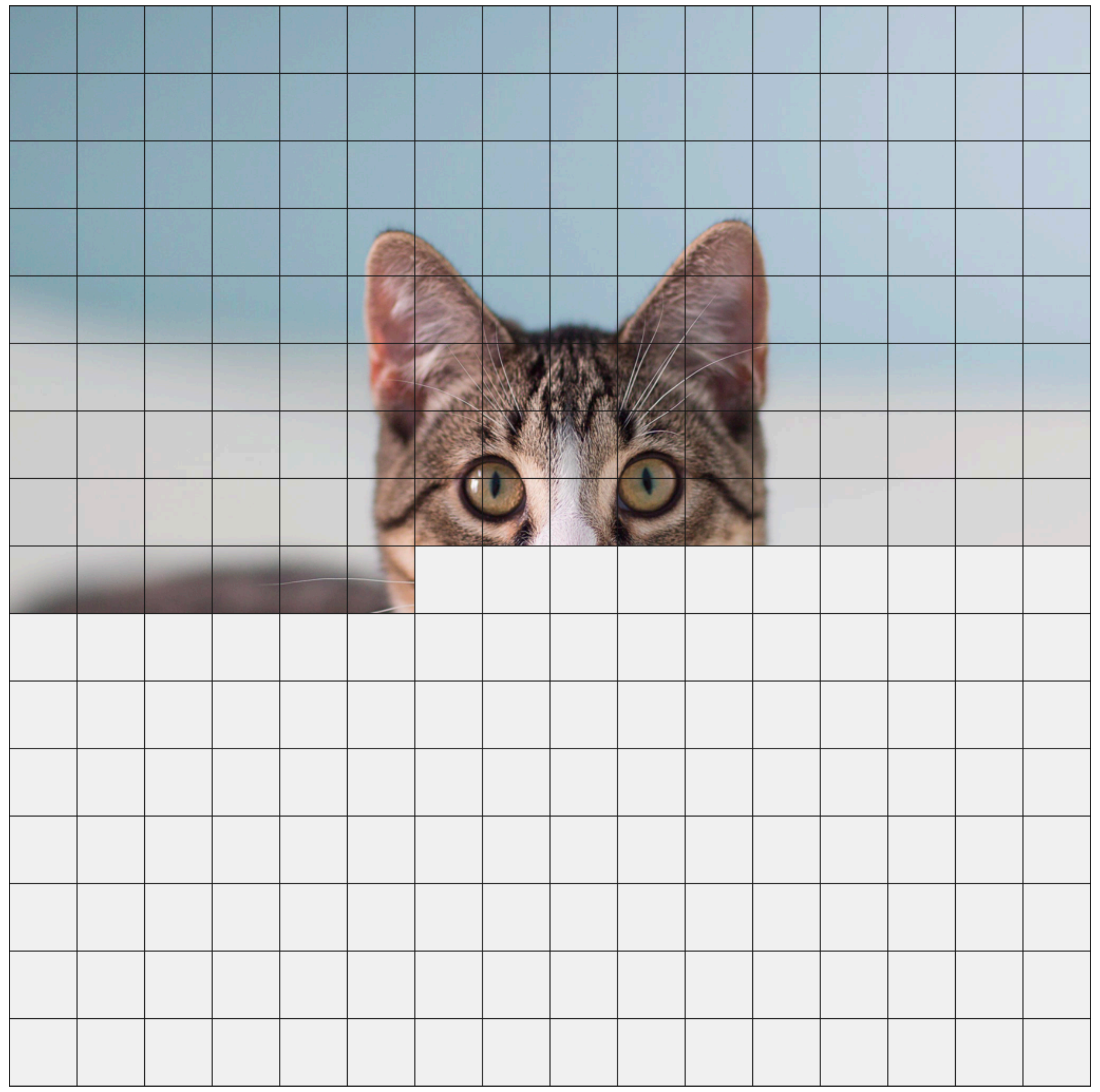
Slide Credit: Shubham Tulsiani

Background: Autoregressive Models

Training: $\max \mathbb{E}_{x \sim \mathcal{D}} \log \ p_\theta(x)$

Image courtesy: PixelTransformer: Sample Conditioned Signal Generation. *Tulsiani et. al.*

Background: Autoregressive Models

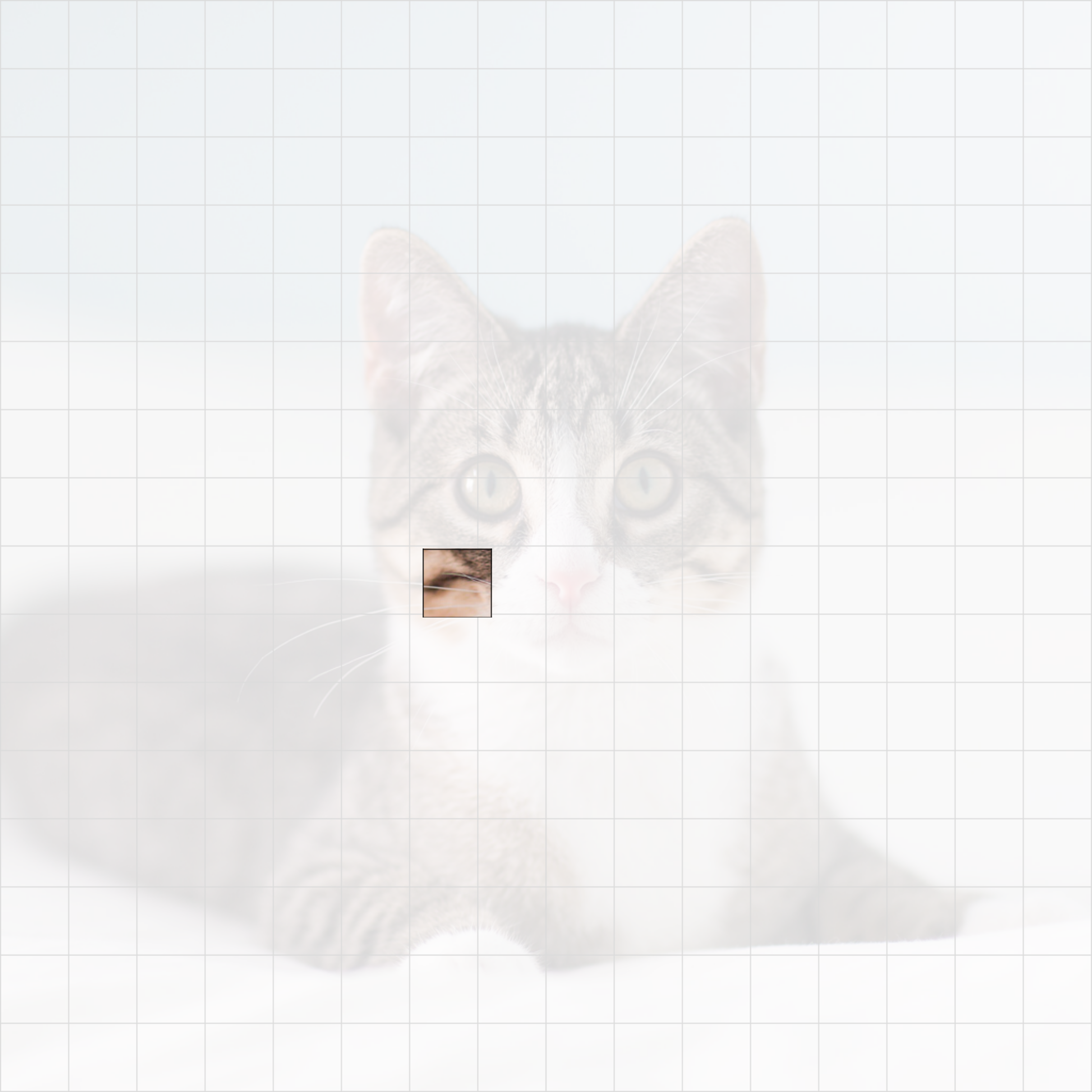
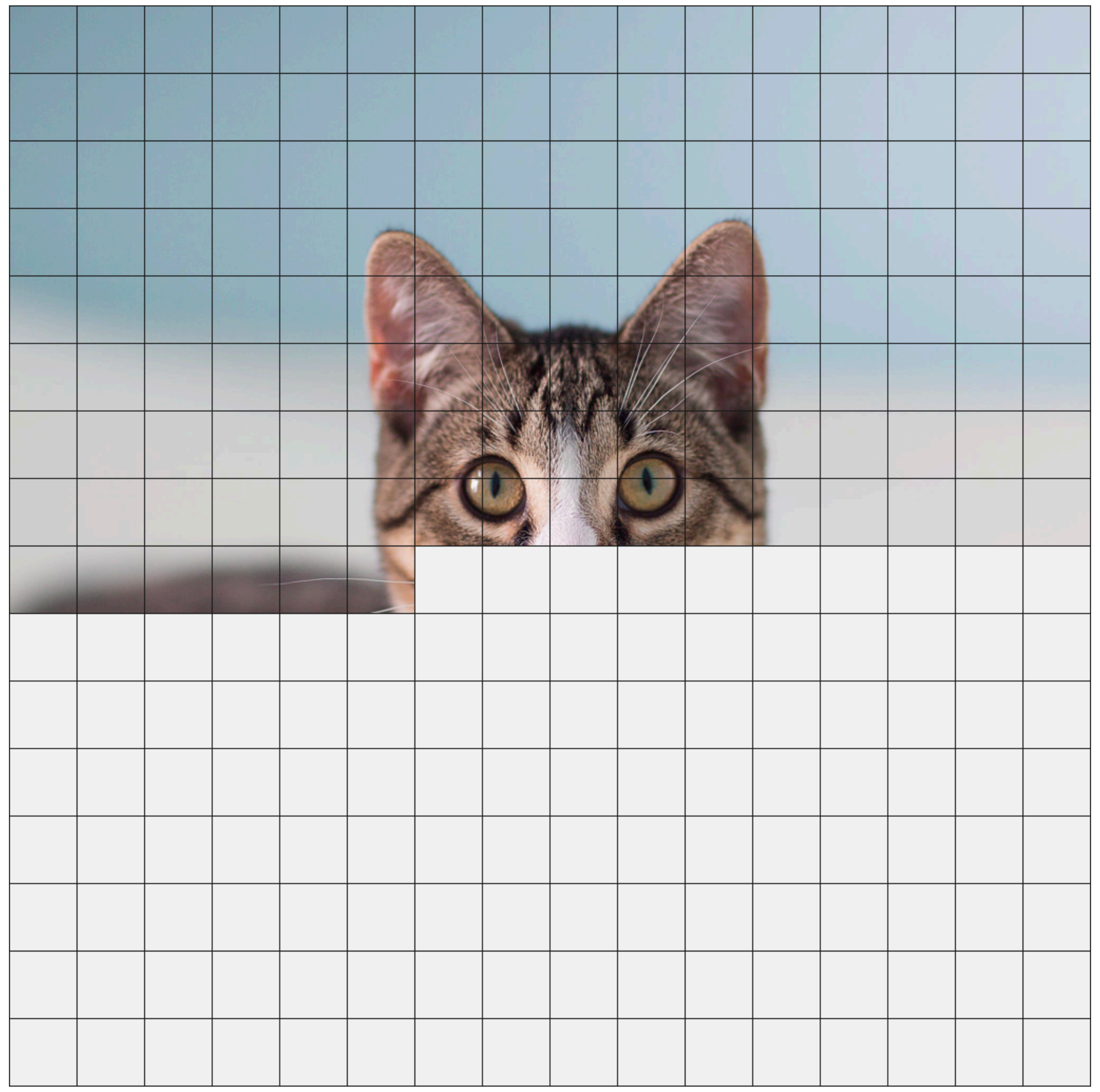
$p\Big($  $\Big|$  $\Big)$

Training: $\max \mathbb{E}_{x \sim \mathcal{D}} \; \log \; p_\theta(x)$

Image courtesy: PixelTransformer: Sample Conditioned Signal Generation. *Tulsiani et. al.*

Slide Credit: Shubham Tulsiani

Background: Autoregressive Models

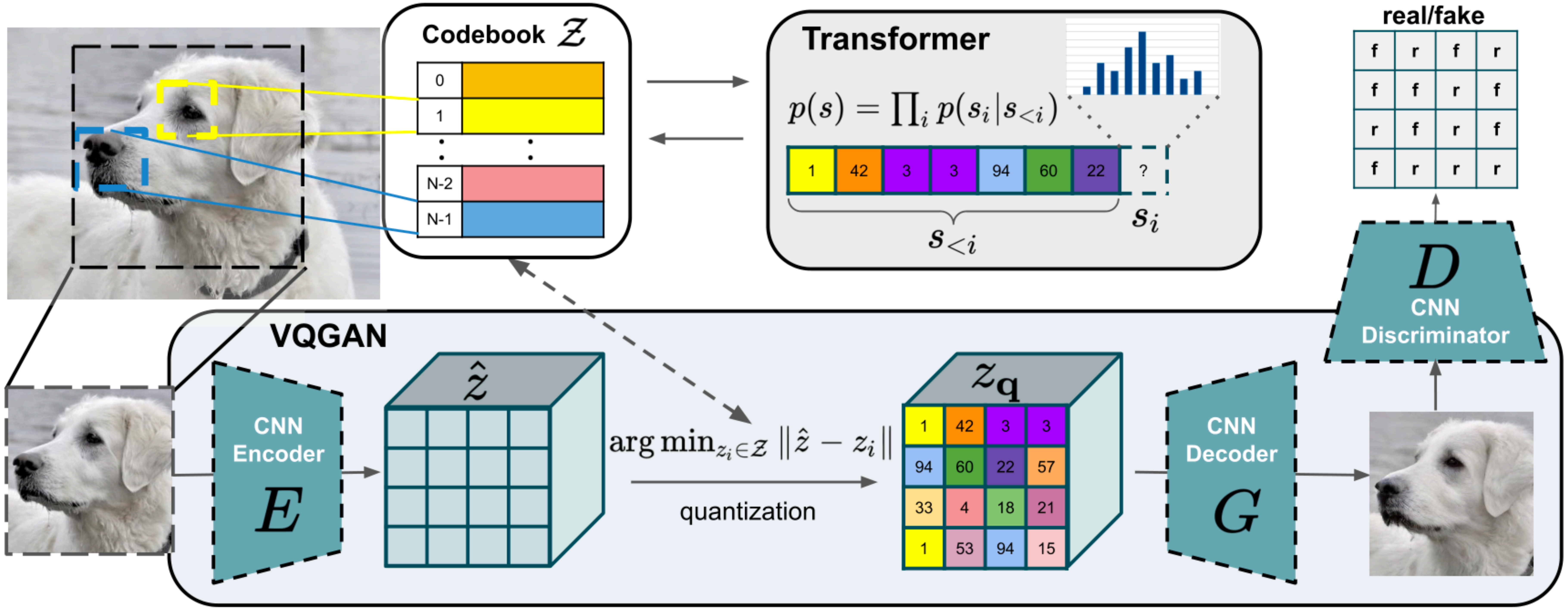
$p($  $|$  $)$

Training: $\max \mathbb{E}_{x \sim \mathcal{D}} \ \log \ p_\theta(x)$

Inference: Sample images, one pixel at a time

Image courtesy: PixelTransformer: Sample Conditioned Signal Generation. *Tulsiani et. al.*

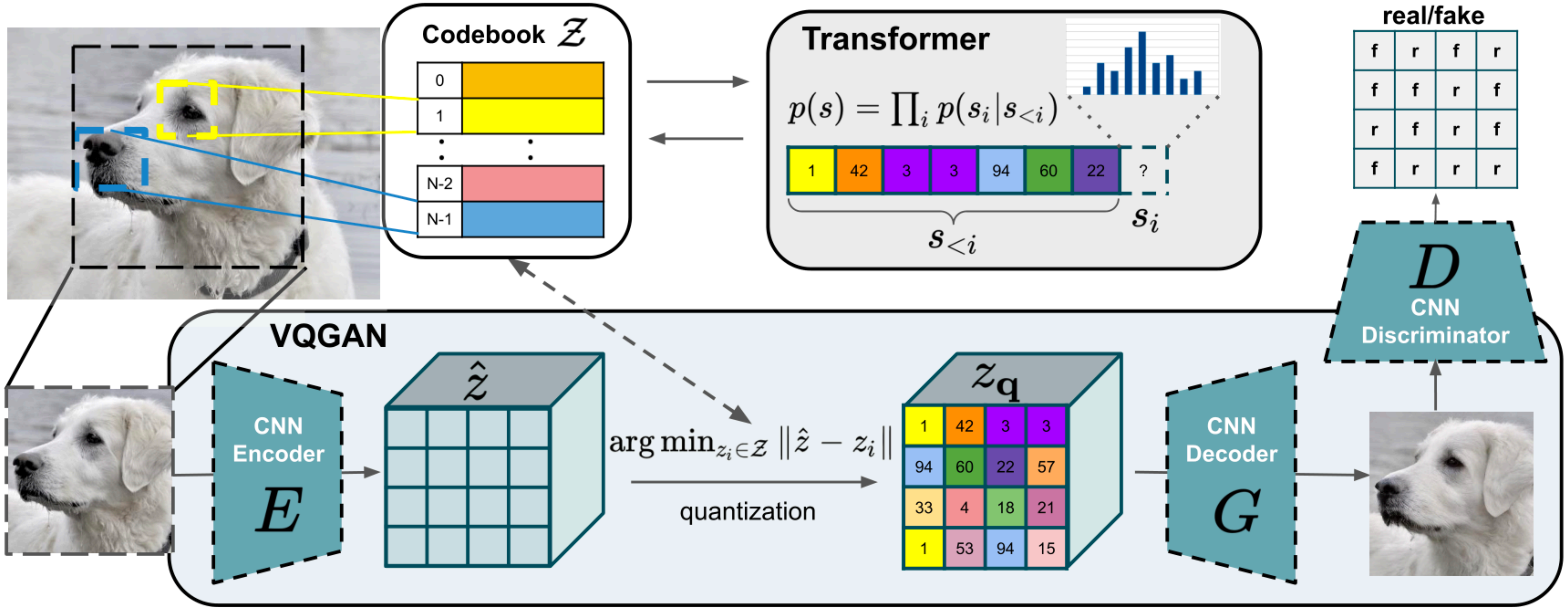
Background: Autoregressive Models



Taming Transformers for High-Resolution Image Synthesis. *Esser et. al.*

Slide Credit: Shubham Tulsiani

Background: Autoregressive Models

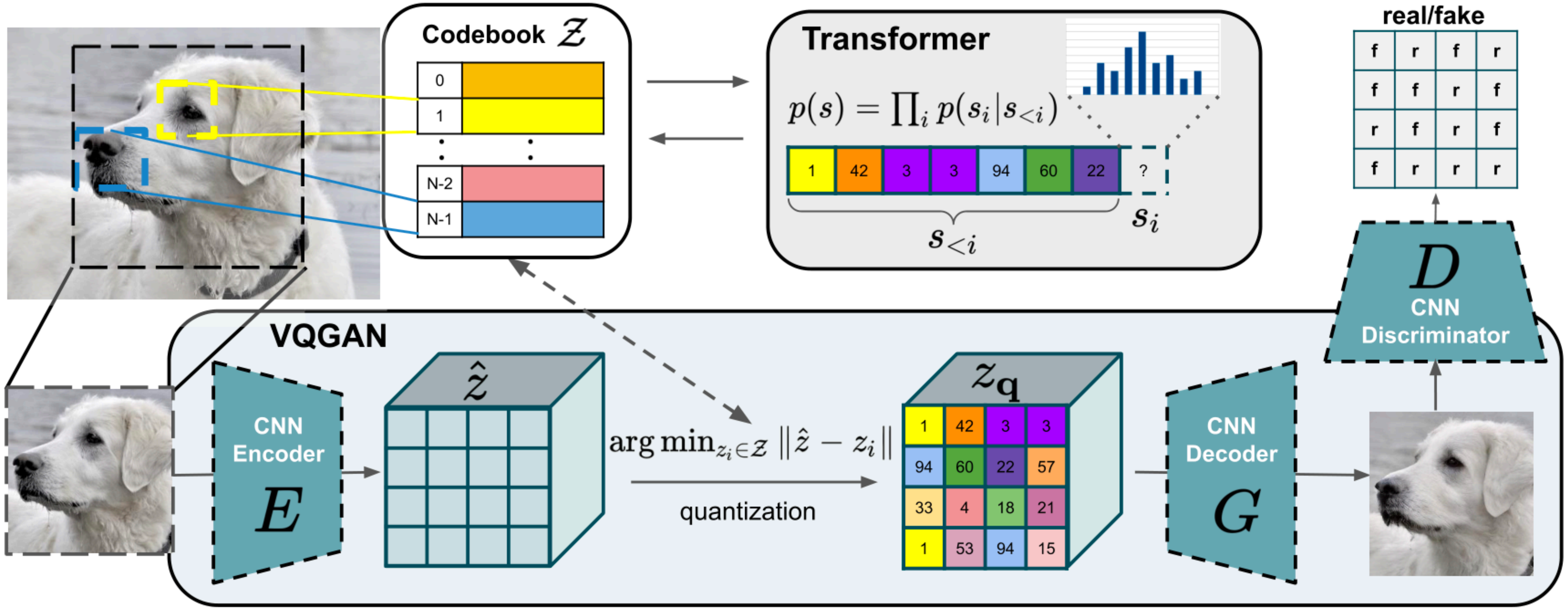


Step 1: Learn a 'codebook' of discrete patch representations

Taming Transformers for High-Resolution Image Synthesis. *Esser et. al.*

Slide Credit: Shubham Tulsiani

Background: Autoregressive Models



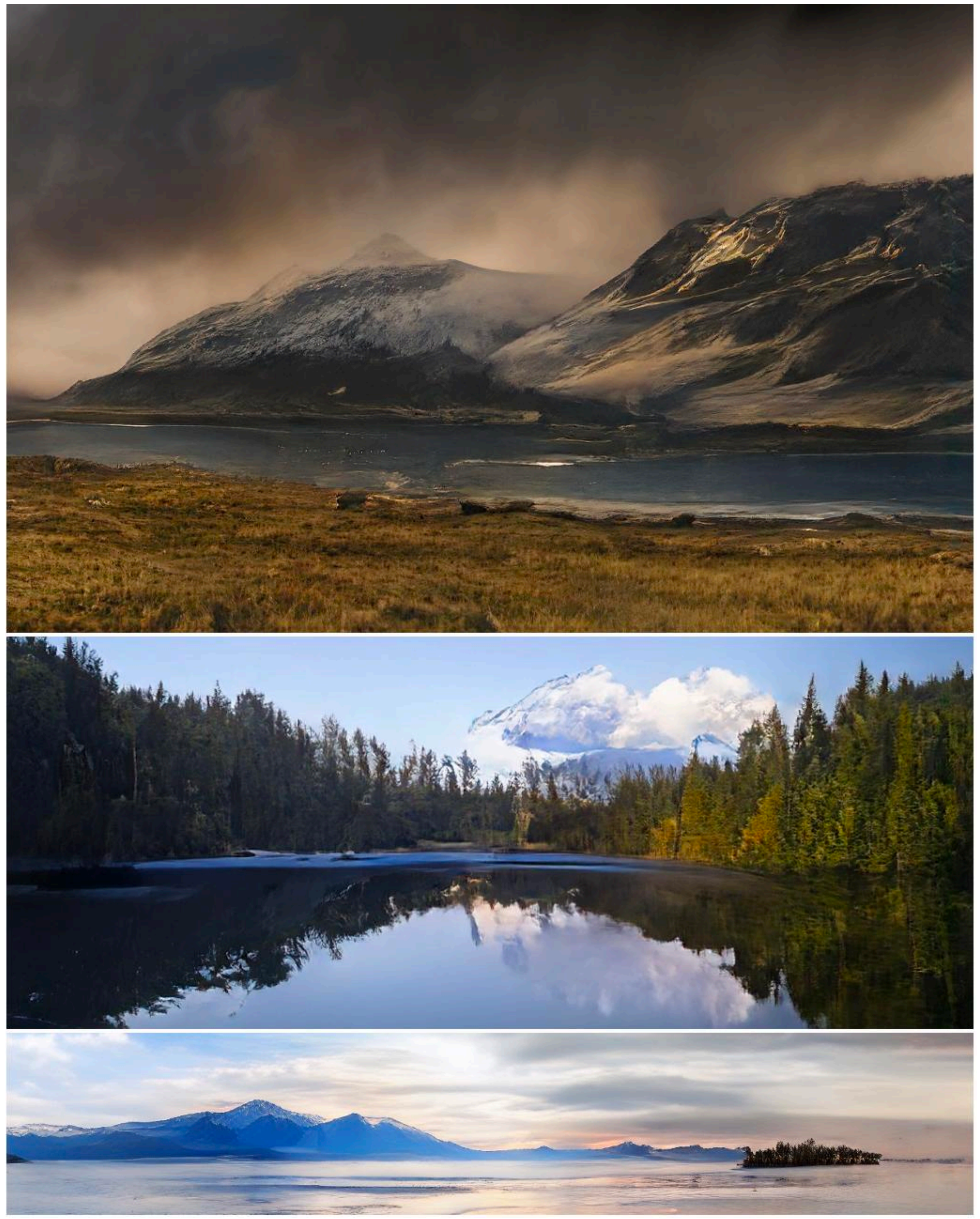
Step 1: Learn a 'codebook' of discrete patch representations

Step 2: Learn transformer-based Autoregressive models

Taming Transformers for High-Resolution Image Synthesis. *Esser et. al.*

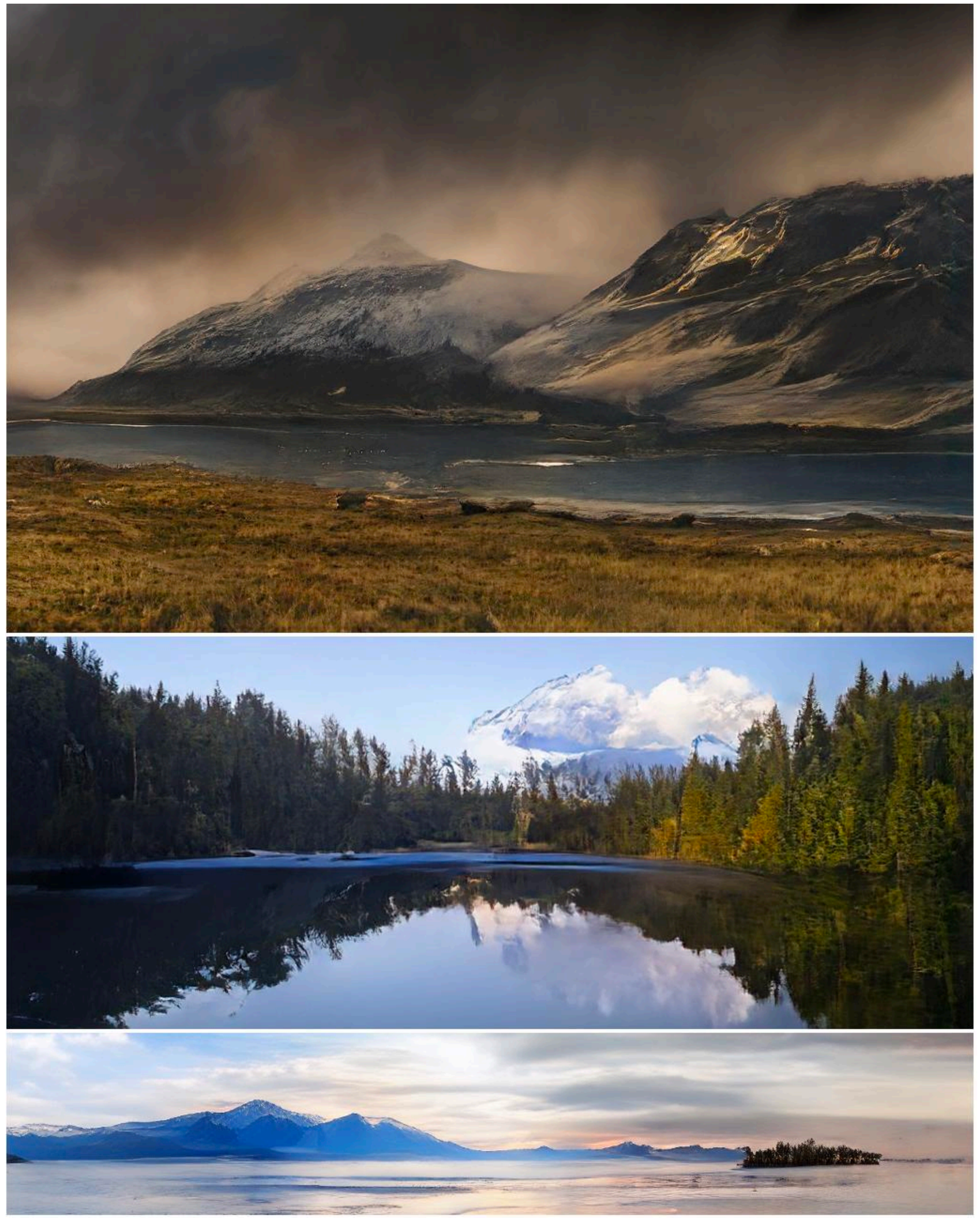
Slide Credit: Shubham Tulsiani

Background: Autoregressive Models



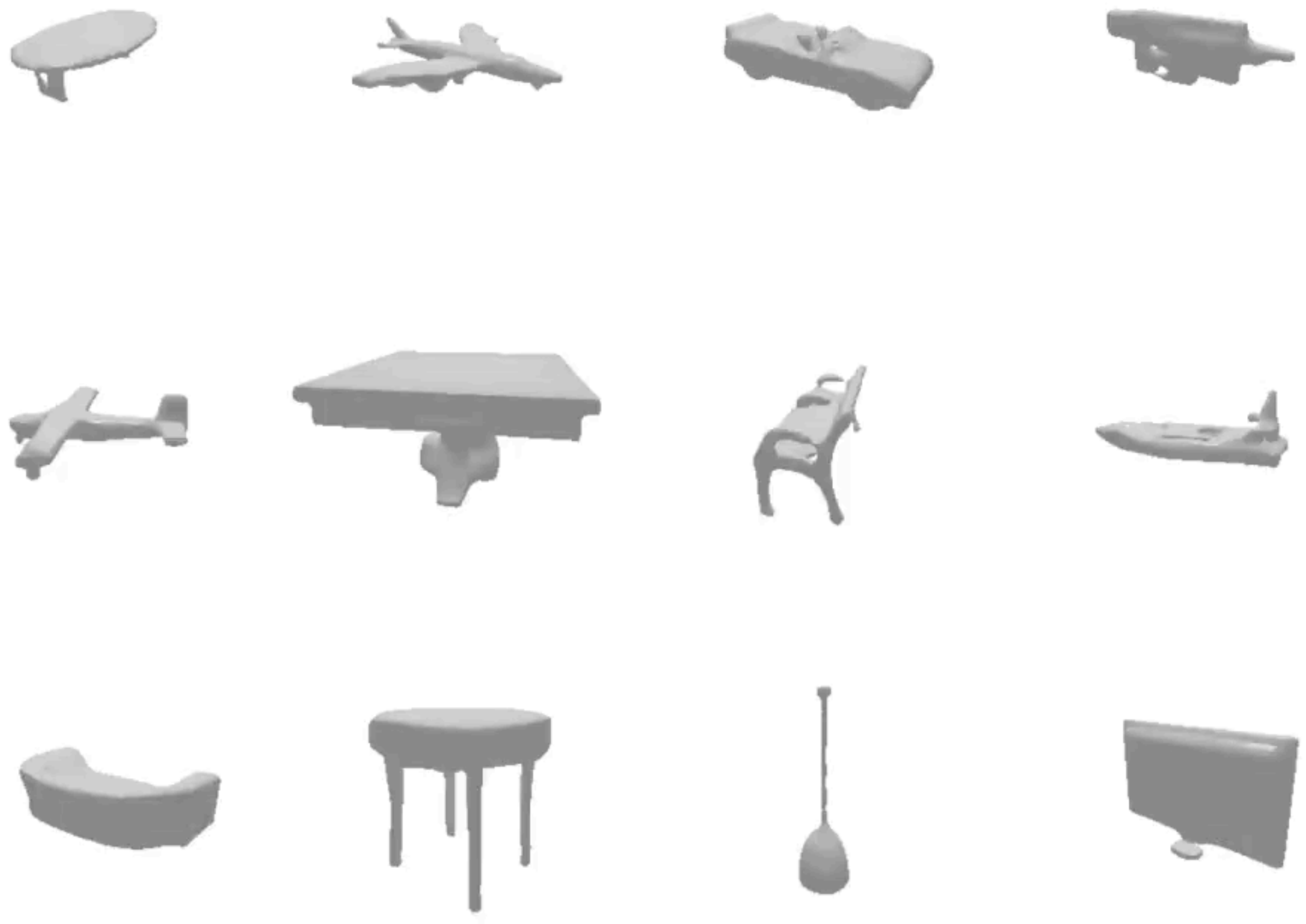
Taming Transformers for High-Resolution Image Synthesis. *Esser et. al.*

Background: Autoregressive Models



Taming Transformers for High-Resolution Image Synthesis. *Esser et. al.*

Autoregressive Generation of 3D Shapes

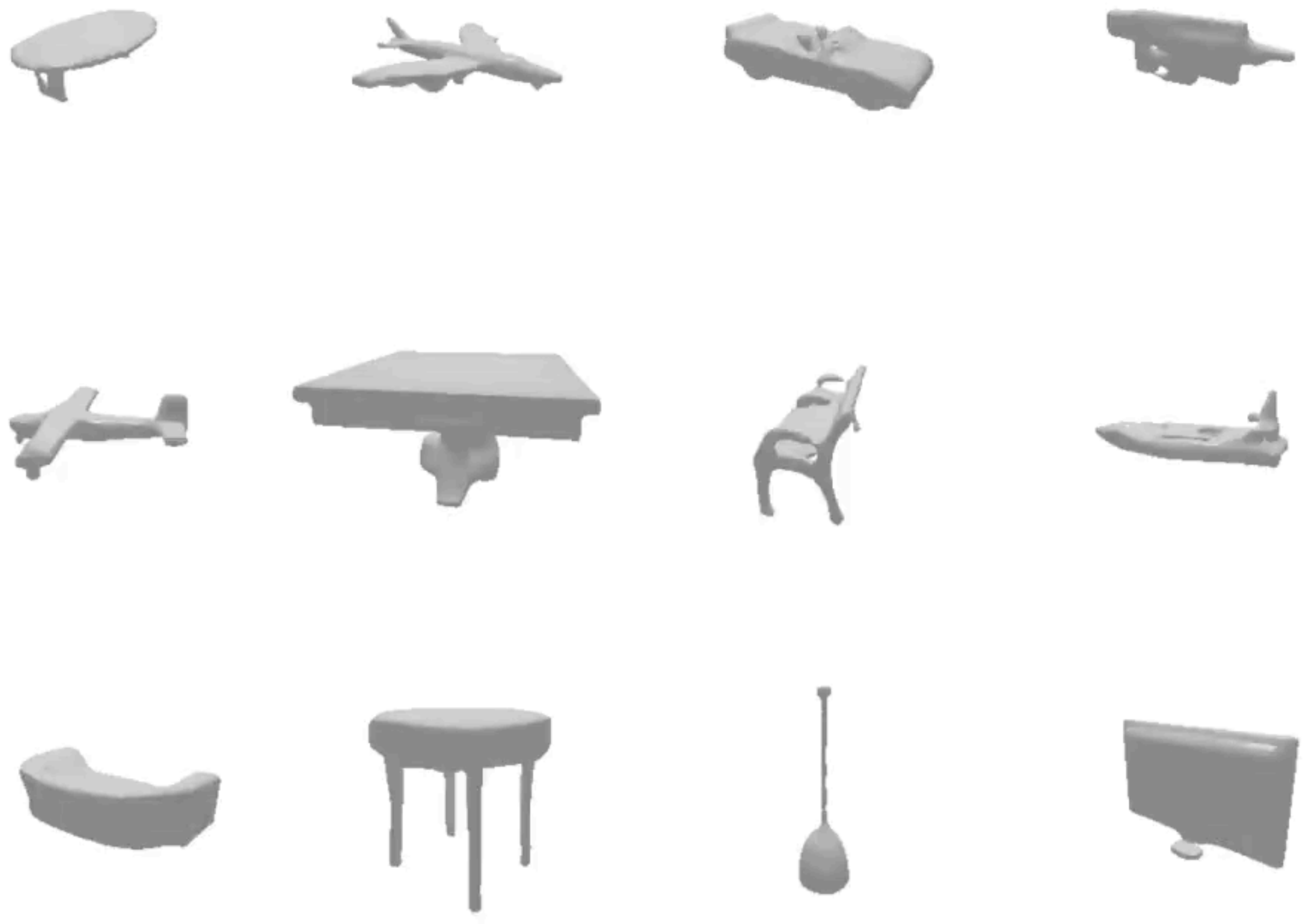


Unconditional generation from model trained over 50 categories

AutoSDF: Shape Priors for 3D Completion, Reconstruction and Generation. *Mittal*, Cheng* et. al.*

Slide Credit: Shubham Tulsiani

Autoregressive Generation of 3D Shapes



Unconditional generation from model trained over 50 categories

AutoSDF: Shape Priors for 3D Completion, Reconstruction and Generation. *Mittal*, Cheng* et. al.*

Slide Credit: Shubham Tulsiani

Autoregressive Generation of 3D Shapes

Trivially allows shape completion (missing regions in red)

AutoSDF: Shape Priors for 3D Completion, Reconstruction and Generation. *Mittal*, Cheng* et. al.*

Slide Credit: Shubham Tulsiani

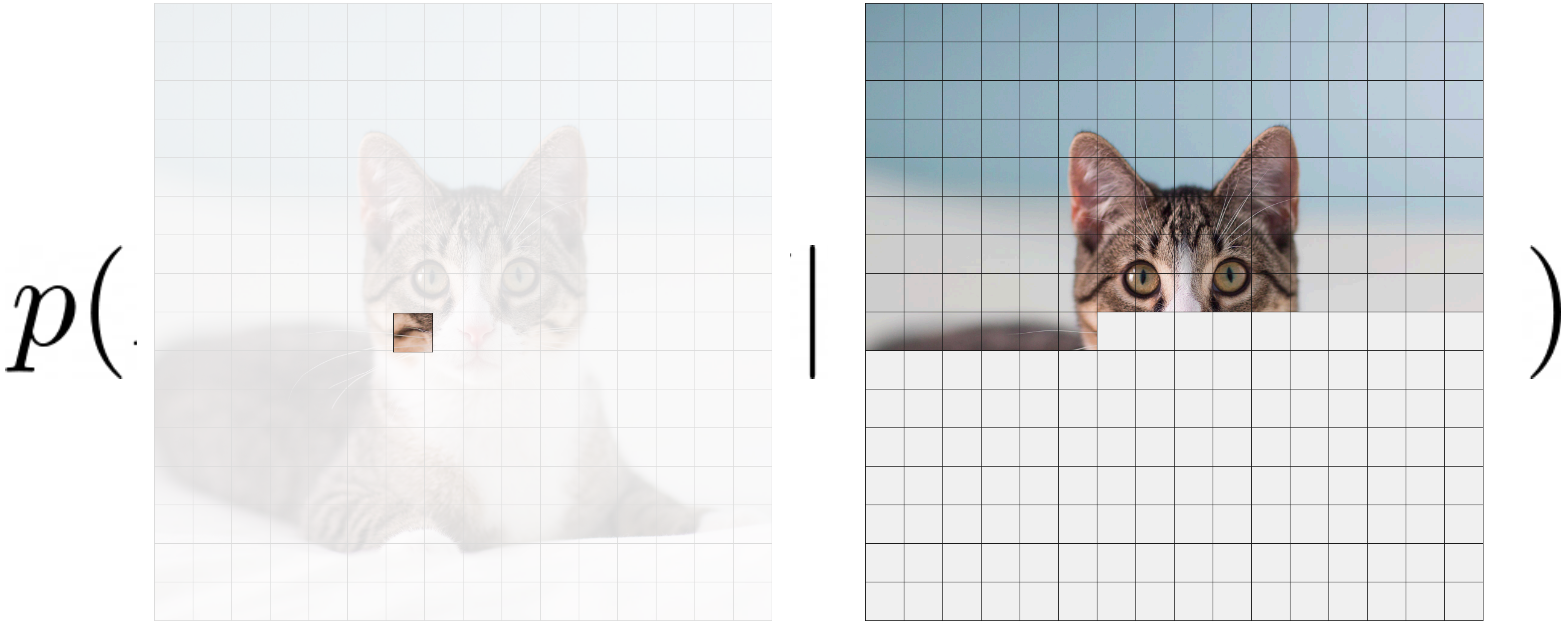
Autoregressive Generation of 3D Shapes

Trivially allows shape completion (missing regions in red)

AutoSDF: Shape Priors for 3D Completion, Reconstruction and Generation. *Mittal*, Cheng* et. al.*

Slide Credit: Shubham Tulsiani

Background: Autoregressive Models

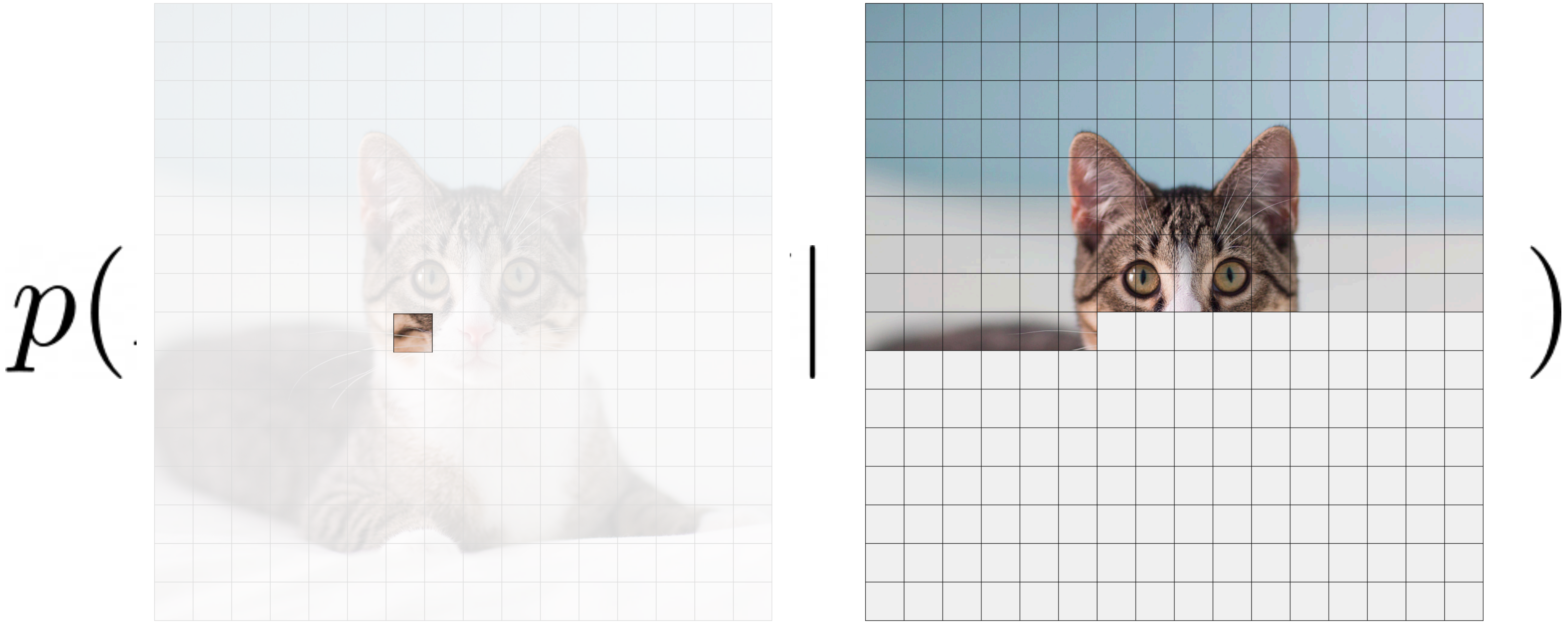


Training: $\max \mathbb{E}_{x \sim \mathcal{D}} \log \ p_\theta(x)$

Image courtesy: PixelTransformer: Sample Conditioned Signal Generation. *Tulsiani et. al.*

Slide Credit: Shubham Tulsiani

Background: Autoregressive Models

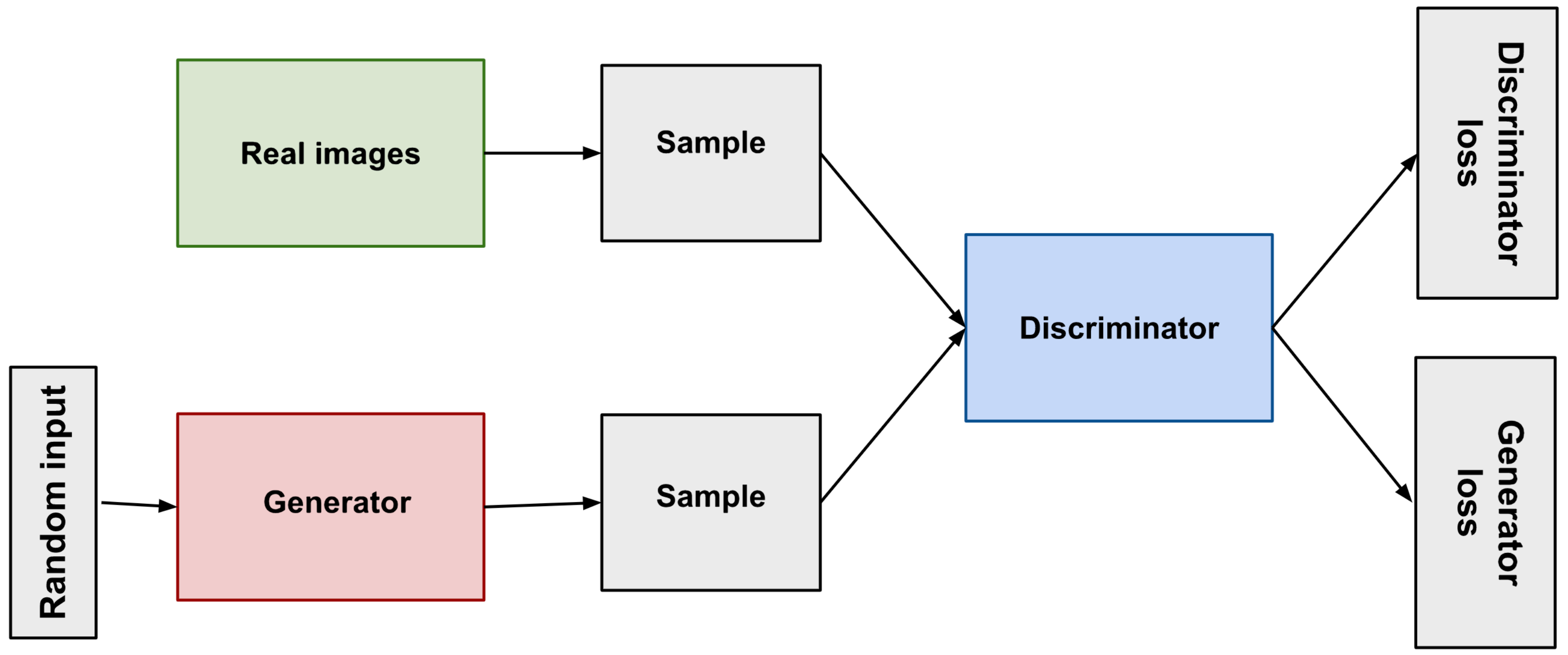


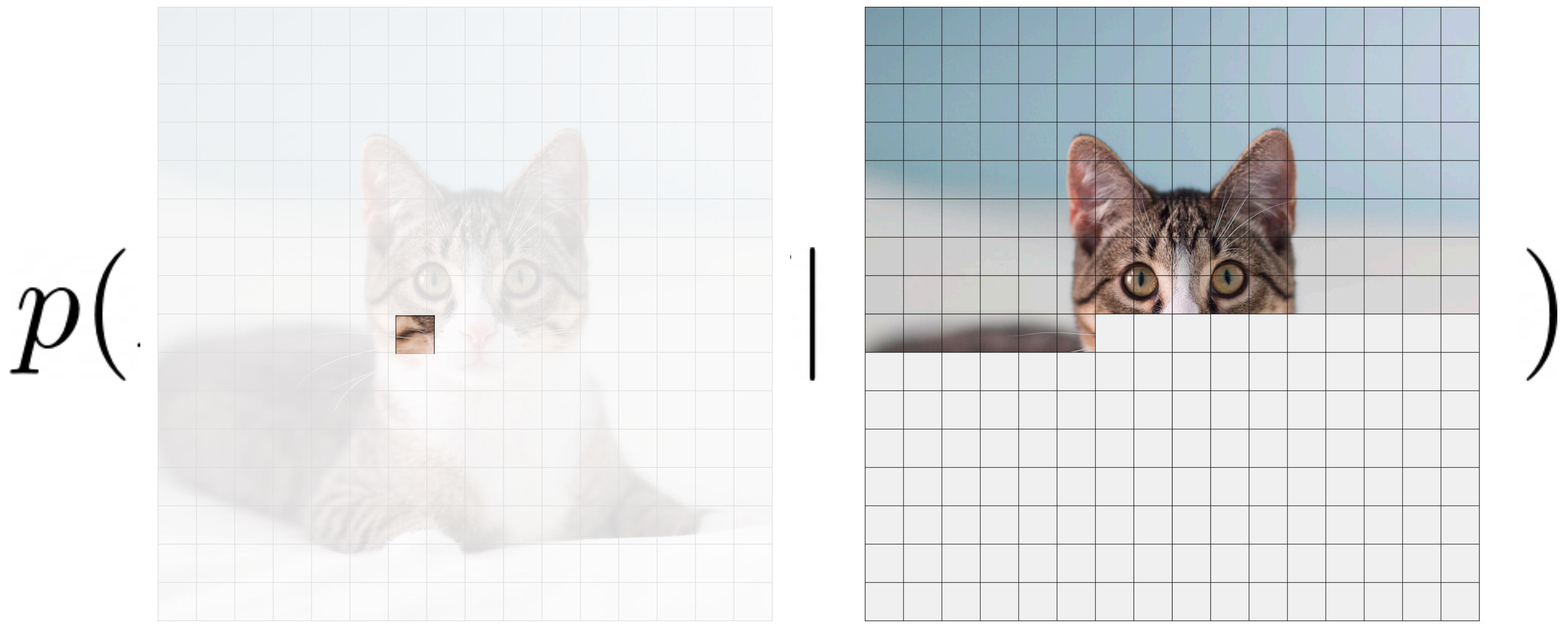
Training: $\max \mathbb{E}_{x \sim \mathcal{D}} \log \; p_\theta(x)$

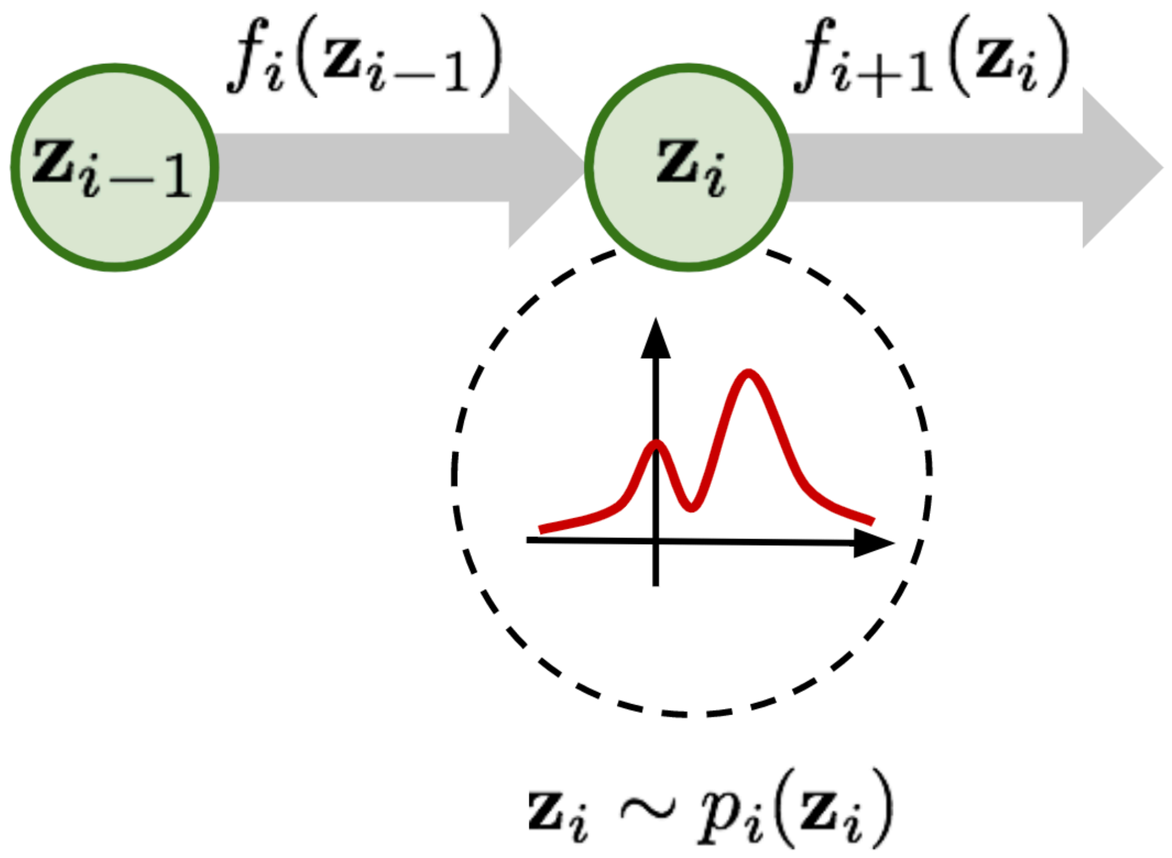
No latent space for interpolation, arithmetic etc.

Image courtesy: PixelTransformer: Sample Conditioned Signal Generation. *Tulsiani et. al.*

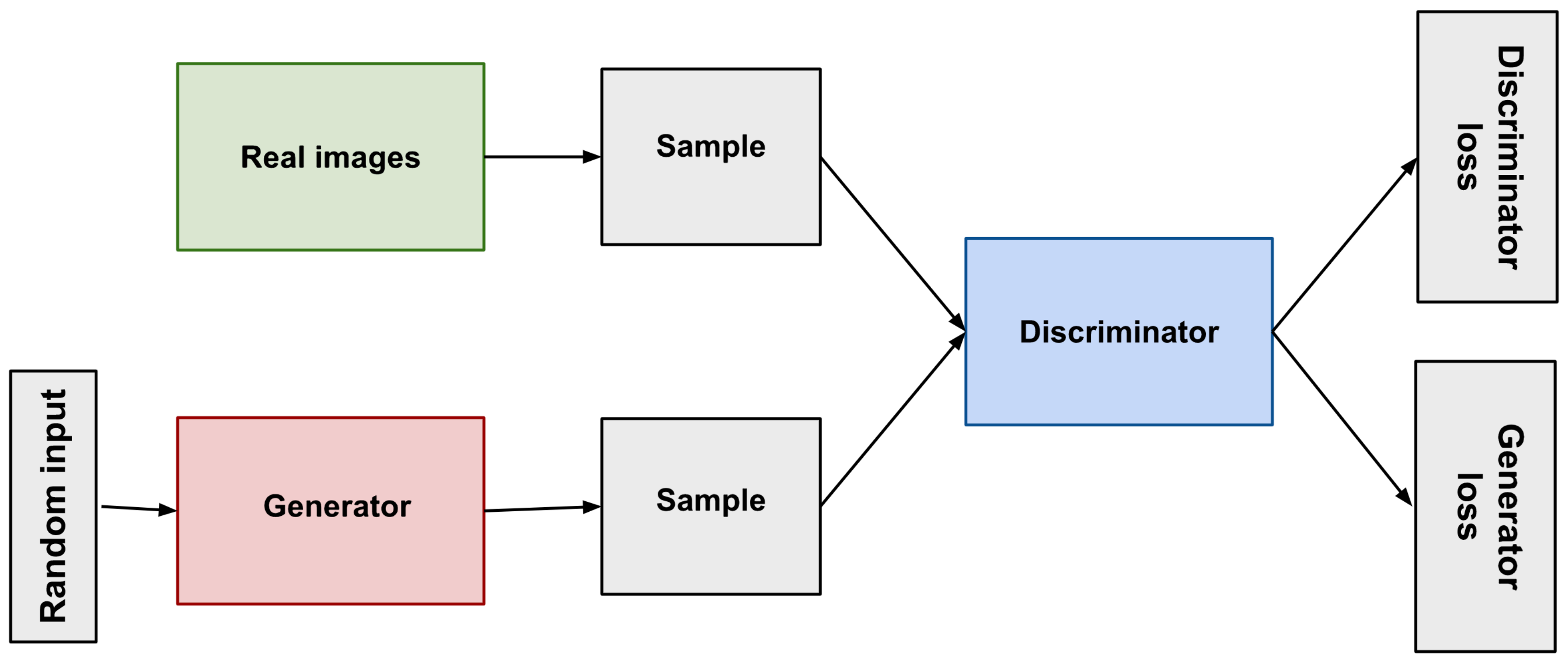
Generating 3D Shapes

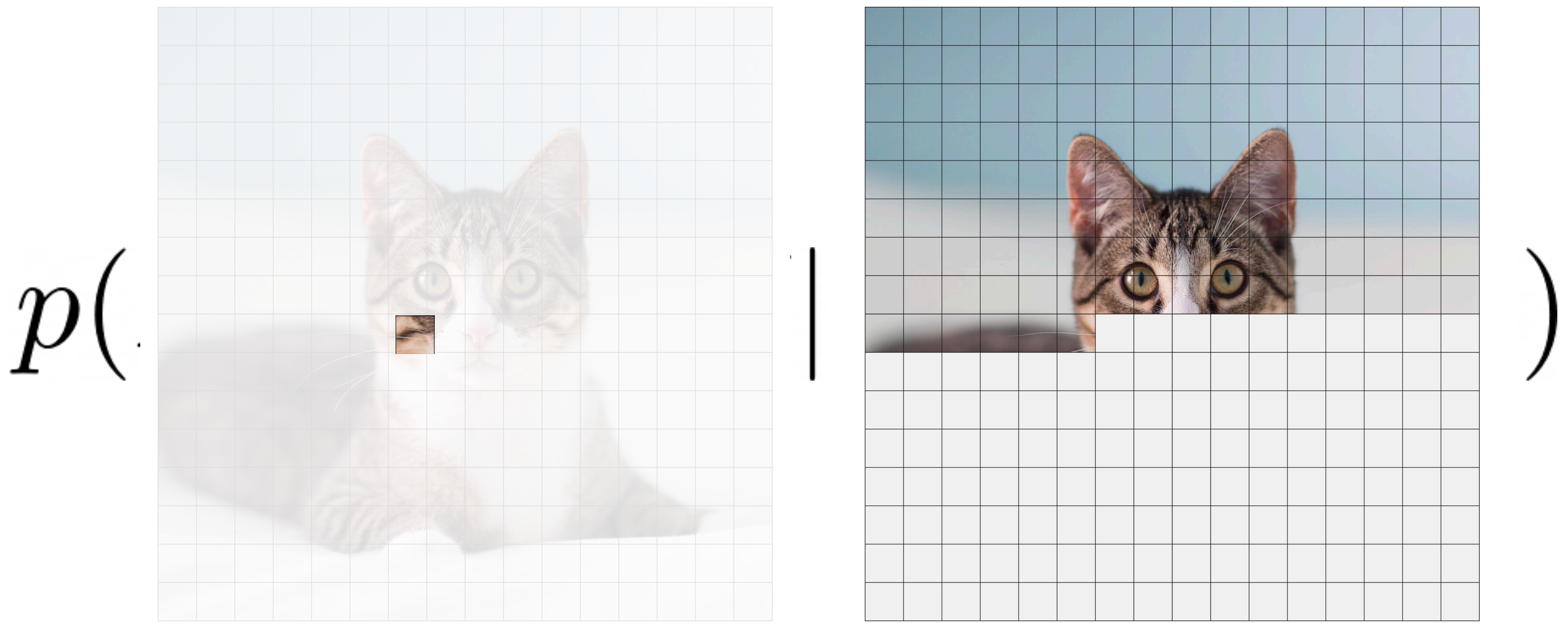


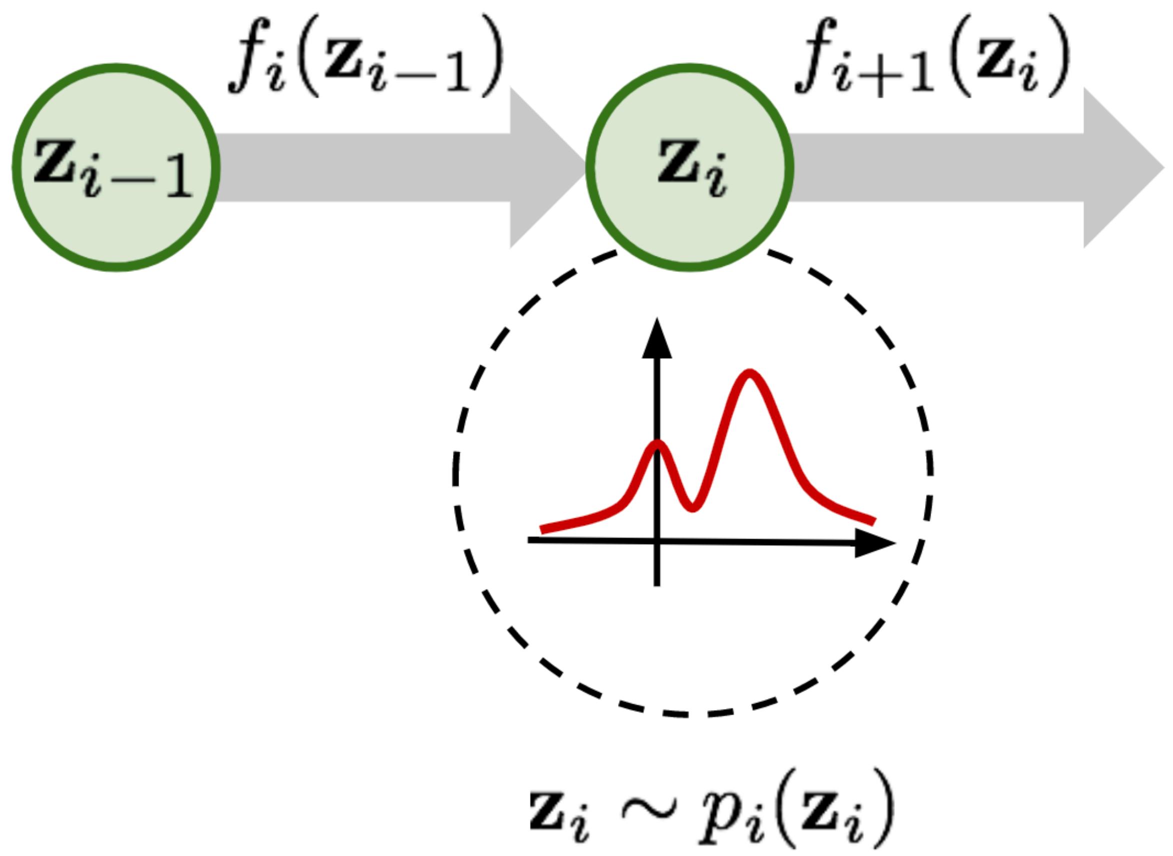




Generating 3D Shapes

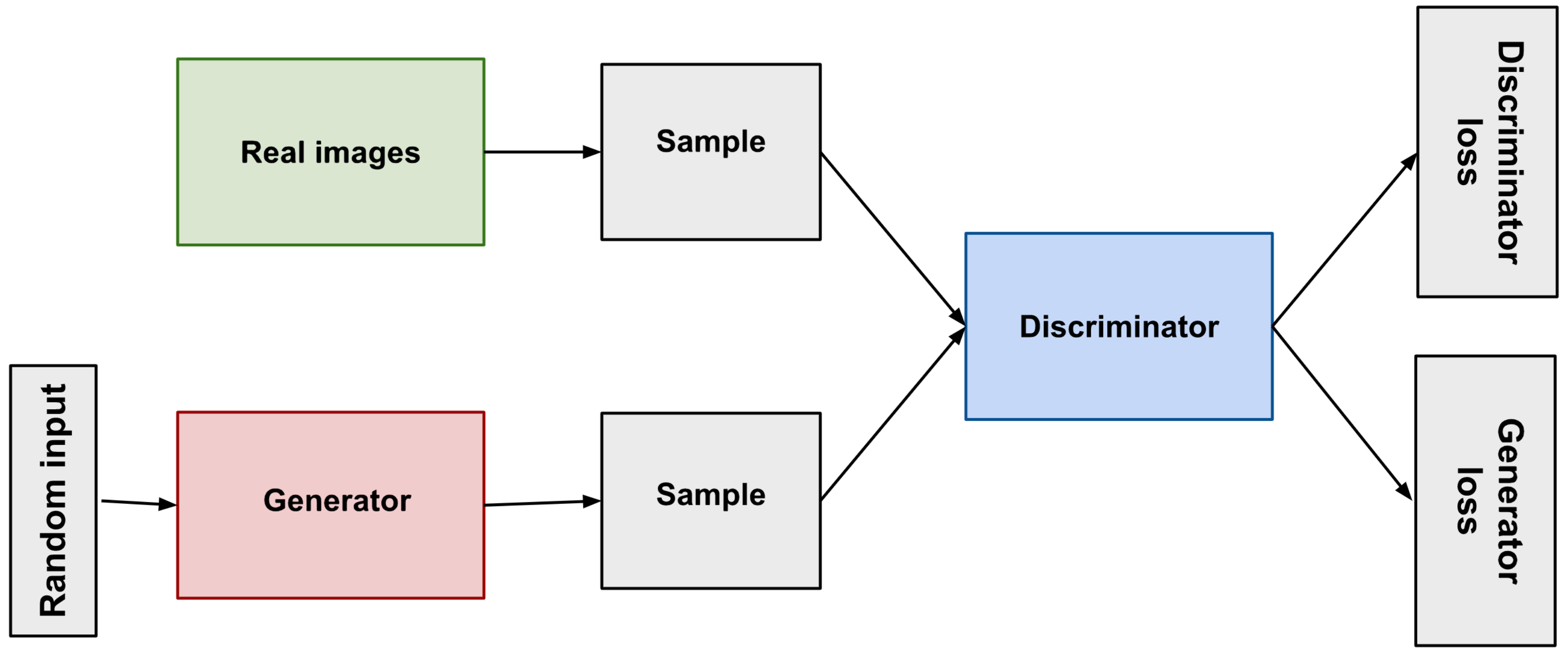


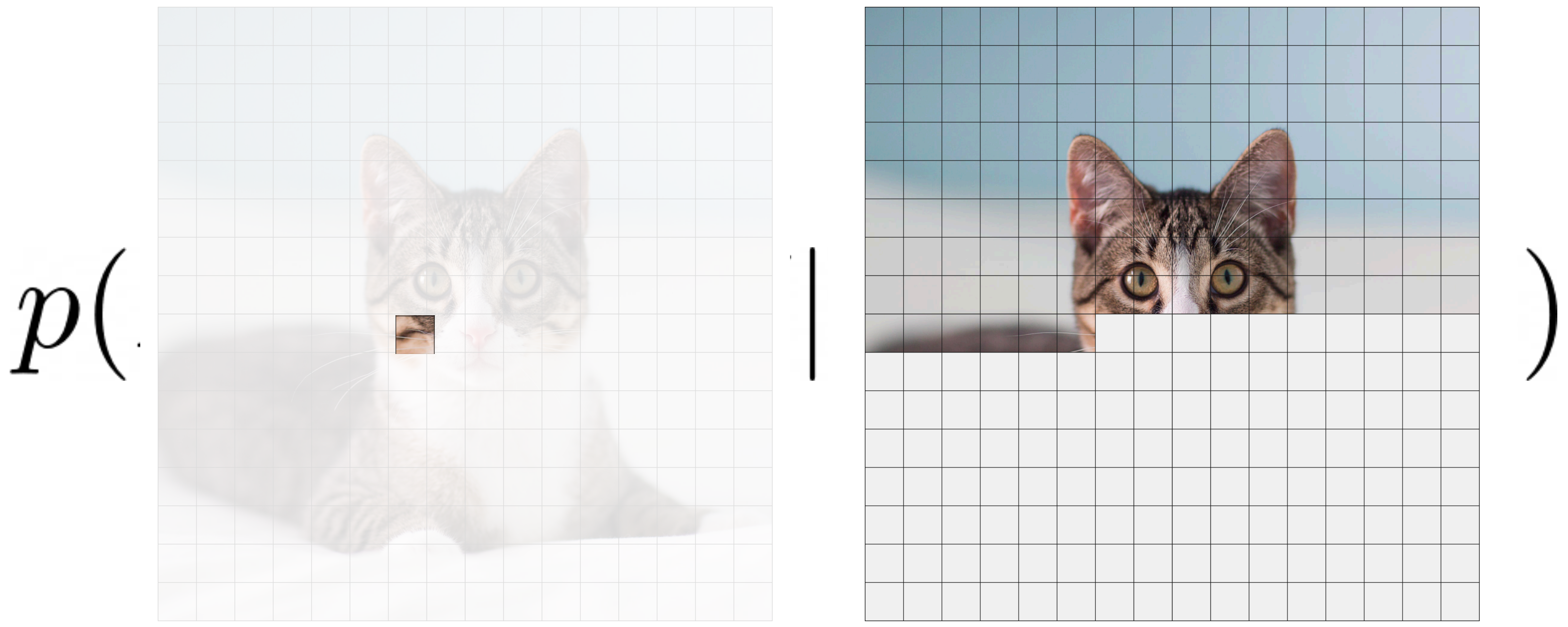


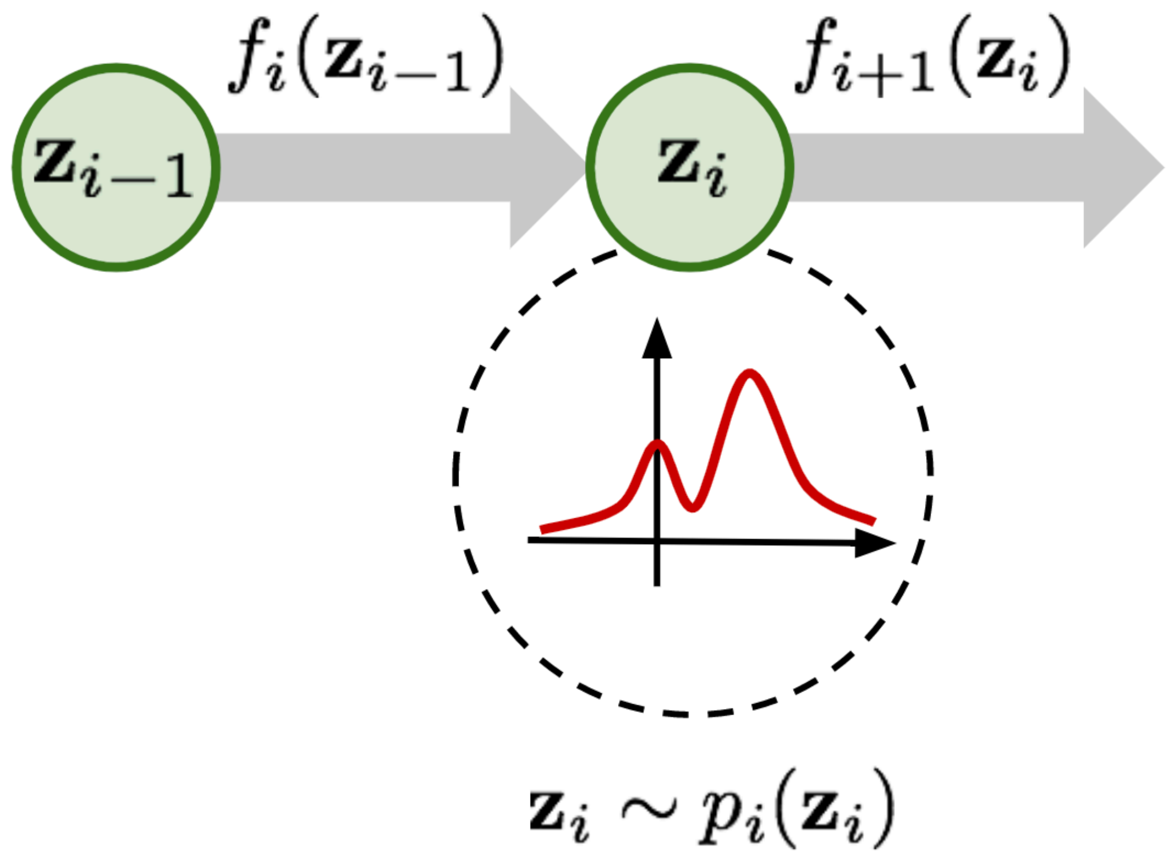


Step 1: Choose your favorite 2D generation model

Generating 3D Shapes





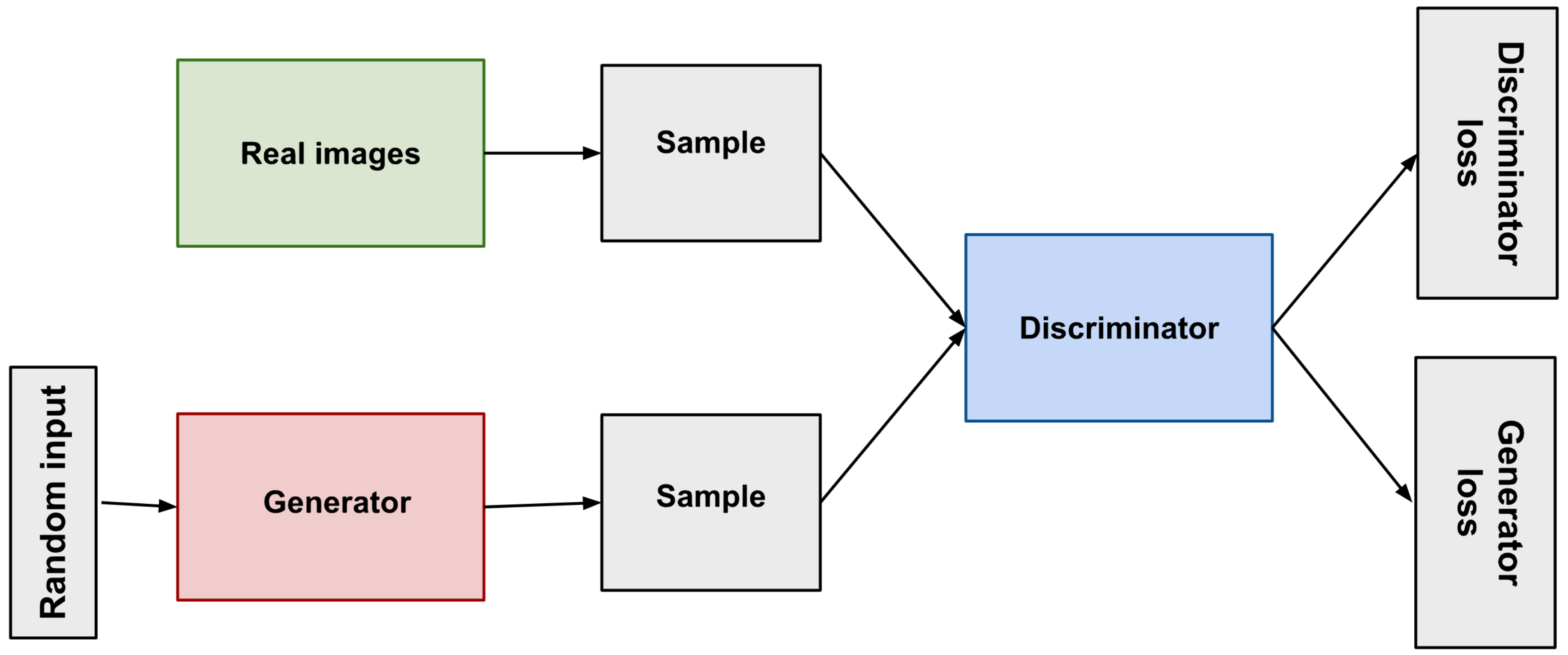


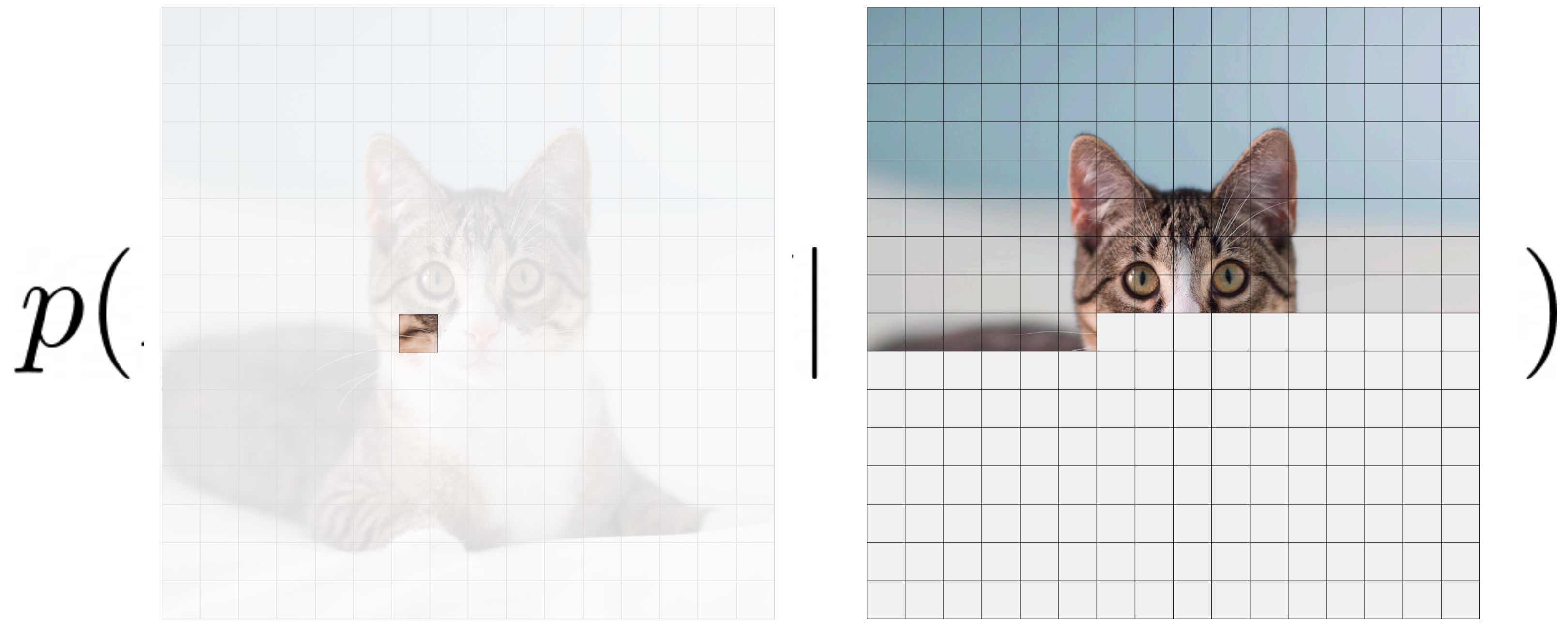
Step 1: Choose your favorite 2D generation model

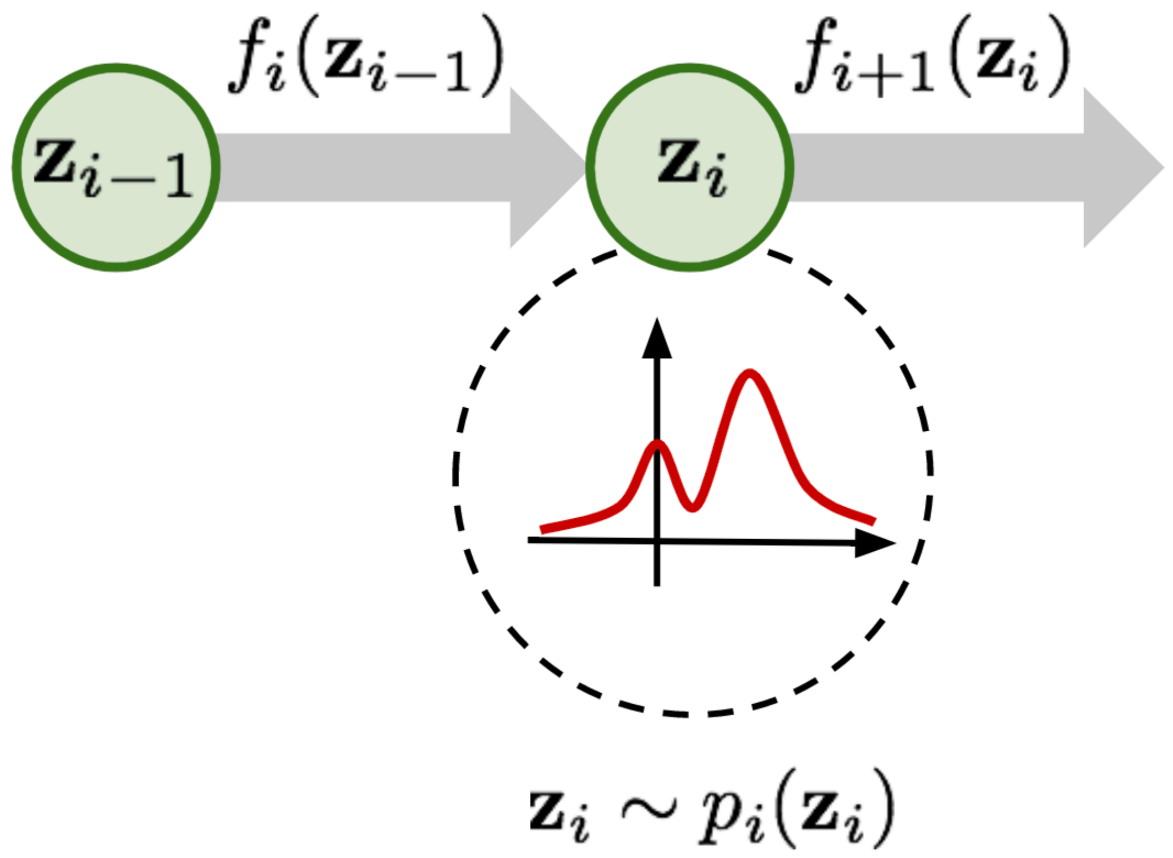
Step 2: Adapt to 3D (replace data, architecture etc.)

Slide Credit: Shubham Tulsiani

Generating 3D Shapes







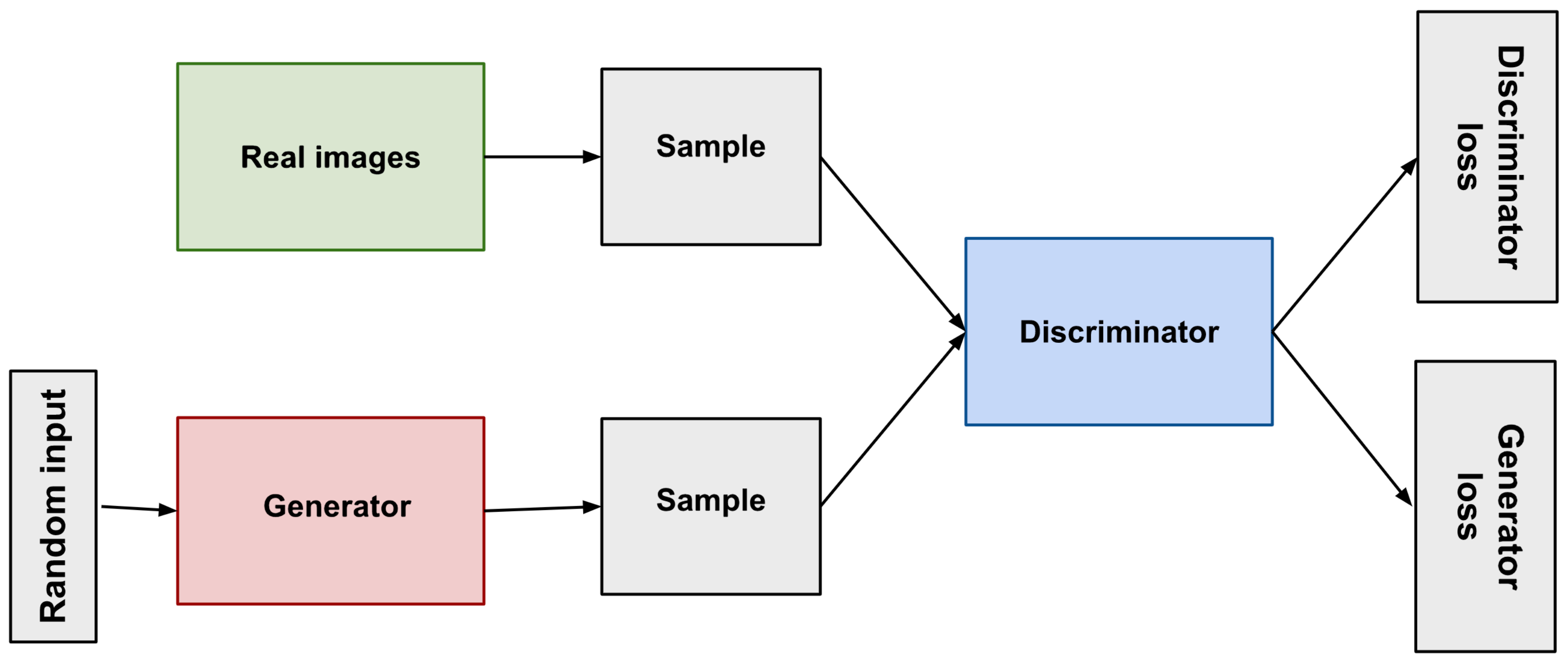
*3D was merely an **application** of generic methods (nothing in technique was 3D-specific)*

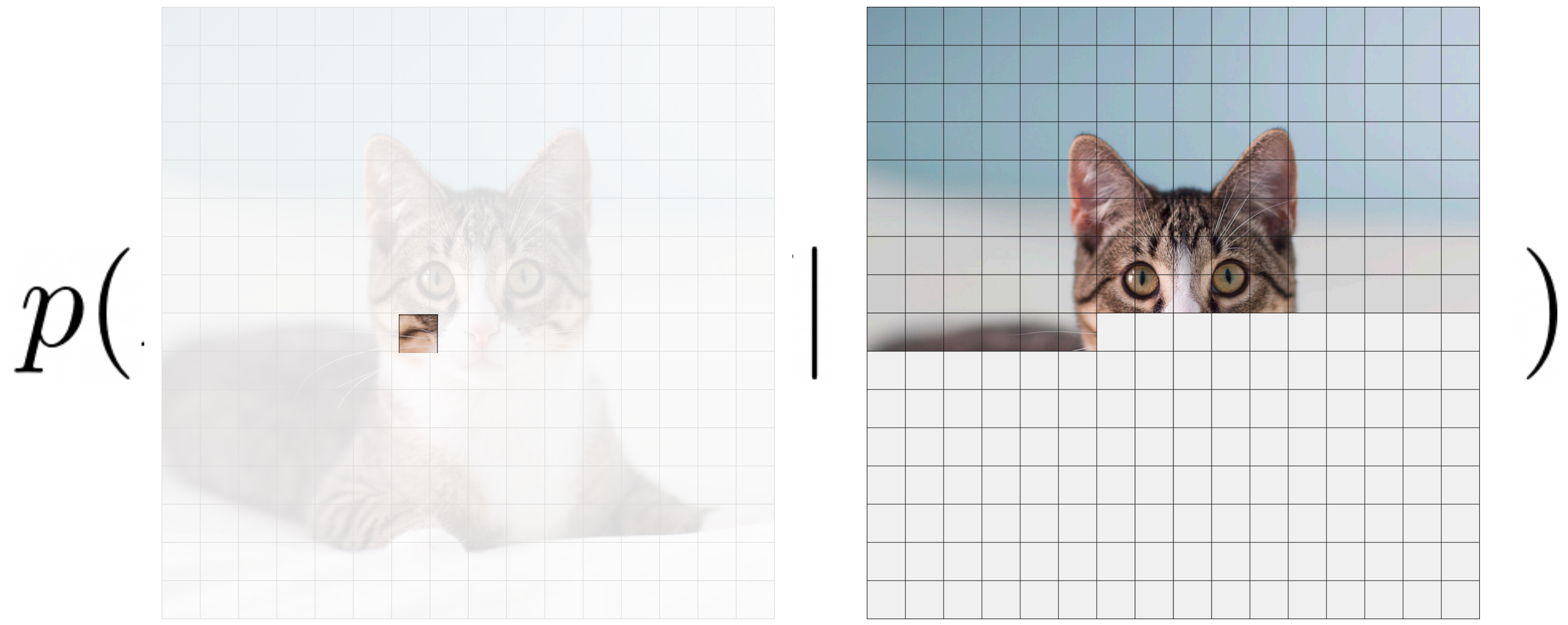
Step 1: Choose your favorite 2D generation model

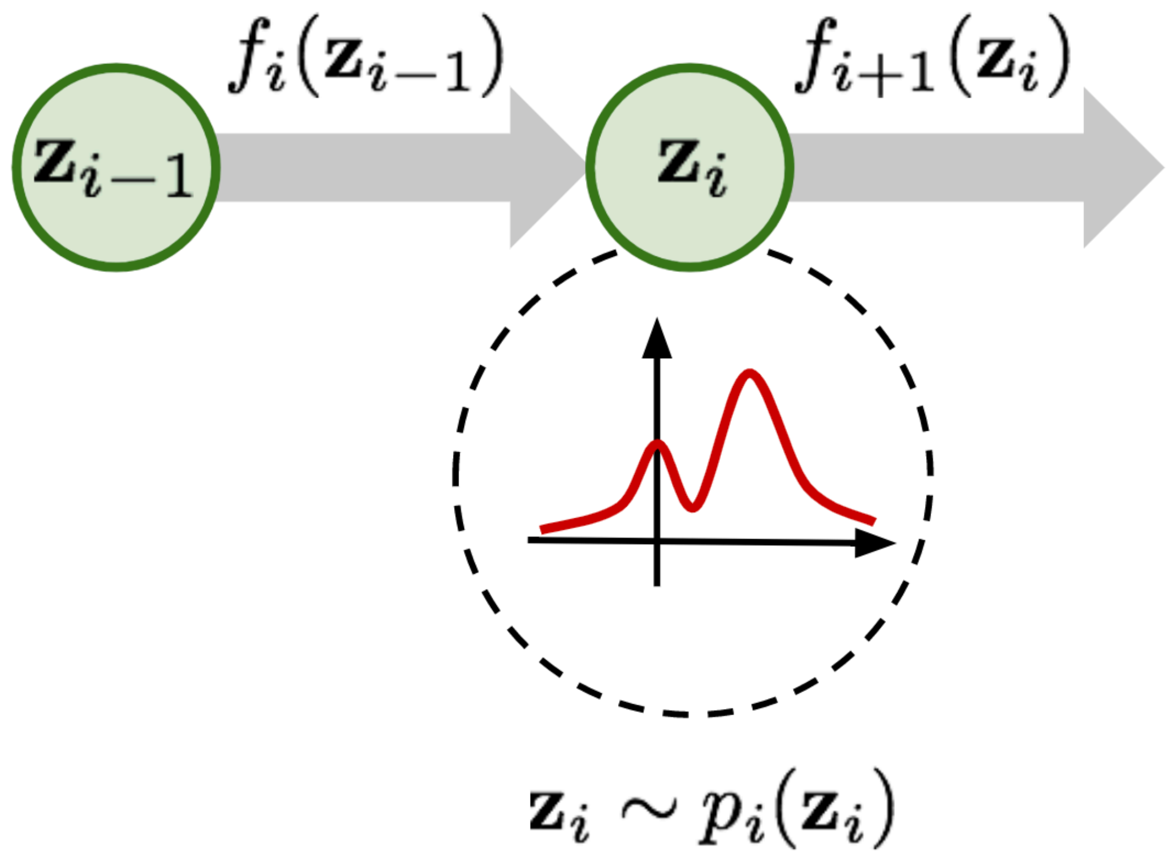
Step 2: Adapt to 3D (replace data, architecture etc.)

Slide Credit: Shubham Tulsiani

Generating 3D Shapes







*3D was merely an **application** of generic methods (nothing in technique was 3D-specific)*

Step 1: Choose your favorite 2D generation model

Step 2: Adapt to 3D (replace data, architecture etc.)

But what if we don't have 3D data for learning?

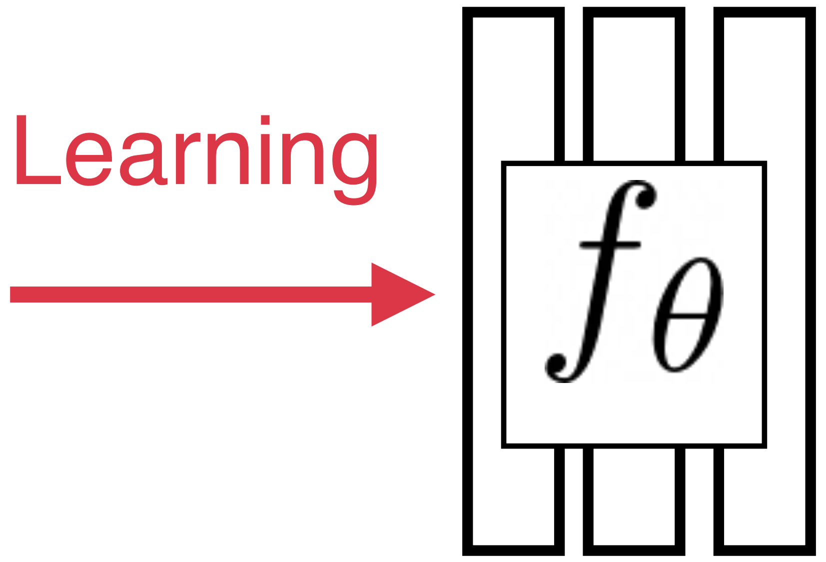
Slide Credit: Shubham Tulsiani

Generating 3D Shapes without 3D training data

Can we learn a generative 3D model from 2D images?

Slide Credit: Shubham Tulsiani

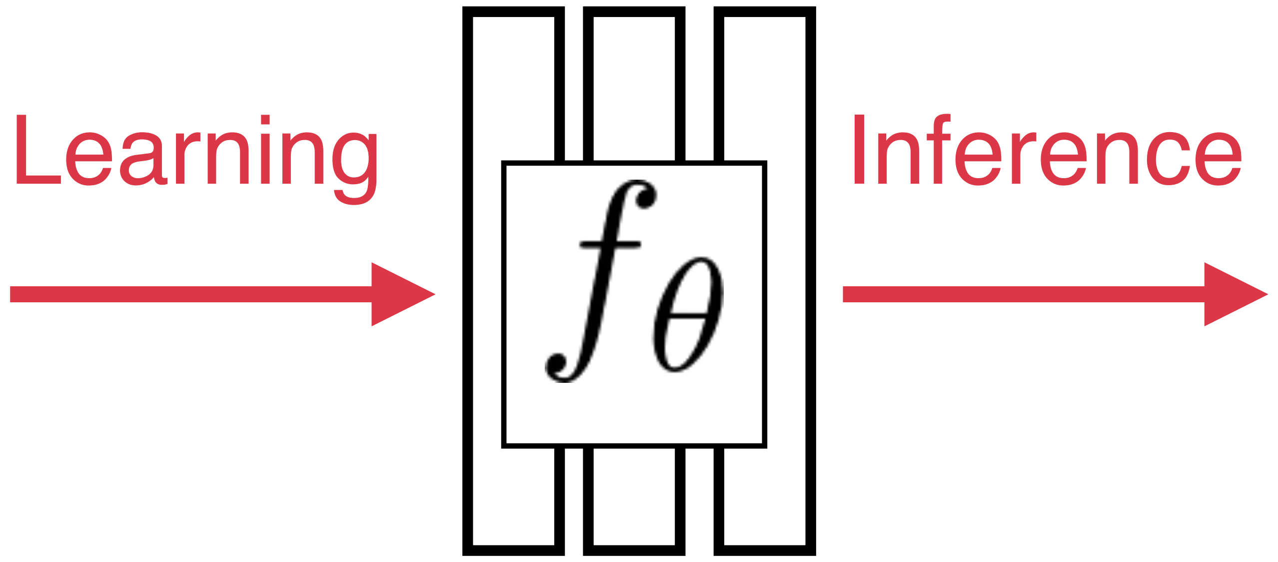
Generating 3D Shapes without 3D training data



Can we learn a generative 3D model from 2D images?

Slide Credit: Shubham Tulsiani

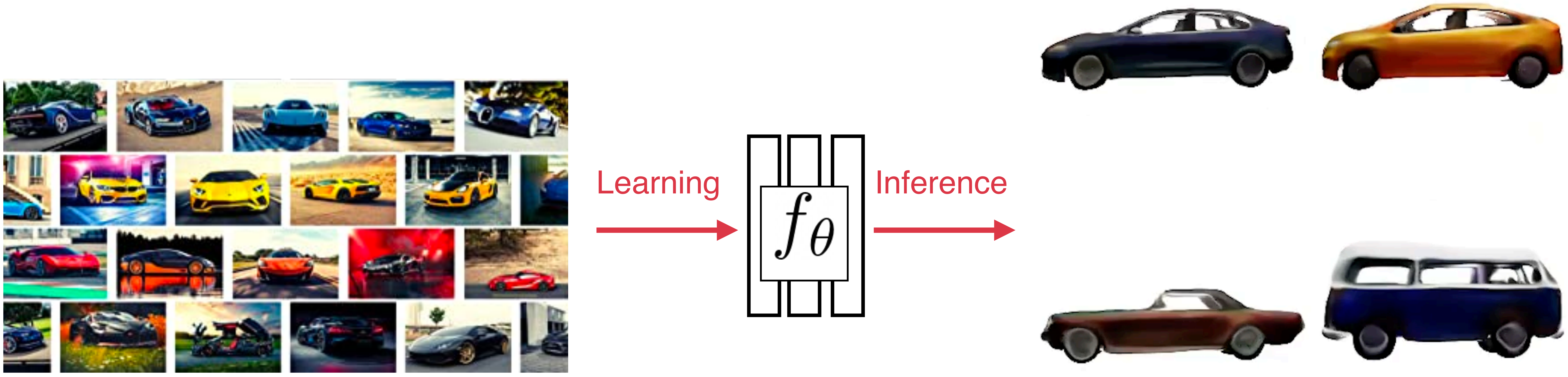
Generating 3D Shapes without 3D training data



Can we learn a generative 3D model from 2D images?

Slide Credit: Shubham Tulsiani

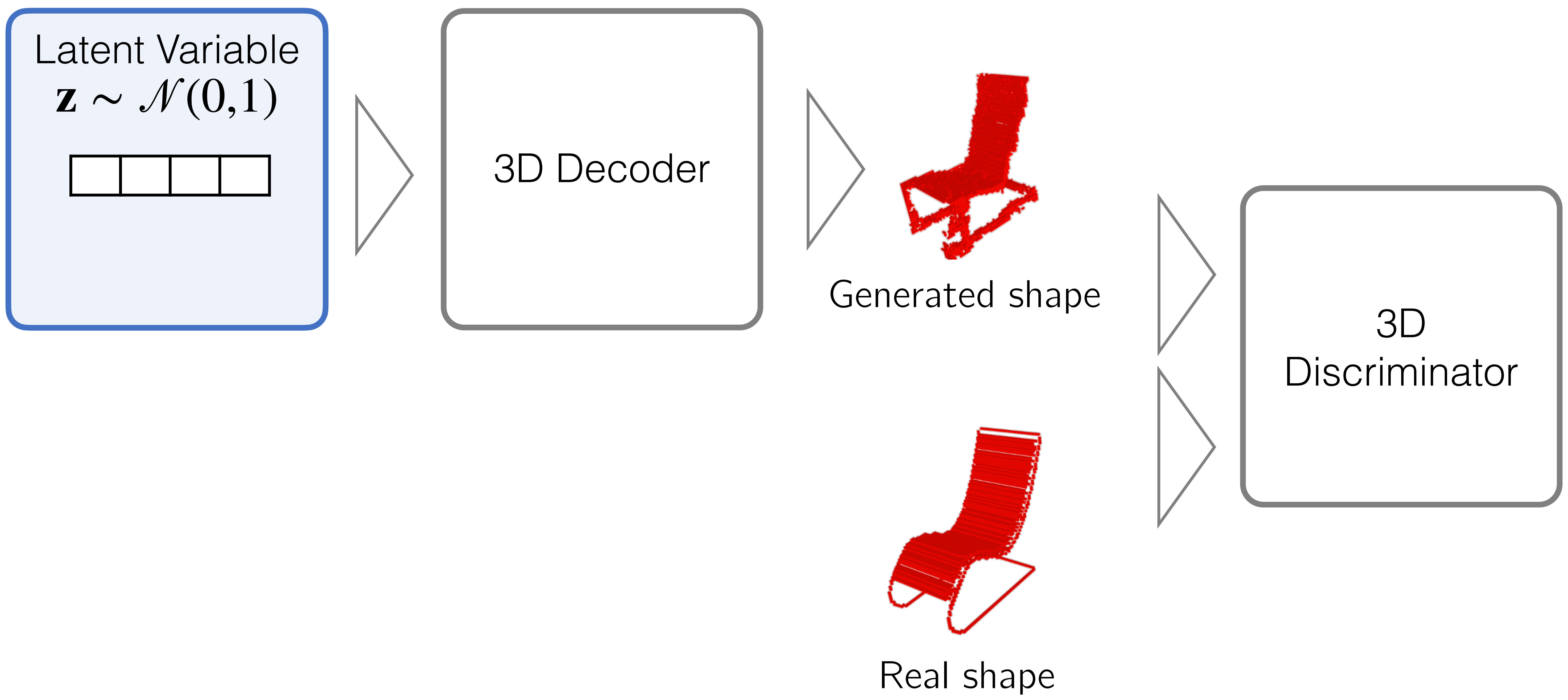
Generating 3D Shapes without 3D training data



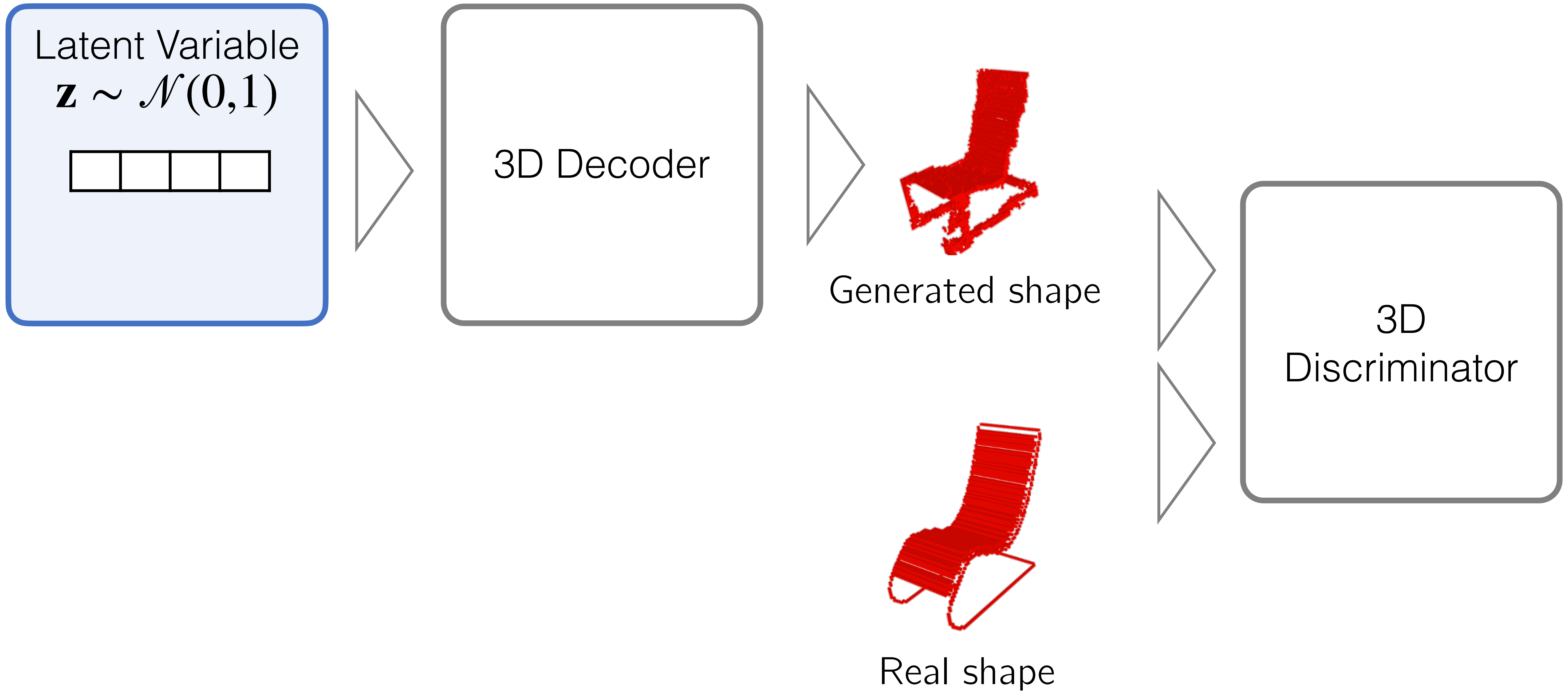
Can we learn a generative 3D model from 2D images?

Slide Credit: Shubham Tulsiani

Generating 3D Shapes without 3D training data

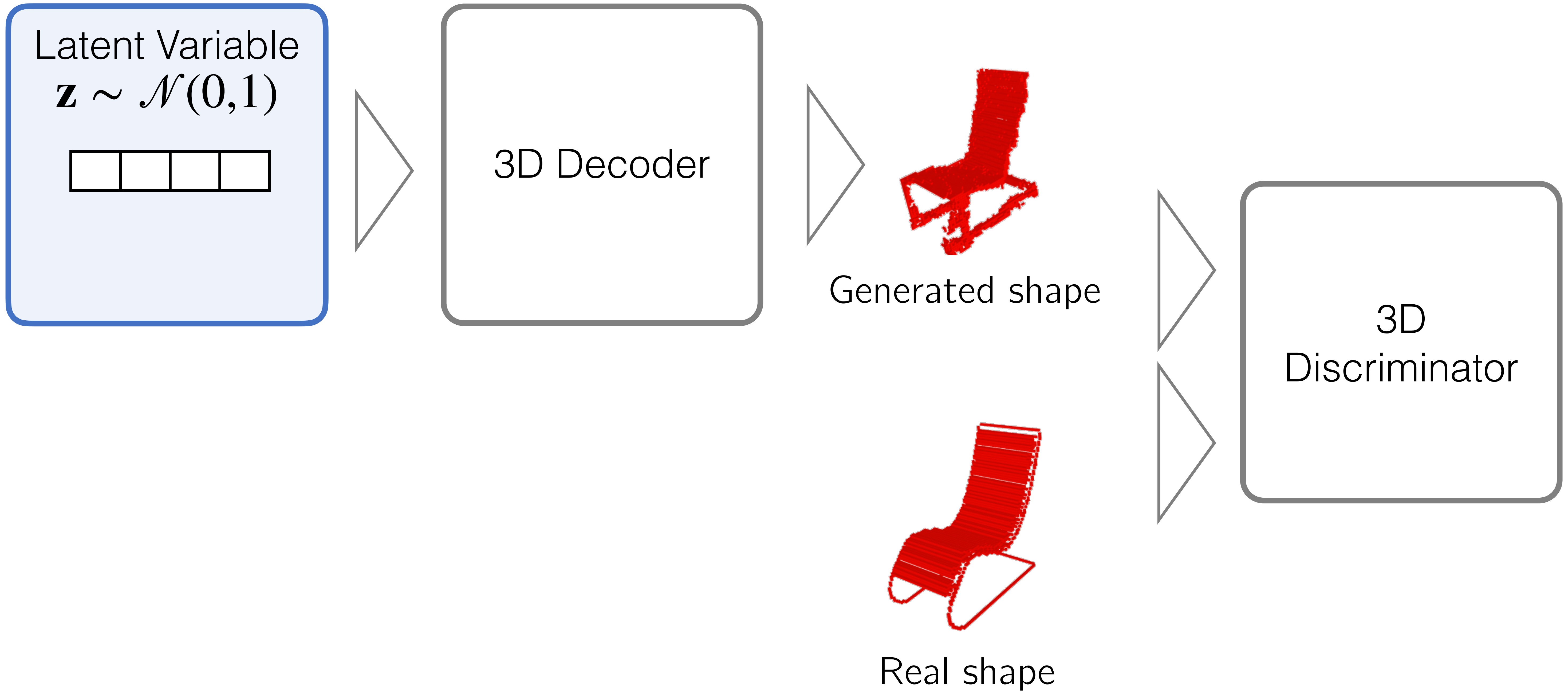


Generating 3D Shapes without 3D training data



Key Idea: Generate 3D representations such that —

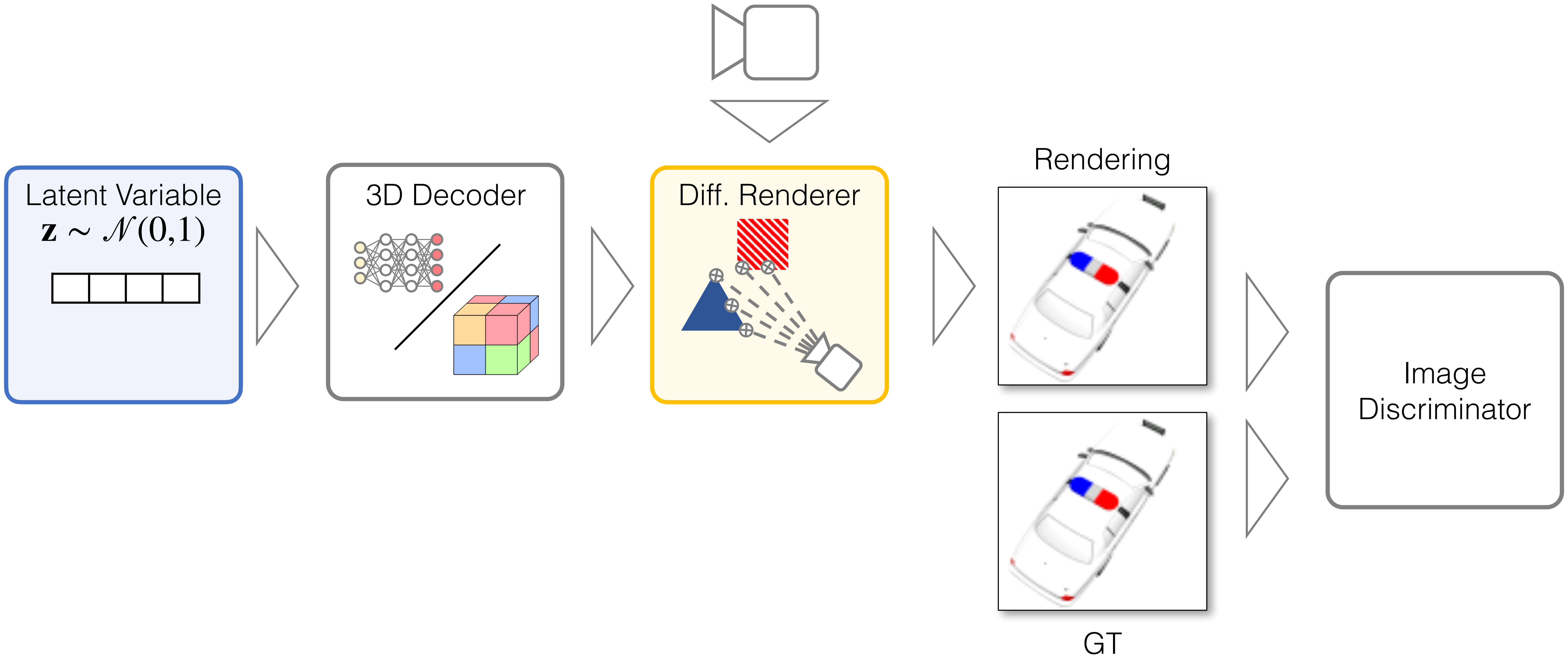
Generating 3D Shapes without 3D training data



Key Idea: Generate 3D representations such that —

they are indistinguishable from real samples

Generating 3D Shapes without 3D training data



Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

Generating 3D using 2D Adversarial Nets

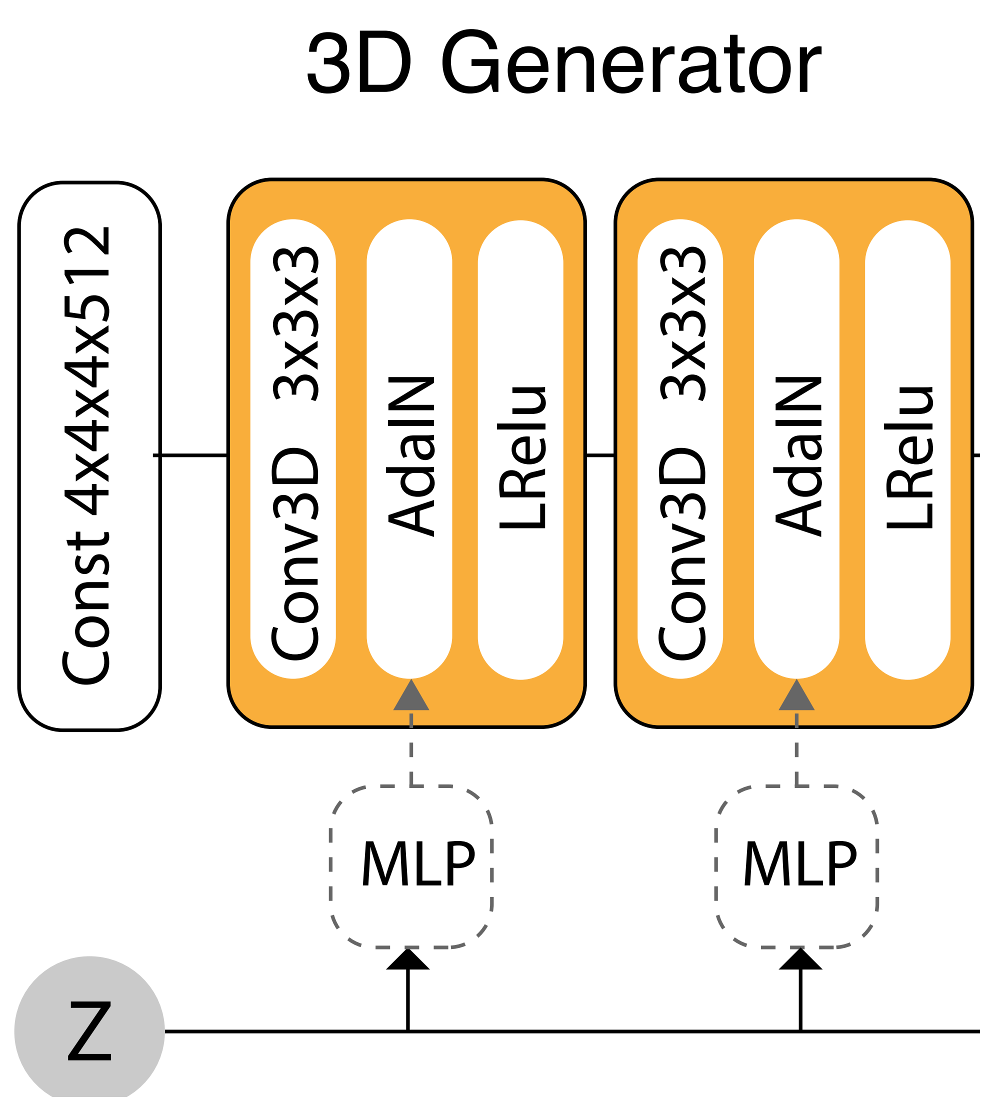
Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets



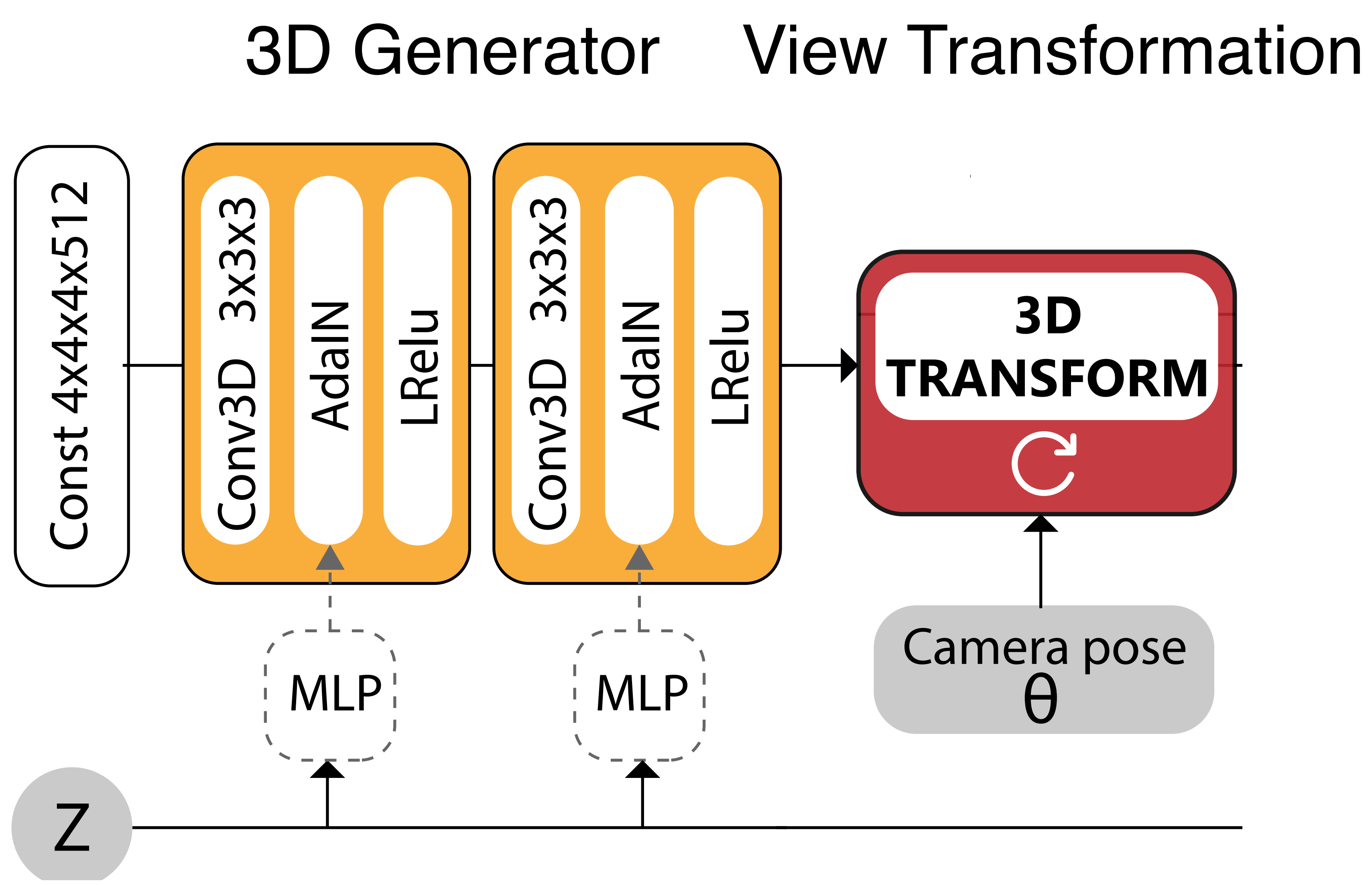
Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets



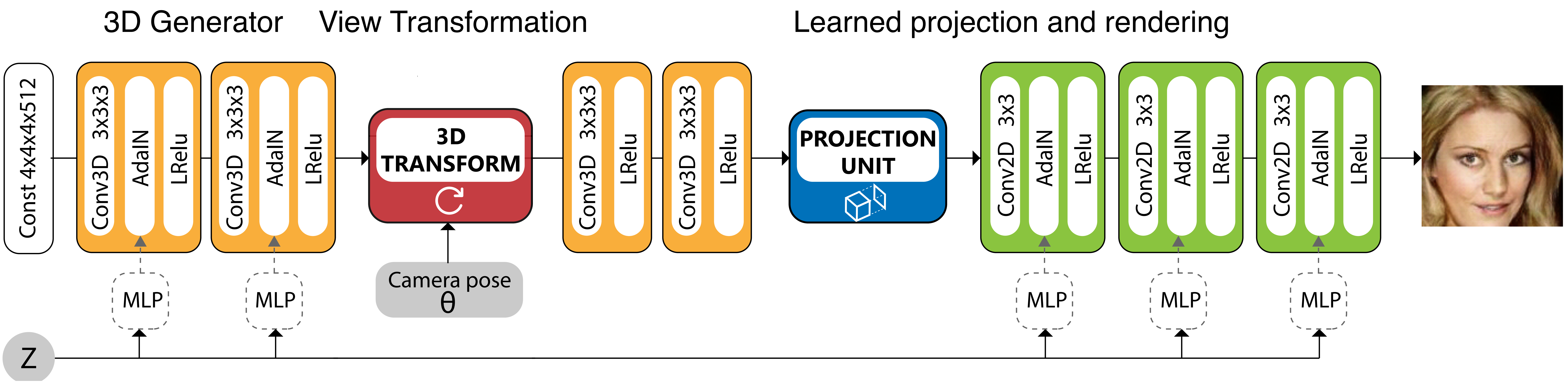
Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets



Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

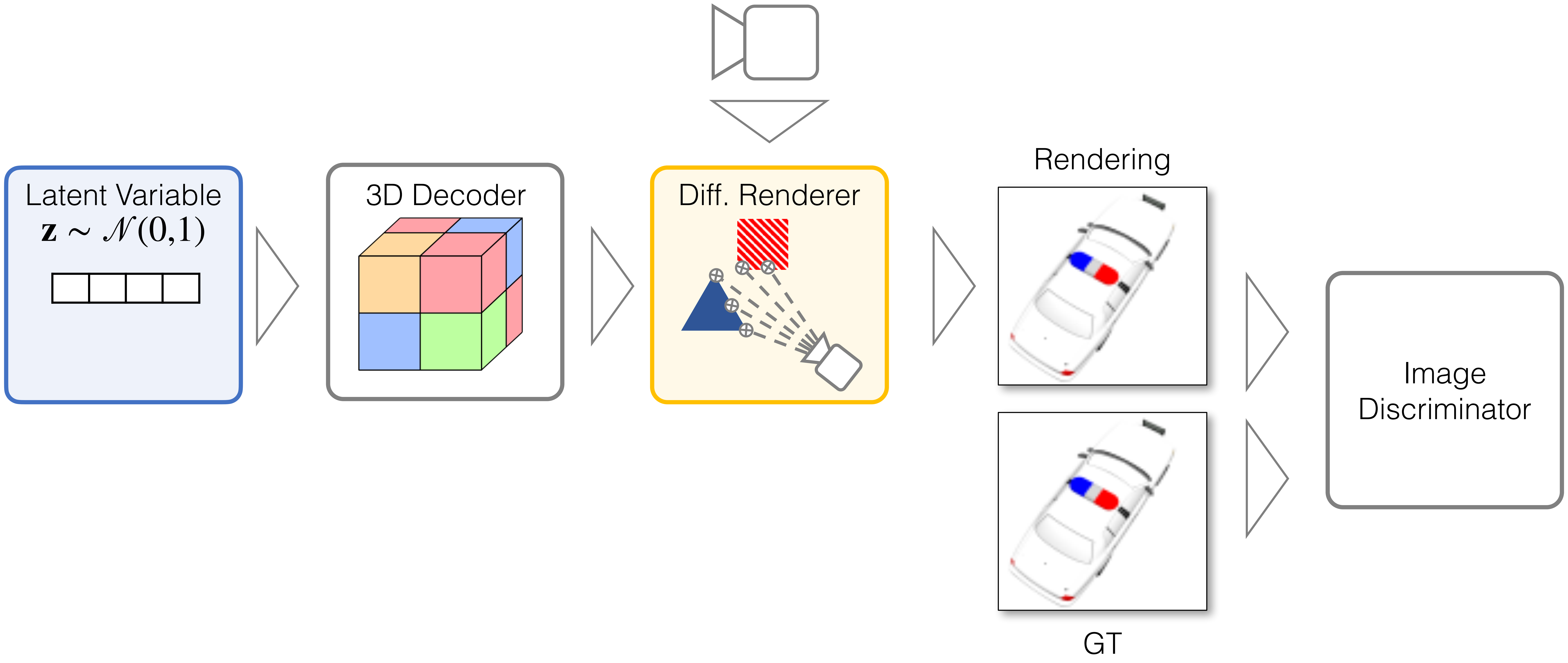
Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

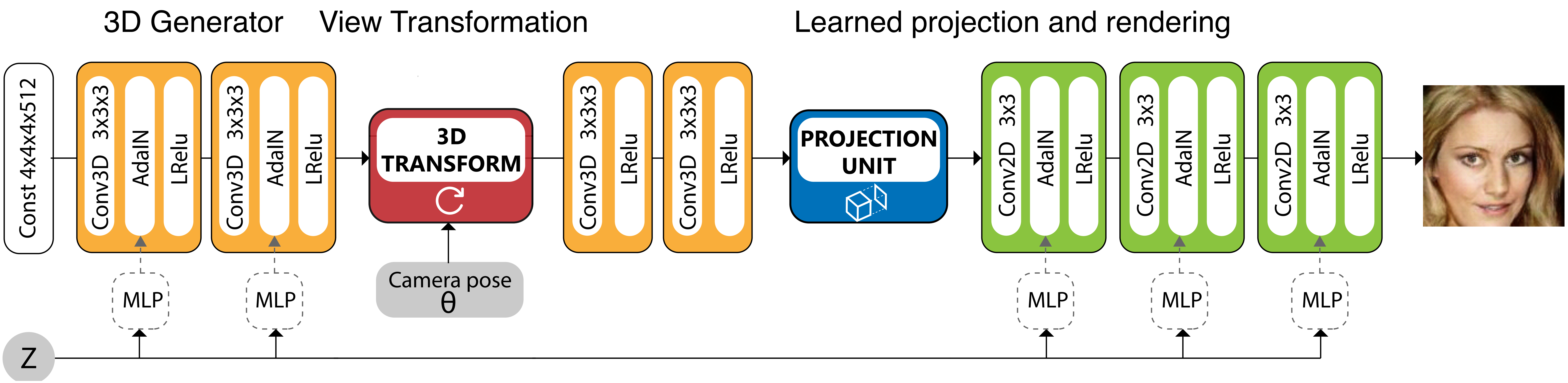
Generating 3D Shapes without 3D training data



Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

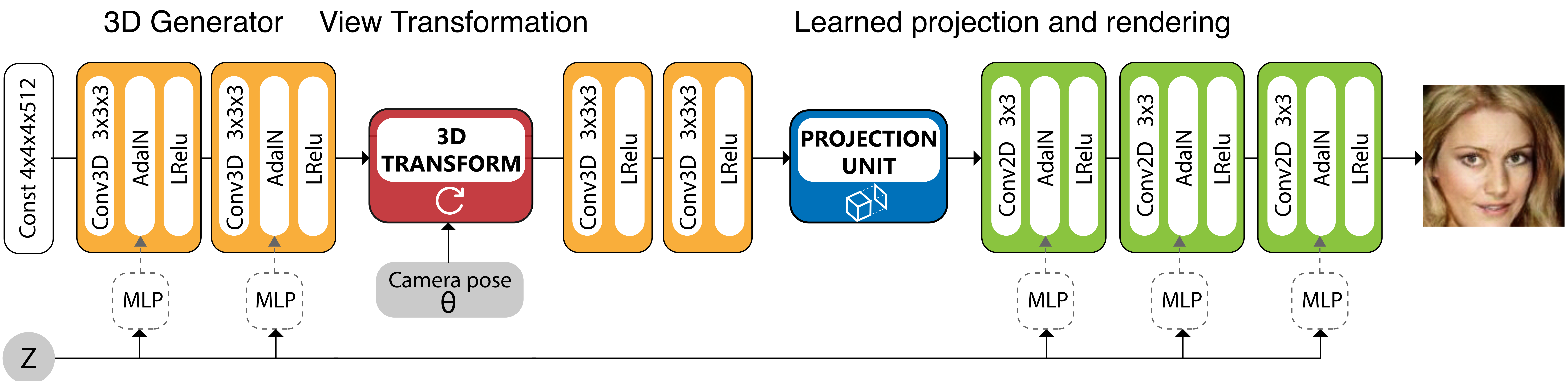
Generating 3D using 2D Adversarial Nets



HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

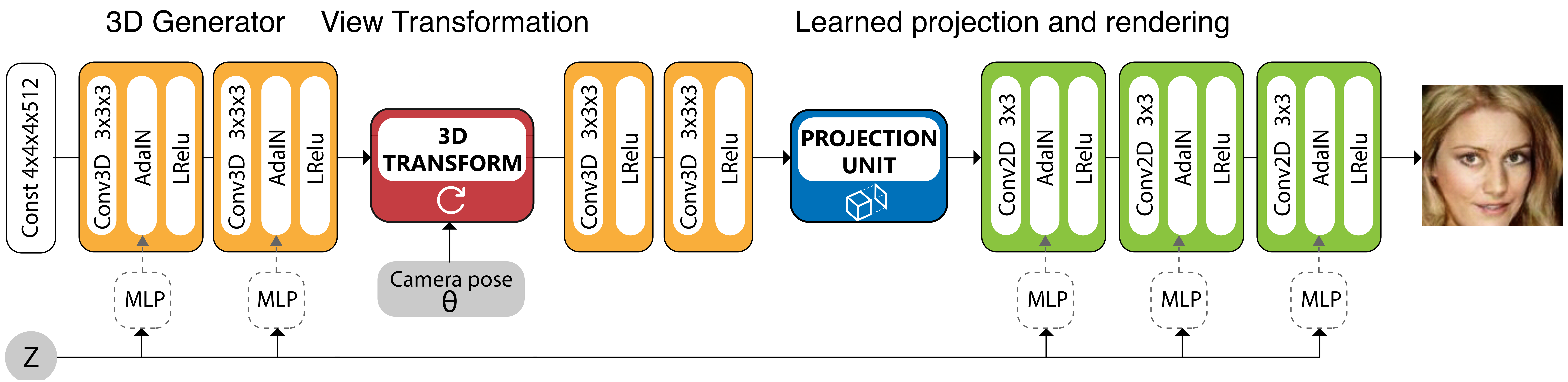


Some inconsistency across views

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets



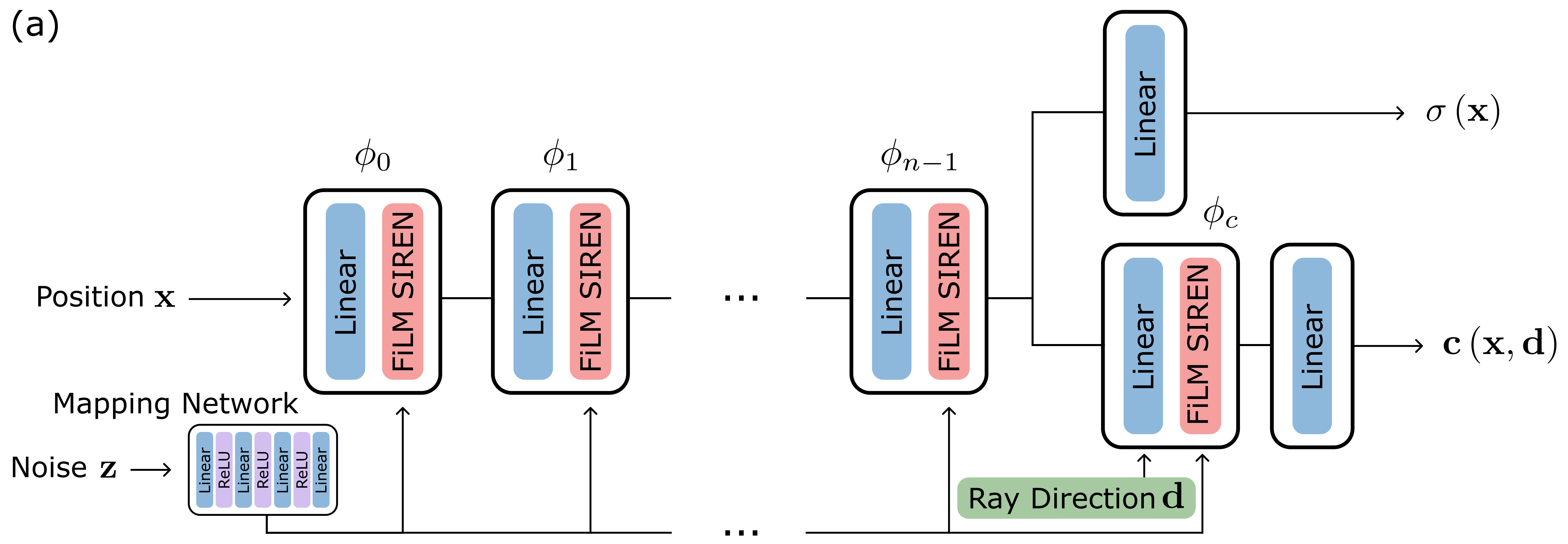
Some inconsistency across views

Some 3D inductive biases, but not enough (e.g. learned instead of volumetric rendering)

HoloGAN: Unsupervised learning of 3D representations from natural images. *Nguyen-Phuoc et. al.*

Slide Credit: Shubham Tulsiani

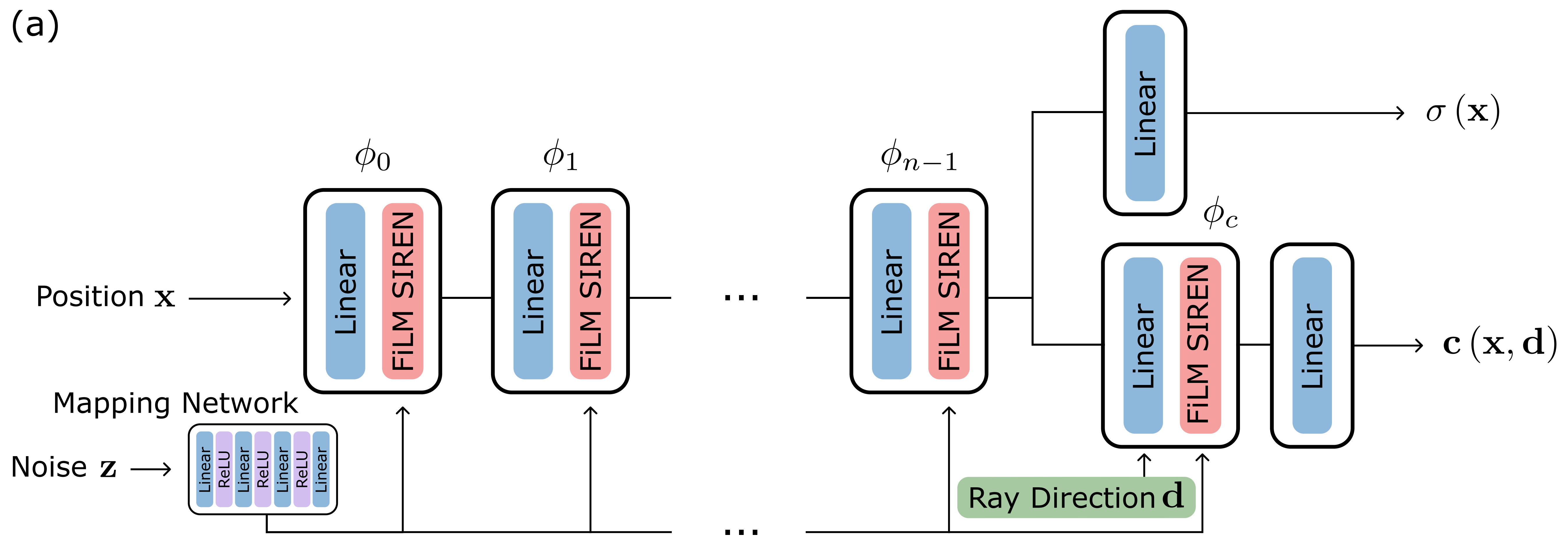
Generating 3D using 2D Adversarial Nets



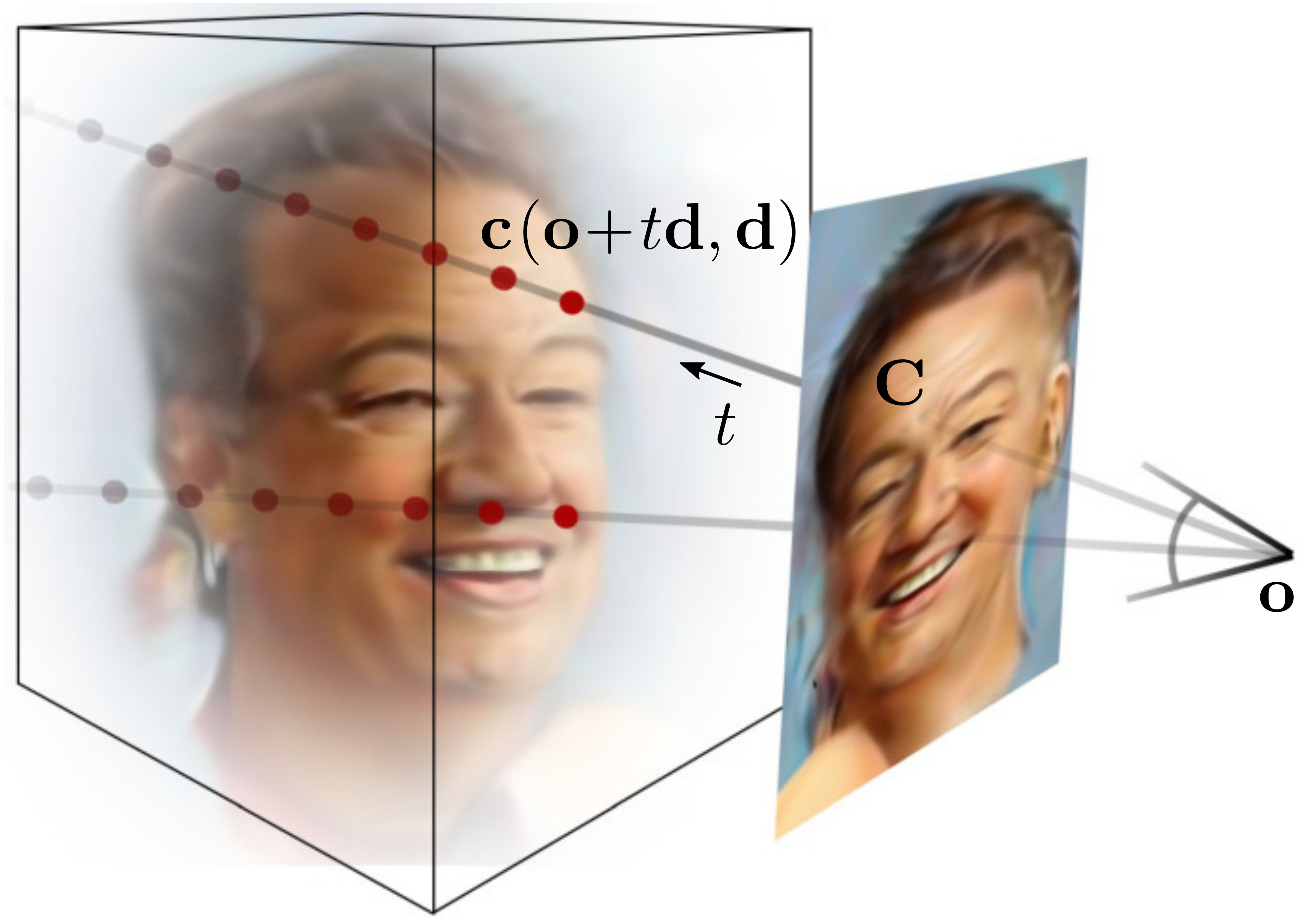
Generator: A conditional NeRF model

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et al.*

Generating 3D using 2D Adversarial Nets



Generator: A conditional NeRF model



Renderer: Volume rendering

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et. al.*

Generating 3D using 2D Adversarial Nets

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

Figure 5: Uncurated generated faces, corresponding to the first 30 random seeds.

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

Also possible to extract some geometry

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

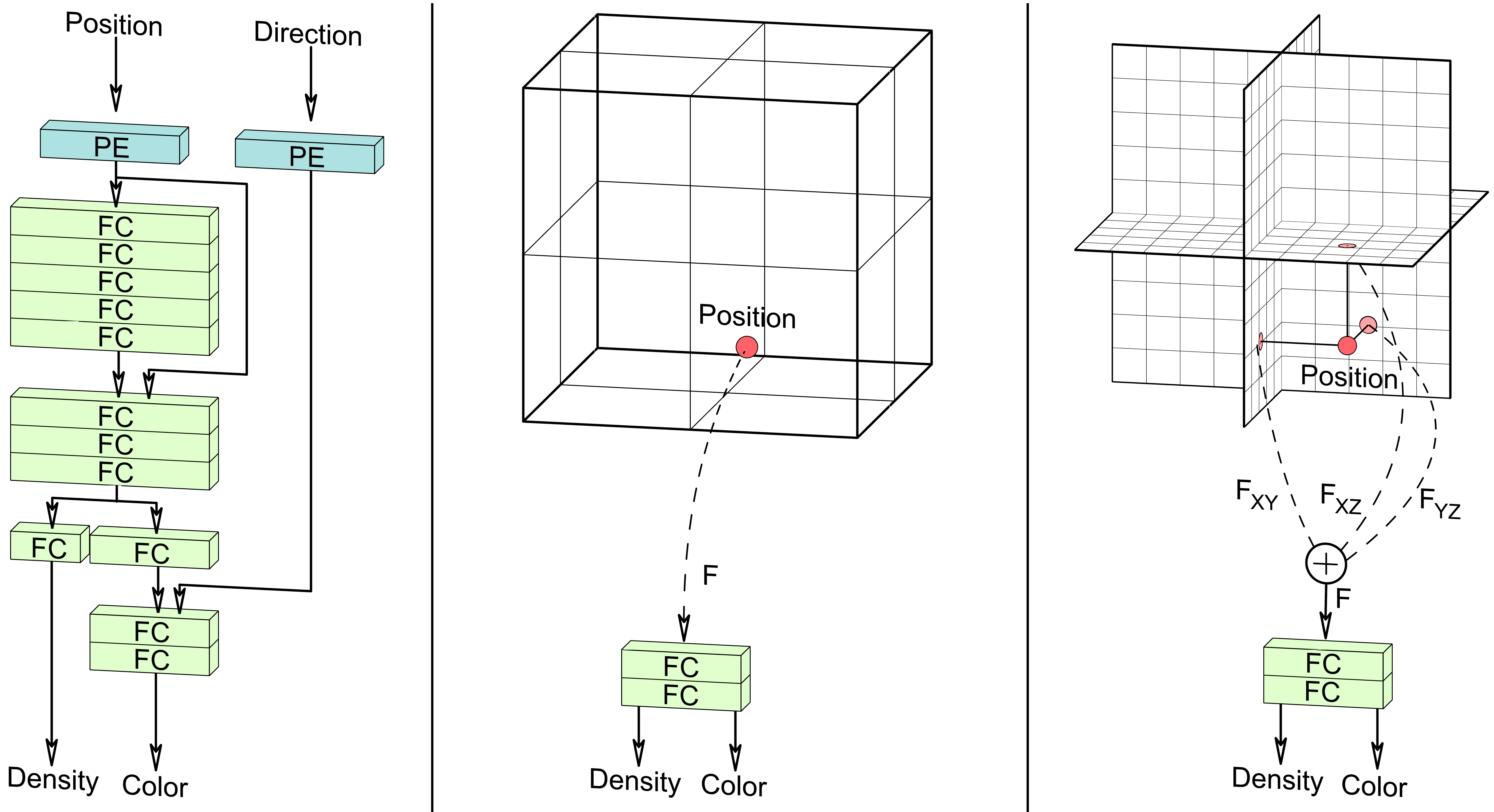
Also possible to extract some geometry

Figure 12: In a failure case reminiscent of the hollow-face illusion, our model sometimes generates objects with inverted sections.

Although not always great

pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis. *Chan et. al.*

Generating 3D using 2D Adversarial Nets



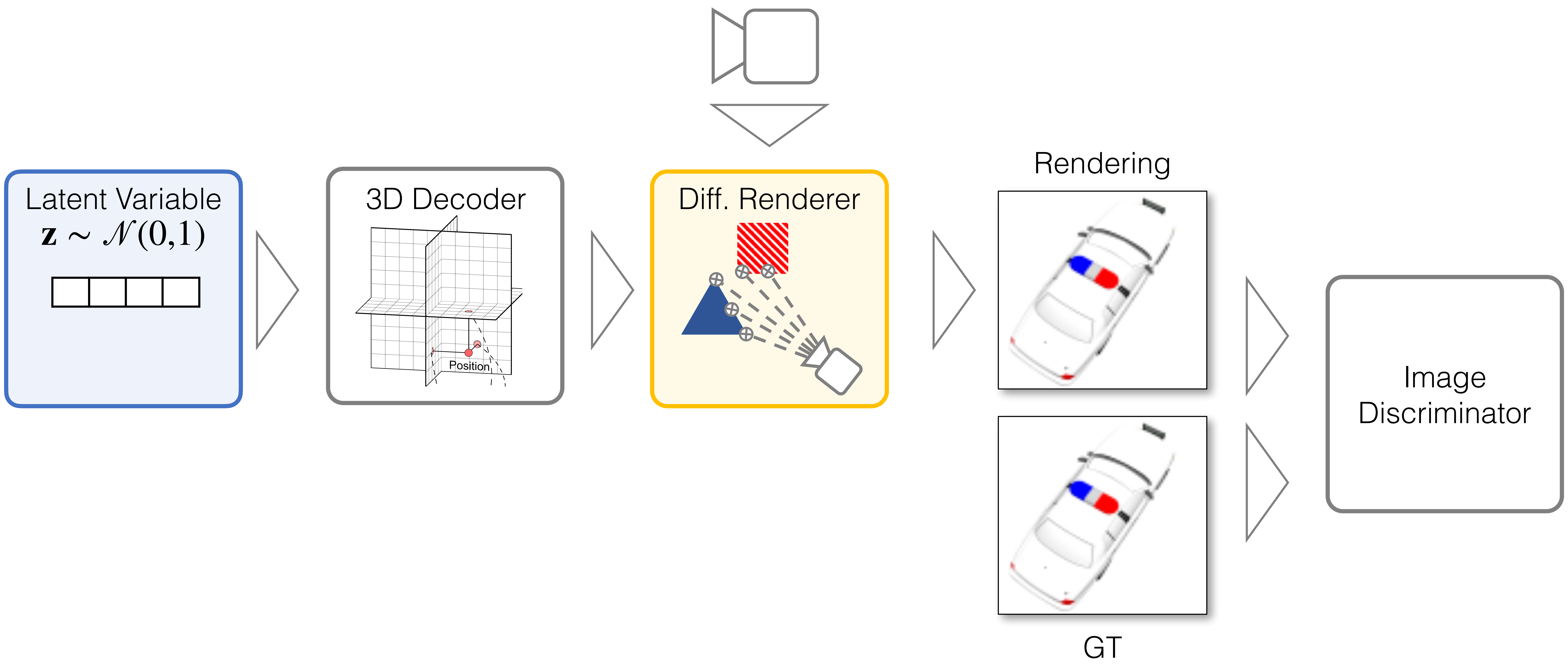
(a) NeRF (Implicit) (b) Voxels (Explicit or Hybrid) (c) Ours (Hybrid)

Can decode into
Voxelgrid,
Neural Field,
Triplane,
Groundplan,
...

EG3D: Efficient Geometry-aware 3D Generative Adversarial Networks. *Chan et. al.*

Slide Credit: Shubham Tulsiani

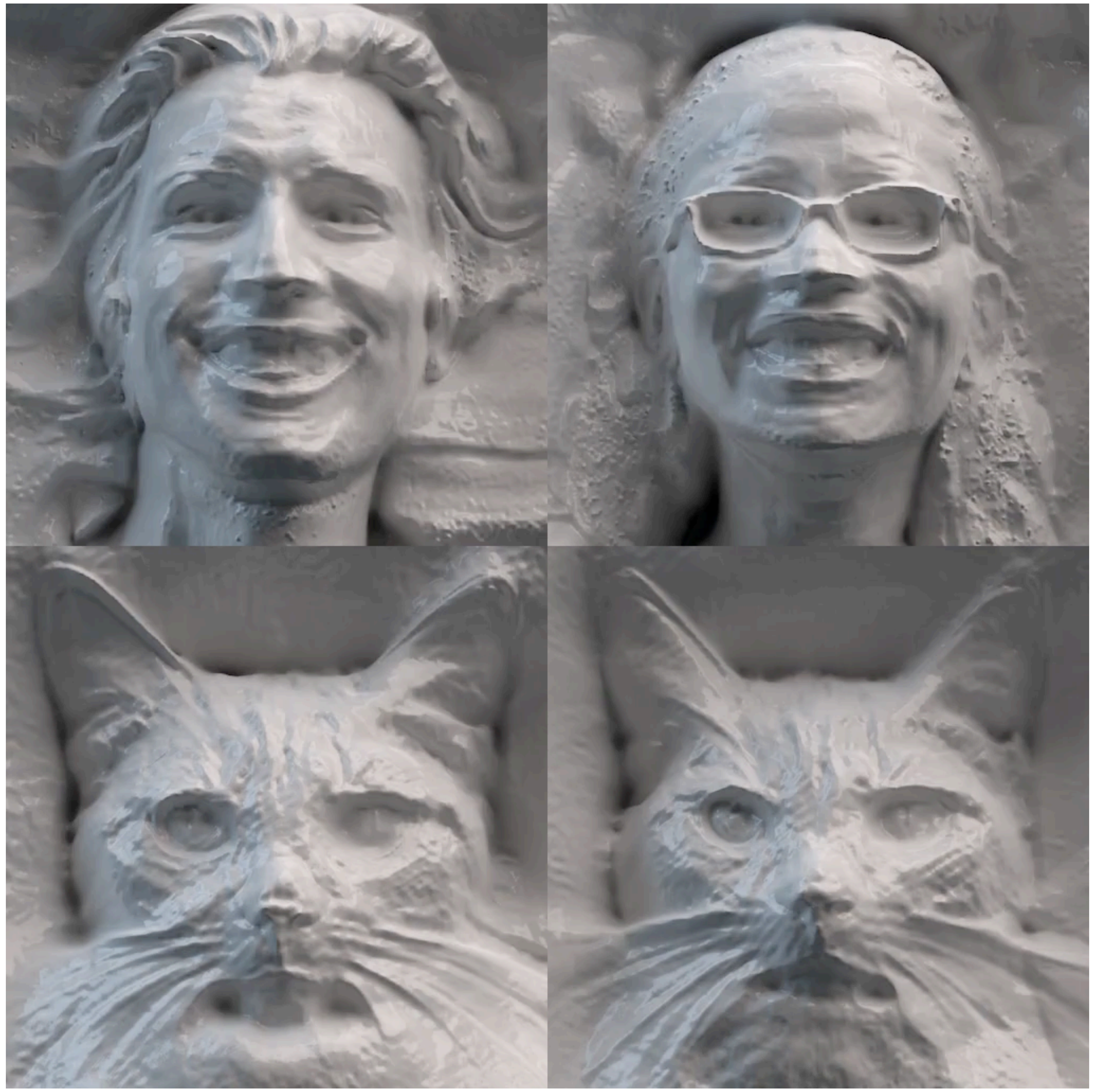
Generating 3D Shapes without 3D training data



Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

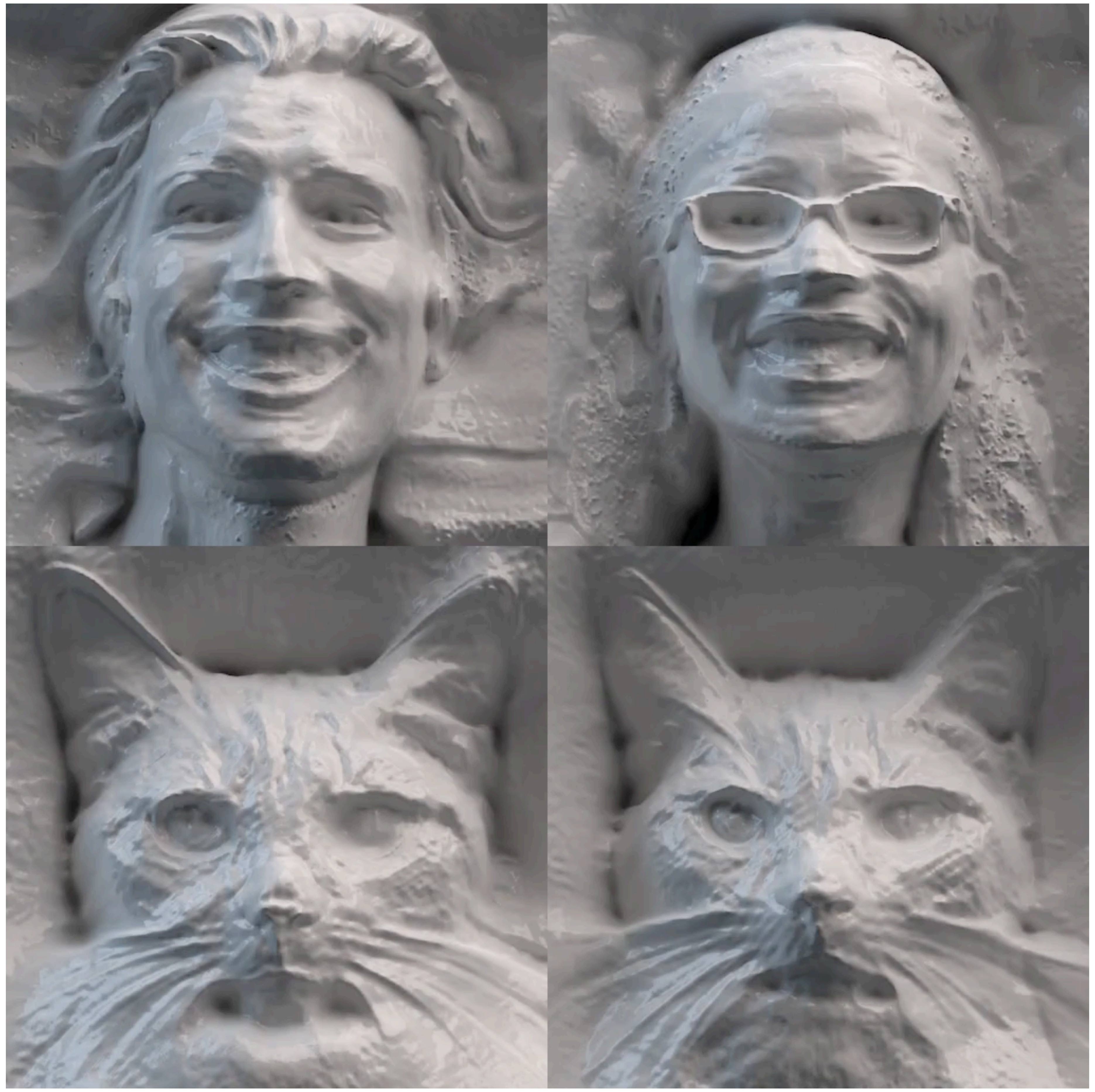
Generating 3D using 2D Adversarial Nets



EG3D: Efficient Geometry-aware 3D Generative Adversarial Networks. *Chan et. al.*

Slide Credit: Shubham Tulsiani

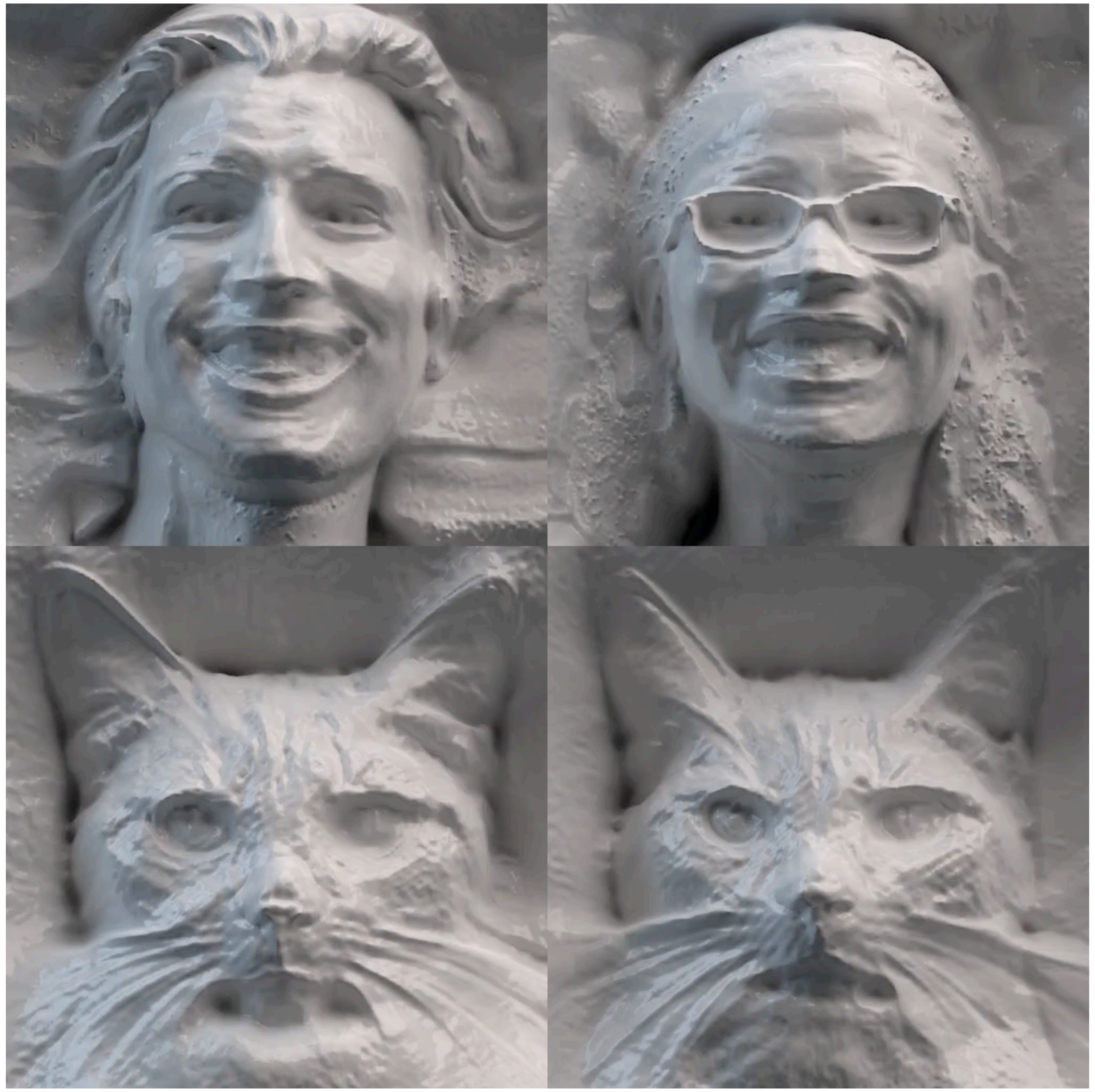
Generating 3D using 2D Adversarial Nets



EG3D: Efficient Geometry-aware 3D Generative Adversarial Networks. *Chan et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets



EG3D: Efficient Geometry-aware 3D Generative Adversarial Networks. *Chan et. al.*

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

Unconstrained Scene Generation with Locally Conditioned Radiance Fields, ICCV 2021, DeVries et al. 2022

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

Unconstrained Scene Generation with Locally Conditioned Radiance Fields, ICCV 2021, DeVries et al. 2022

Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

Locally or globally conditioned?

Unconstrained Scene Generation with Locally Conditioned Radiance Fields, ICCV 2021, DeVries et al. 2022

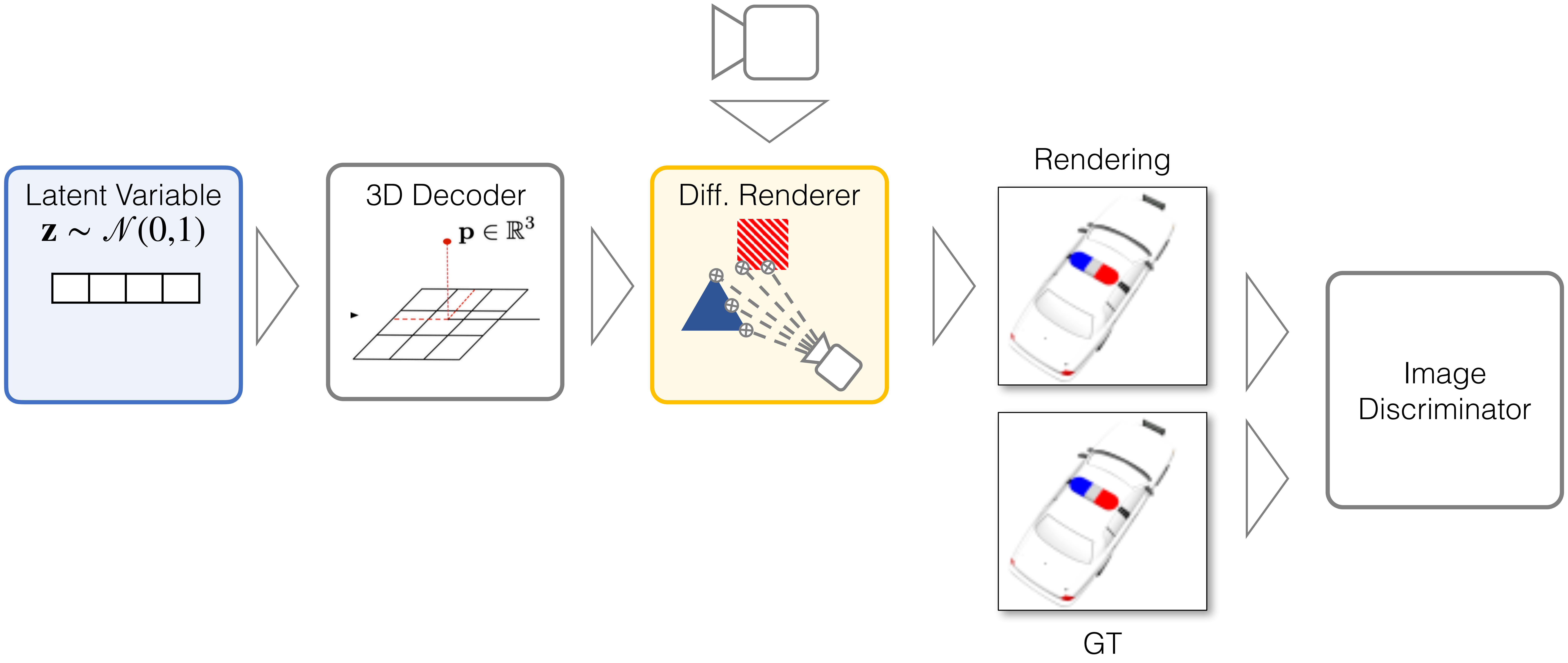
Slide Credit: Shubham Tulsiani

Generating 3D using 2D Adversarial Nets

Unconstrained Scene Generation with Locally Conditioned Radiance Fields, ICCV 2021, DeVries et al. 2022

Slide Credit: Shubham Tulsiani

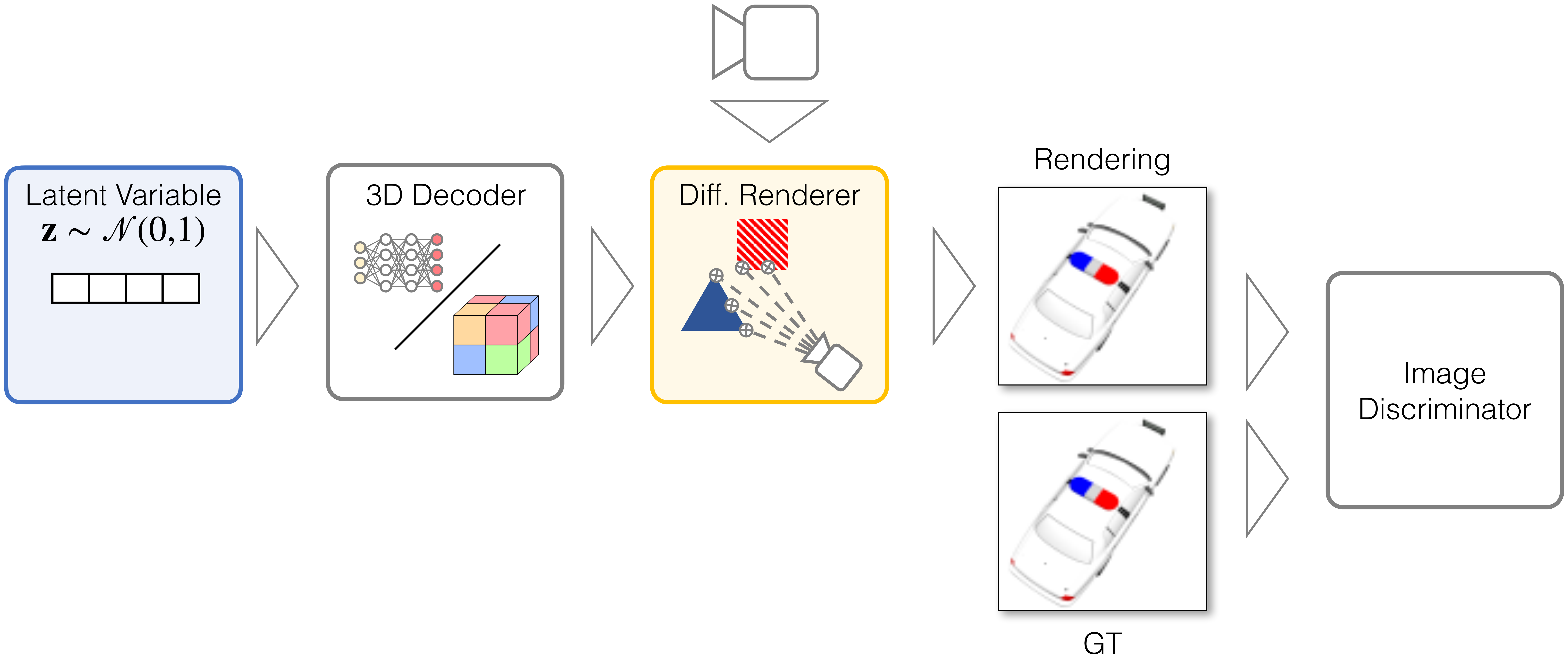
Generating 3D Shapes without 3D training data



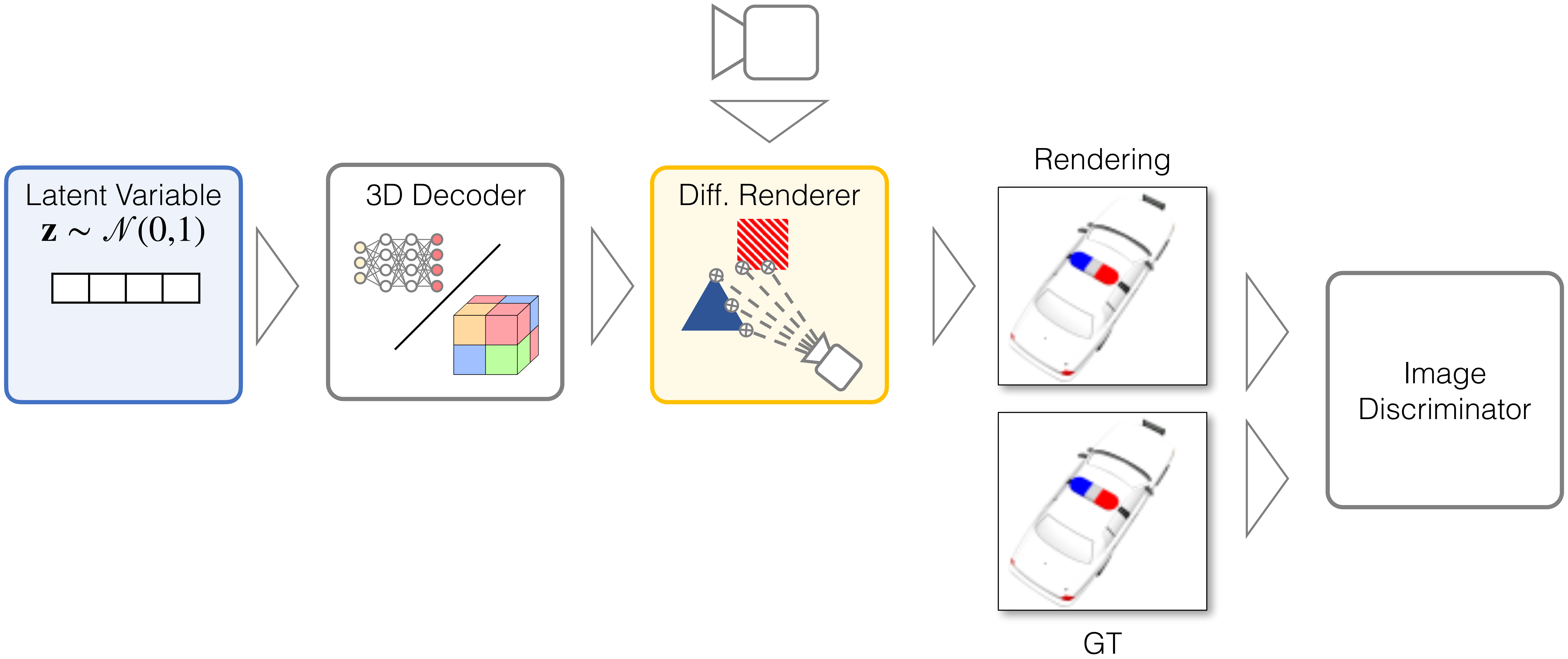
Key Idea: Generate 3D representations such that —

their renderings are indistinguishable from real samples

Generating 3D Shapes without 3D training data

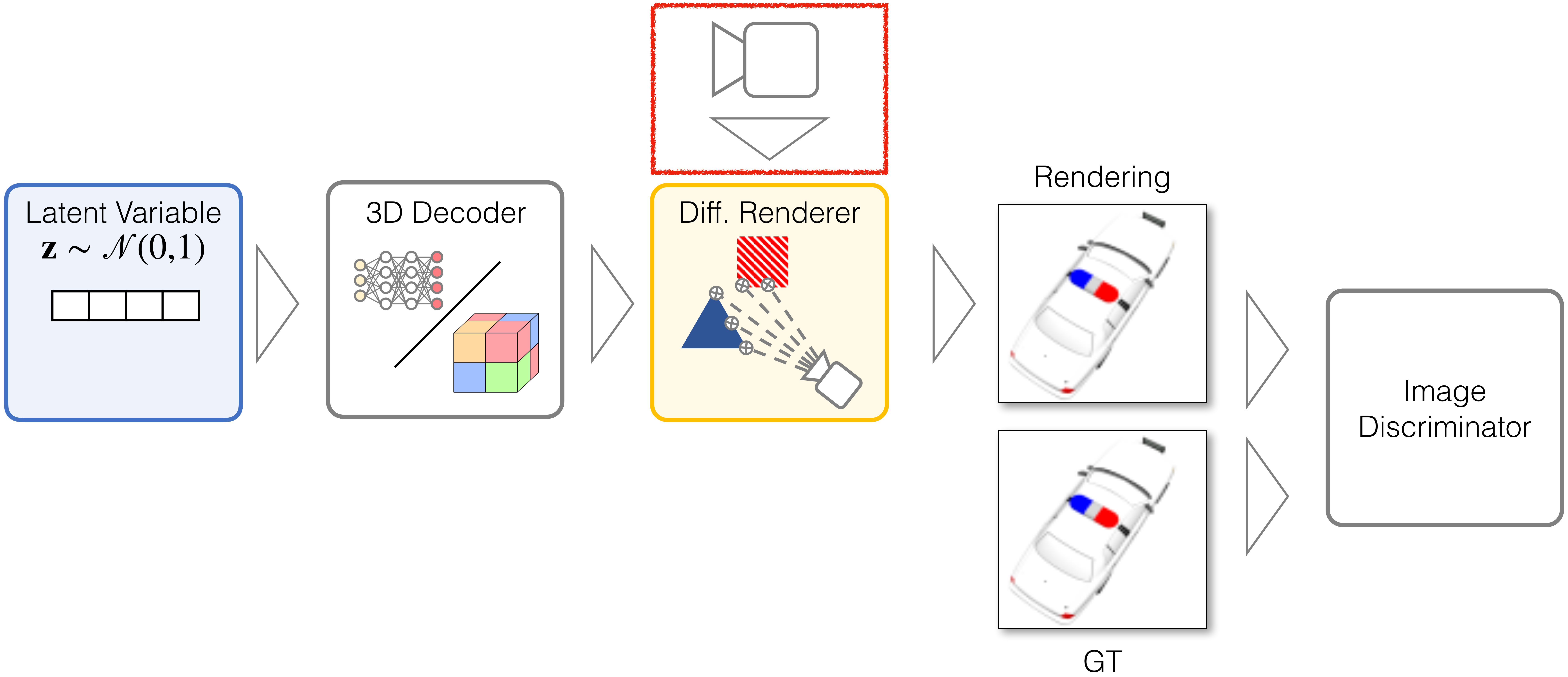


Generating 3D Shapes without 3D training data



Crucial to leverage the 3D structure of the problem

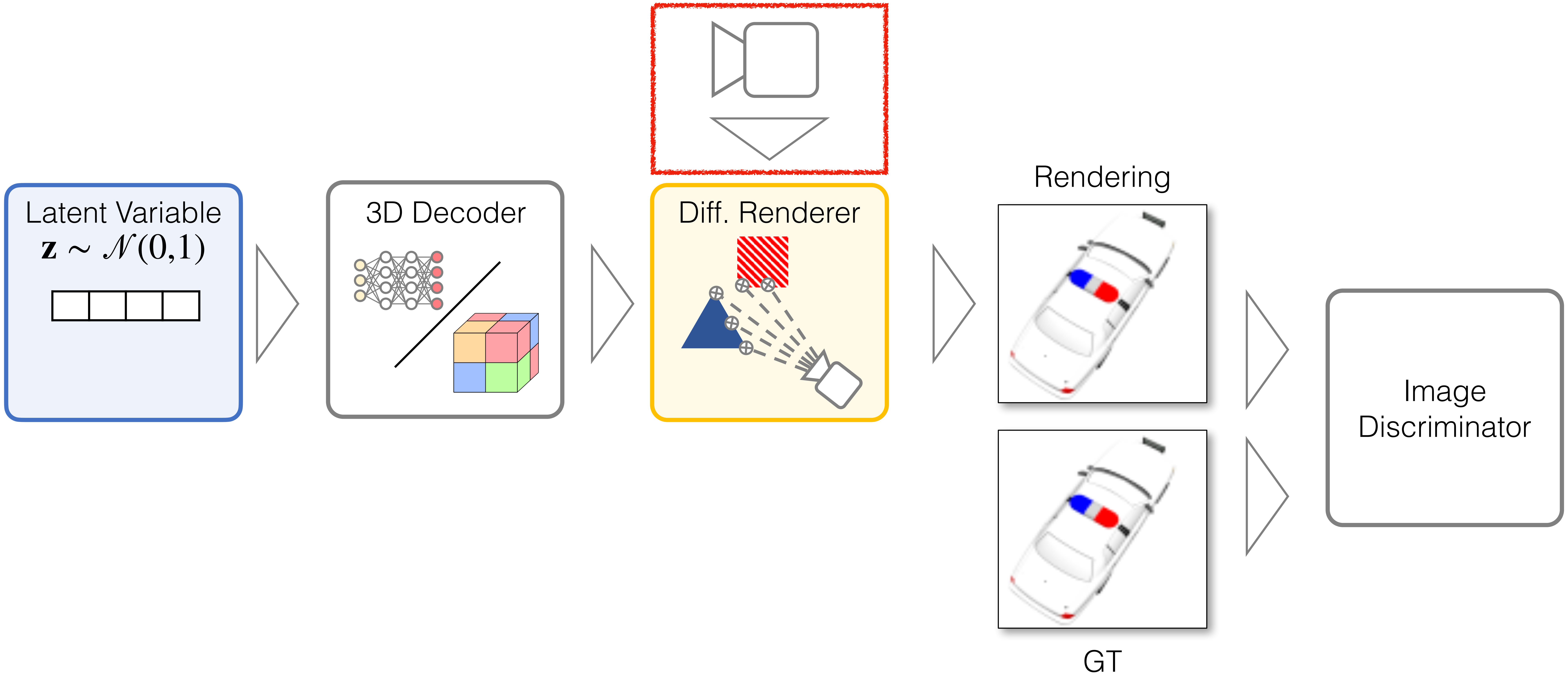
Generating 3D Shapes without 3D training data



Crucial to leverage the 3D structure of the problem

Assumed known view distribution e.g. faces seen frontally

Generating 3D Shapes without 3D training data



Crucial to leverage the 3D structure of the problem

Assumed known view distribution e.g. faces seen frontally

Typical success stories on ‘nice’ categories (lots of clean images, known views…)

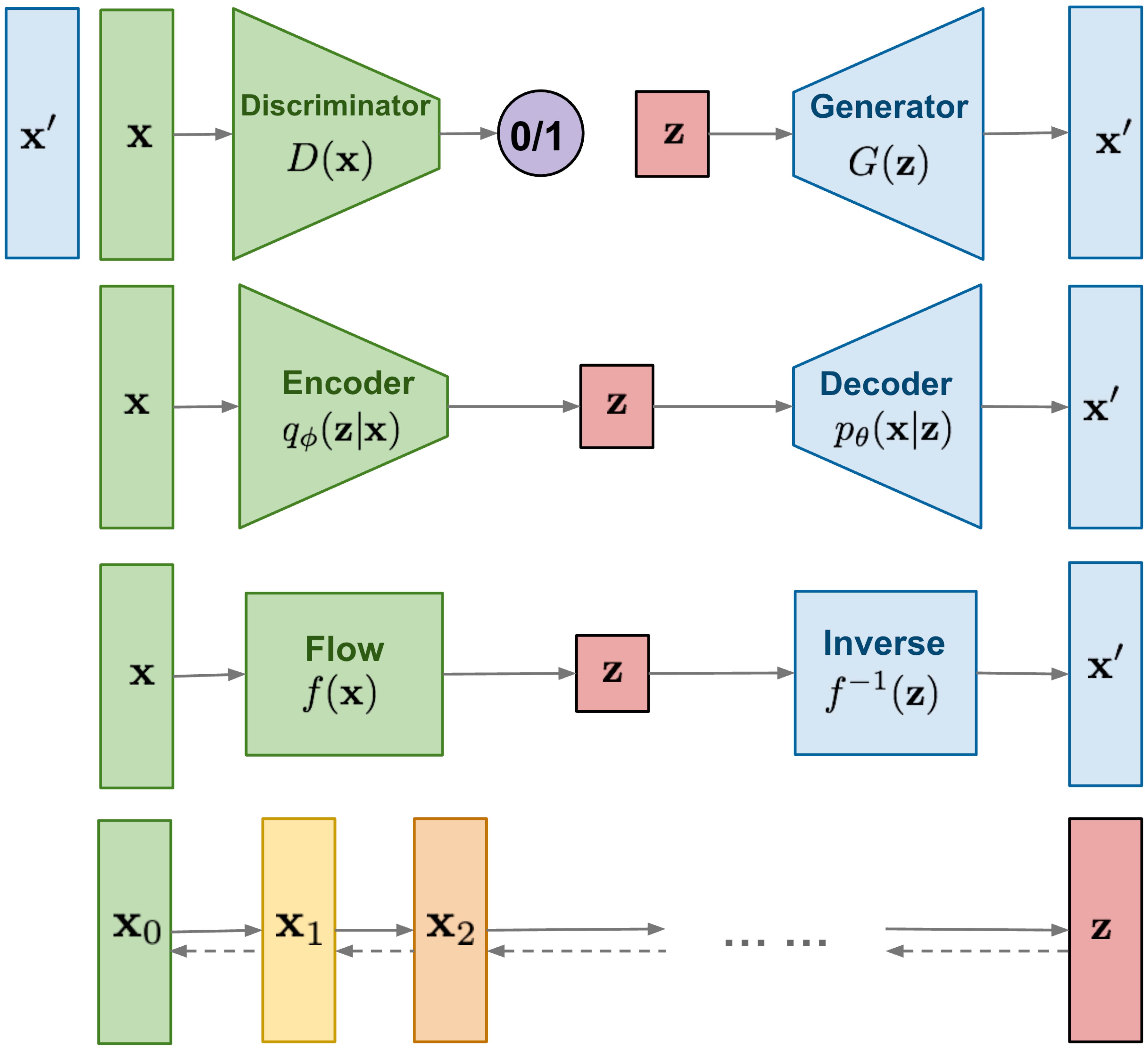
Background: Diffusion Models

GAN: Adversarial training

VAE: maximize variational lower bound

Flow-based models: Invertible transform of distributions

Diffusion models: Gradually add Gaussian noise and then reverse



Background: Diffusion Models

Forward SDE (data → noise)

$$\mathbf{x}(0) \longrightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \longrightarrow \mathbf{x}(T)$$

score function

$$\mathbf{x}(0) \longleftarrow \mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \longleftarrow \mathbf{x}(T)$$

Reverse SDE (noise → data)

Background: Diffusion Models

Forward SDE (data → noise)

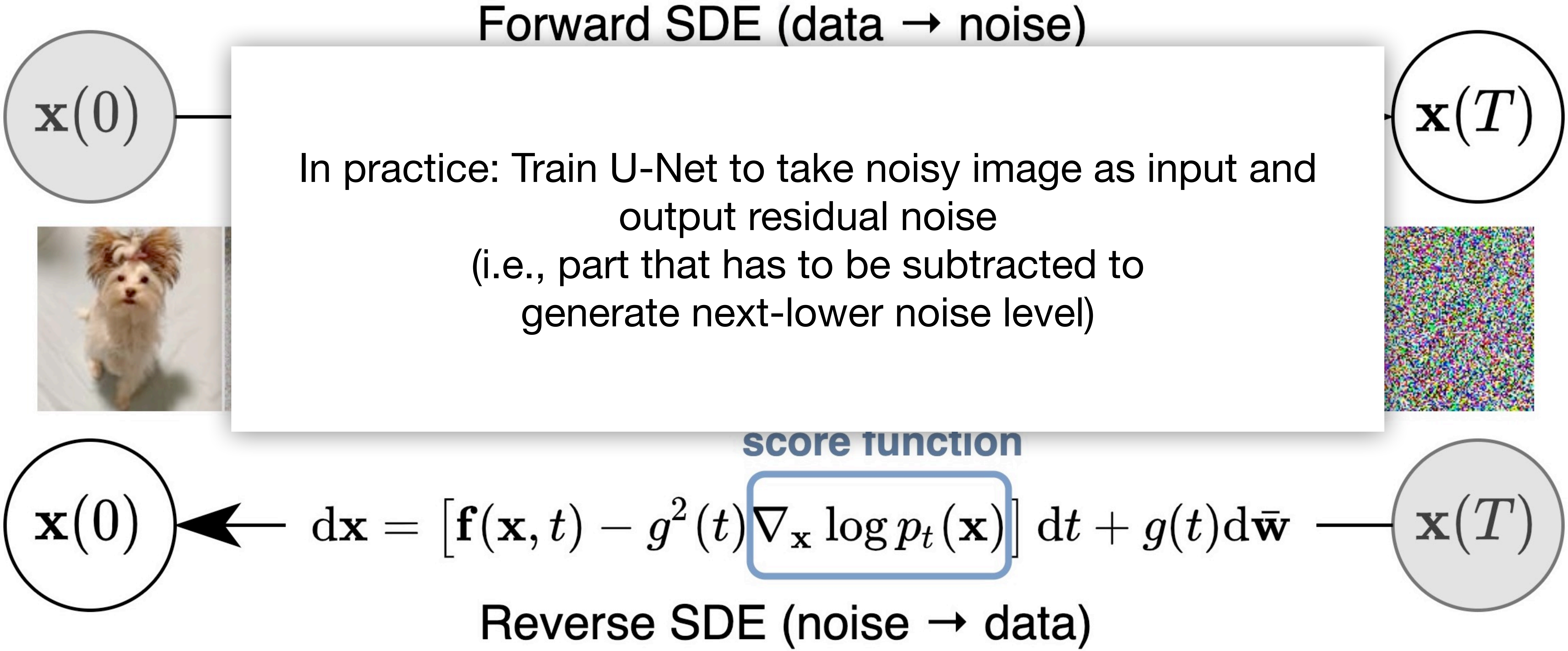
$$\mathbf{x}(0) \longrightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \longrightarrow \mathbf{x}(T)$$

score function

$$\mathbf{x}(0) \longleftarrow \mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\right]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \longleftarrow \mathbf{x}(T)$$

Reverse SDE (noise → data)

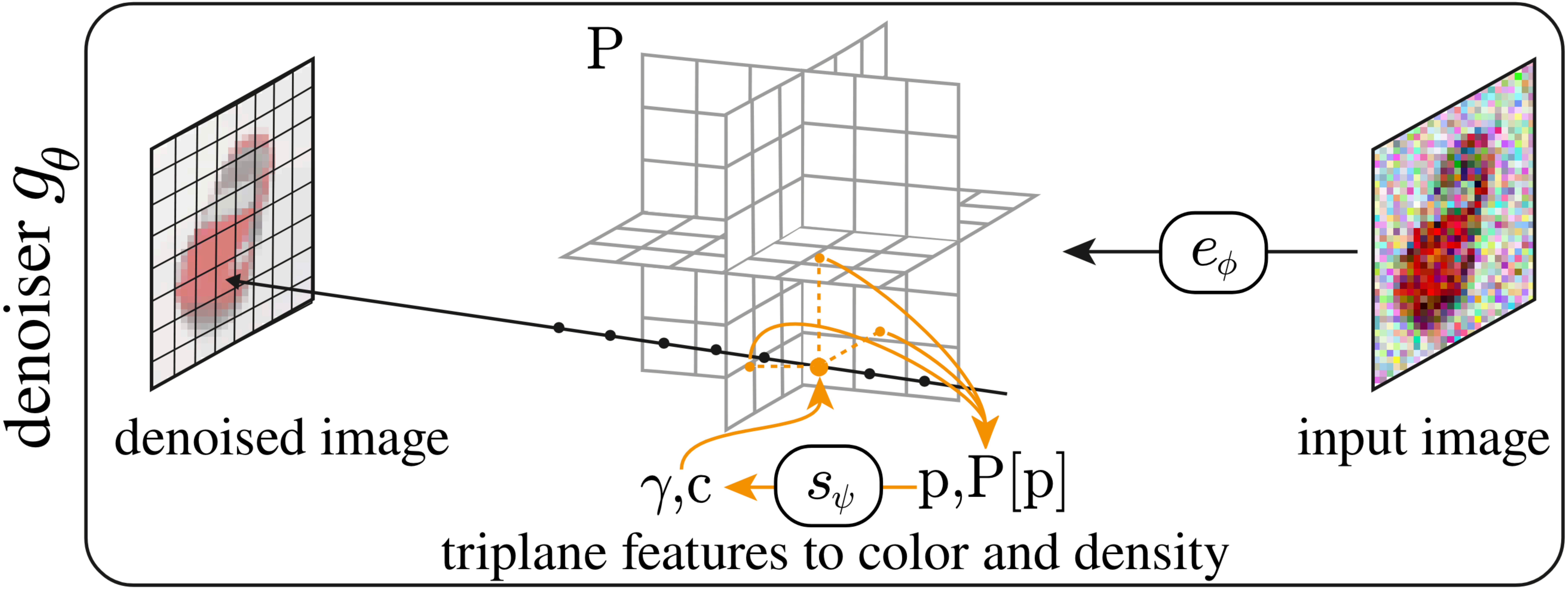
Background: Diffusion Models



In practice: Train U-Net to take noisy image as input and output residual noise
(i.e., part that has to be subtracted to generate next-lower noise level)

RenderDiffusion

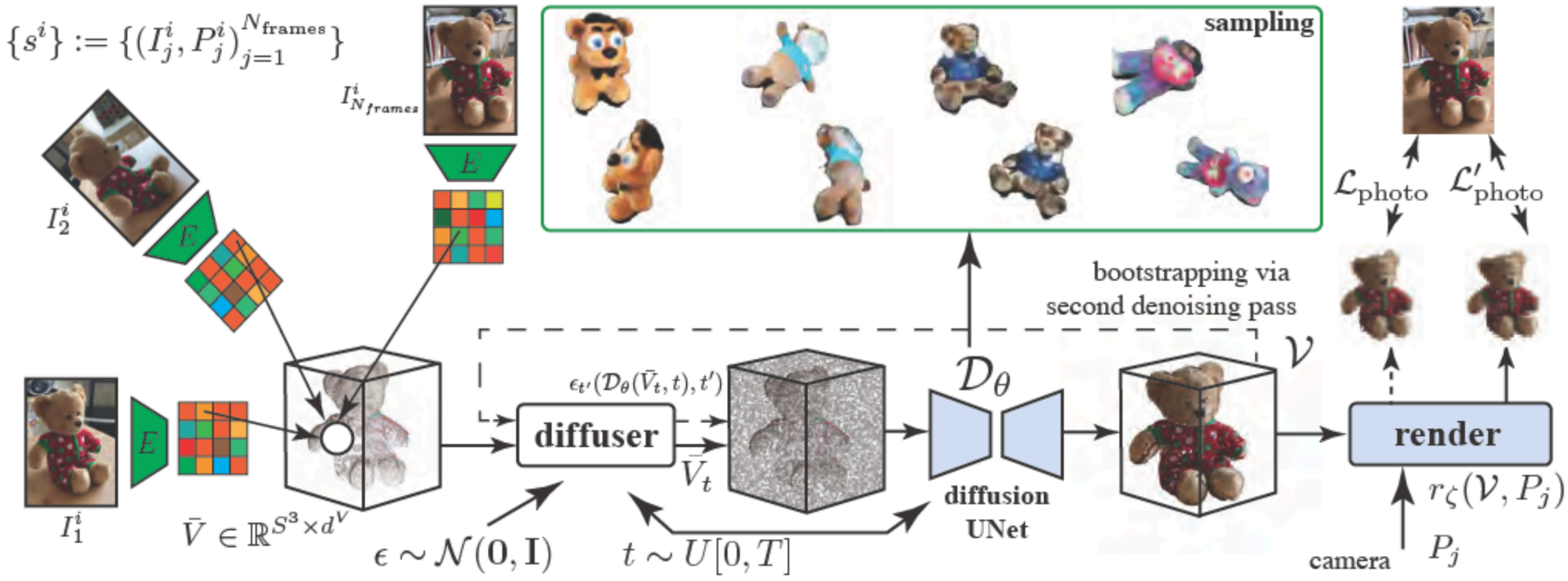
Anciukevicius et al. 2023



RenderDiffusion
EG3D
gen. view
test views
gen. view
test views
test views
test views

HoloDiffusion

Karnewar et al. 2023



HoloDiffusion

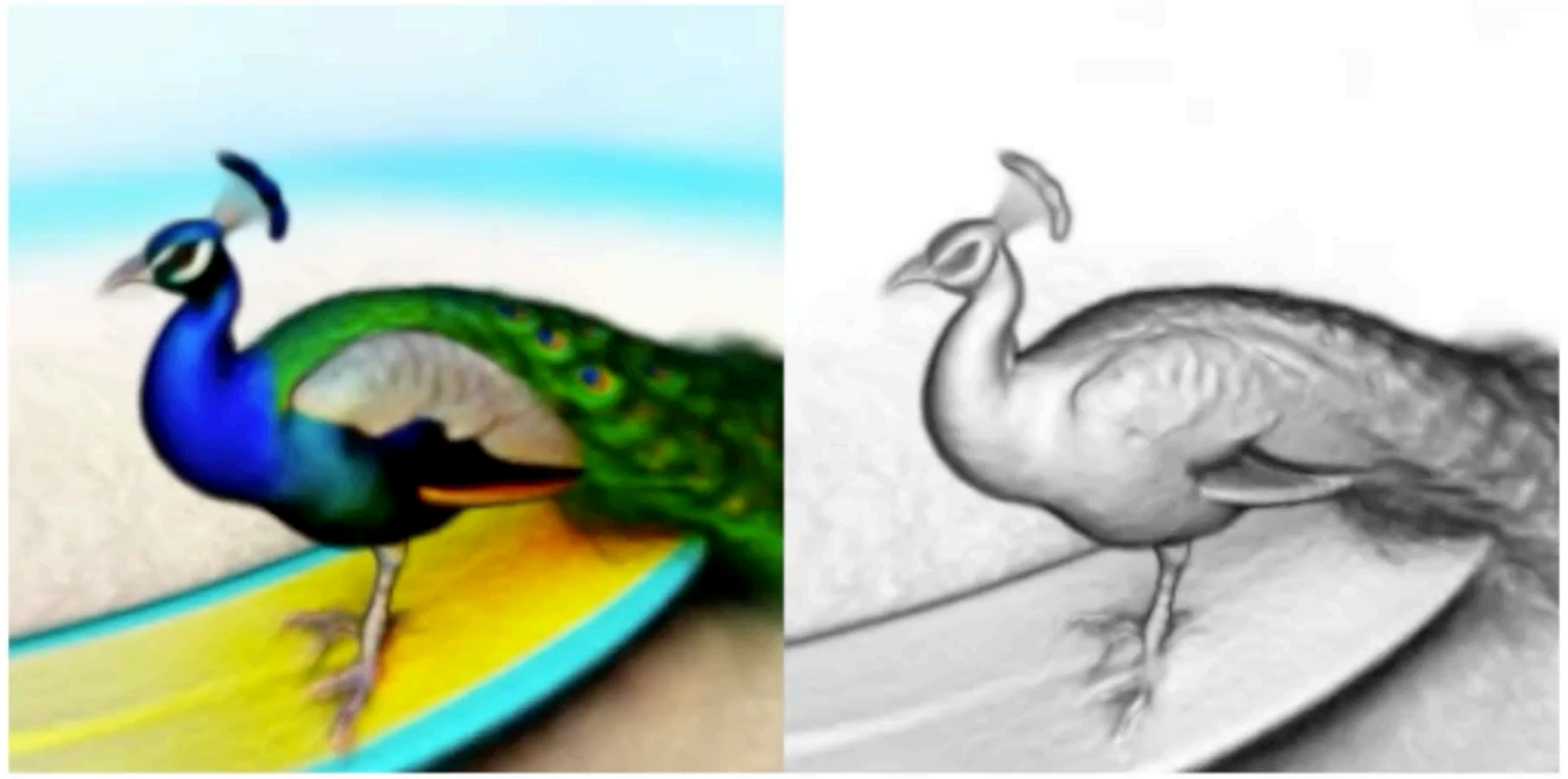
Karnewar et al. 2023

Score Distillation Sampling

Poole et al. 2023

"a DSLR photo of a peacock on a surfboard"

DreamFusion
Automatic text-to-3D

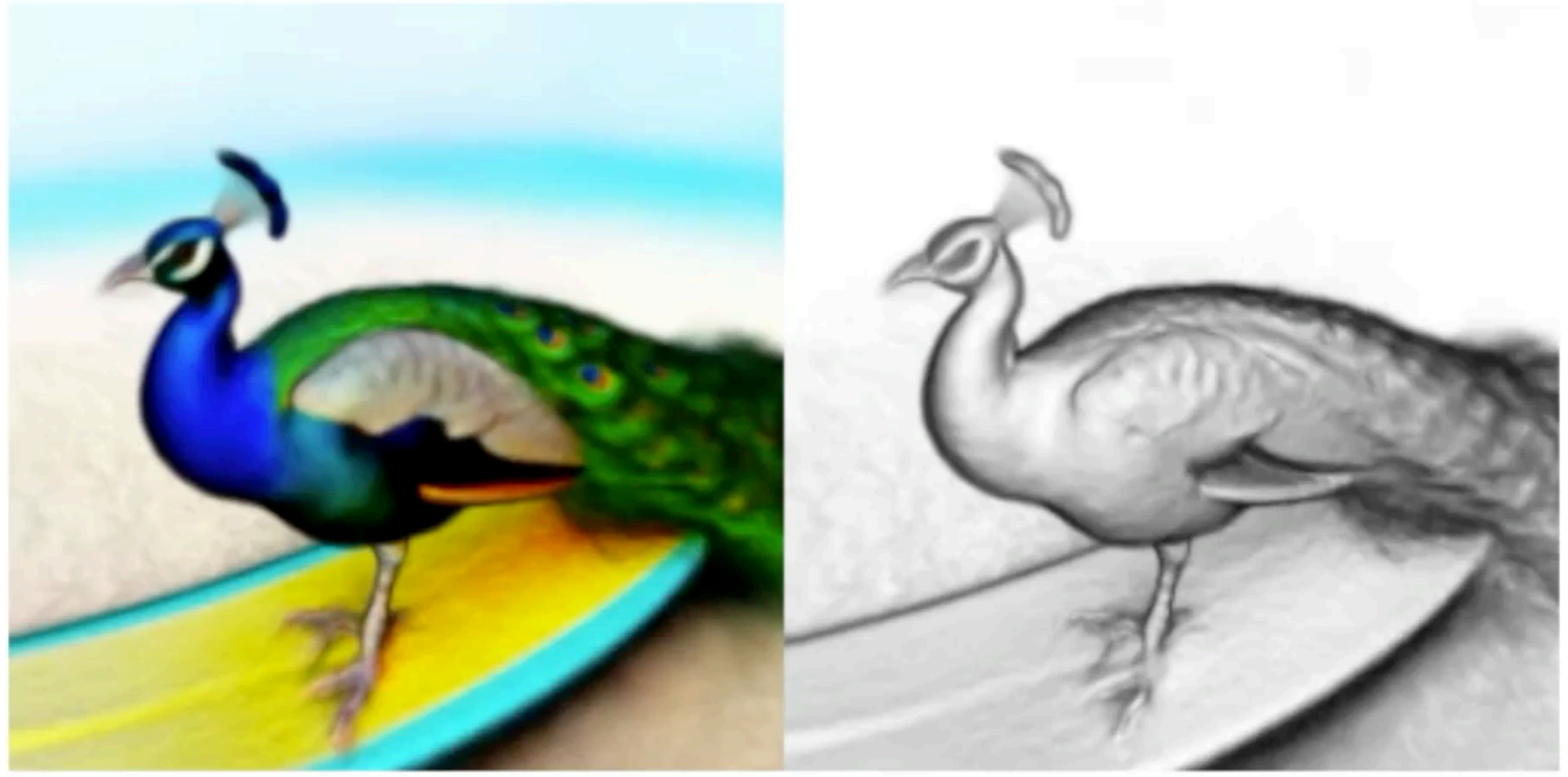


Score Distillation Sampling

Poole et al. 2023

"a DSLR photo of a peacock on a surfboard"

DreamFusion
Automatic text-to-3D



Score Distillation Sampling

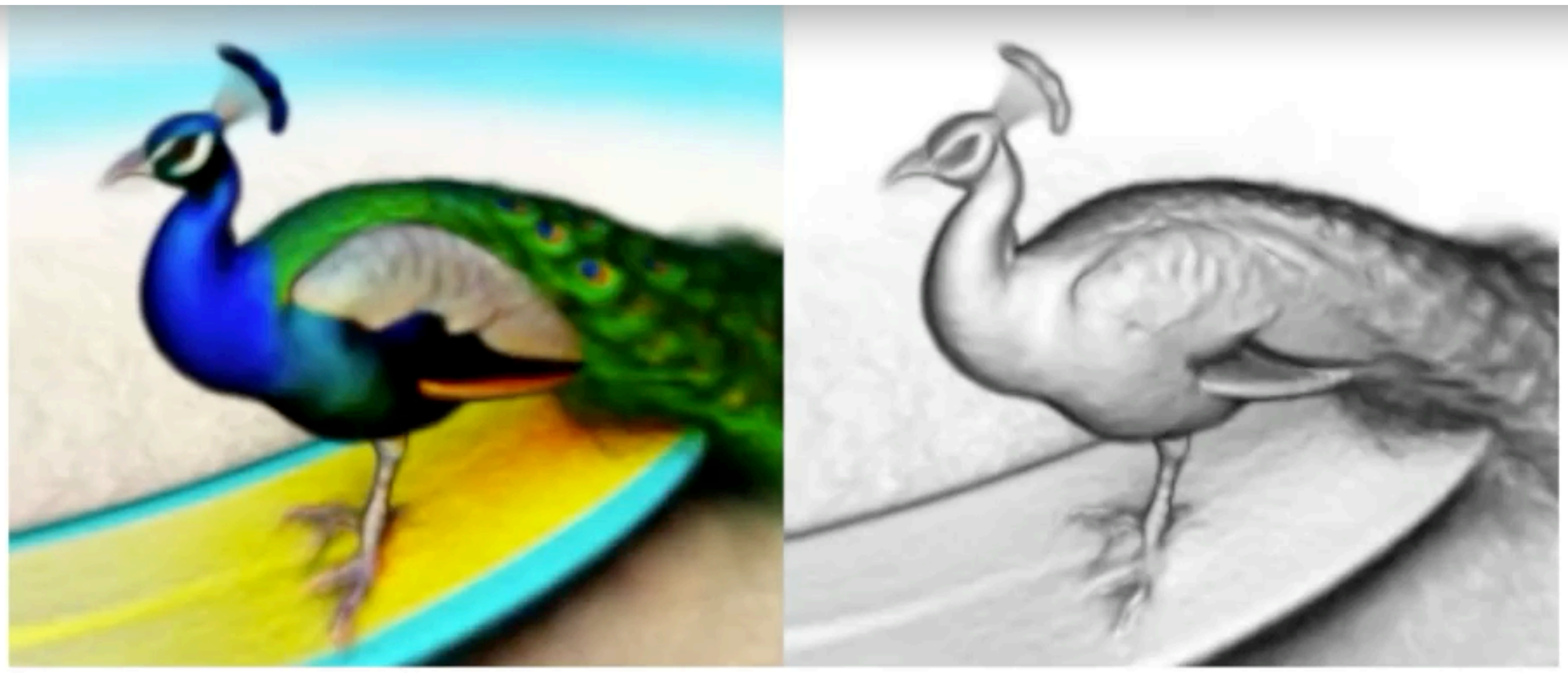
Poole et al. 2023

"a DSLR photo of a peacock on a surfboard"

Diffusion is in 2D

Text-conditional

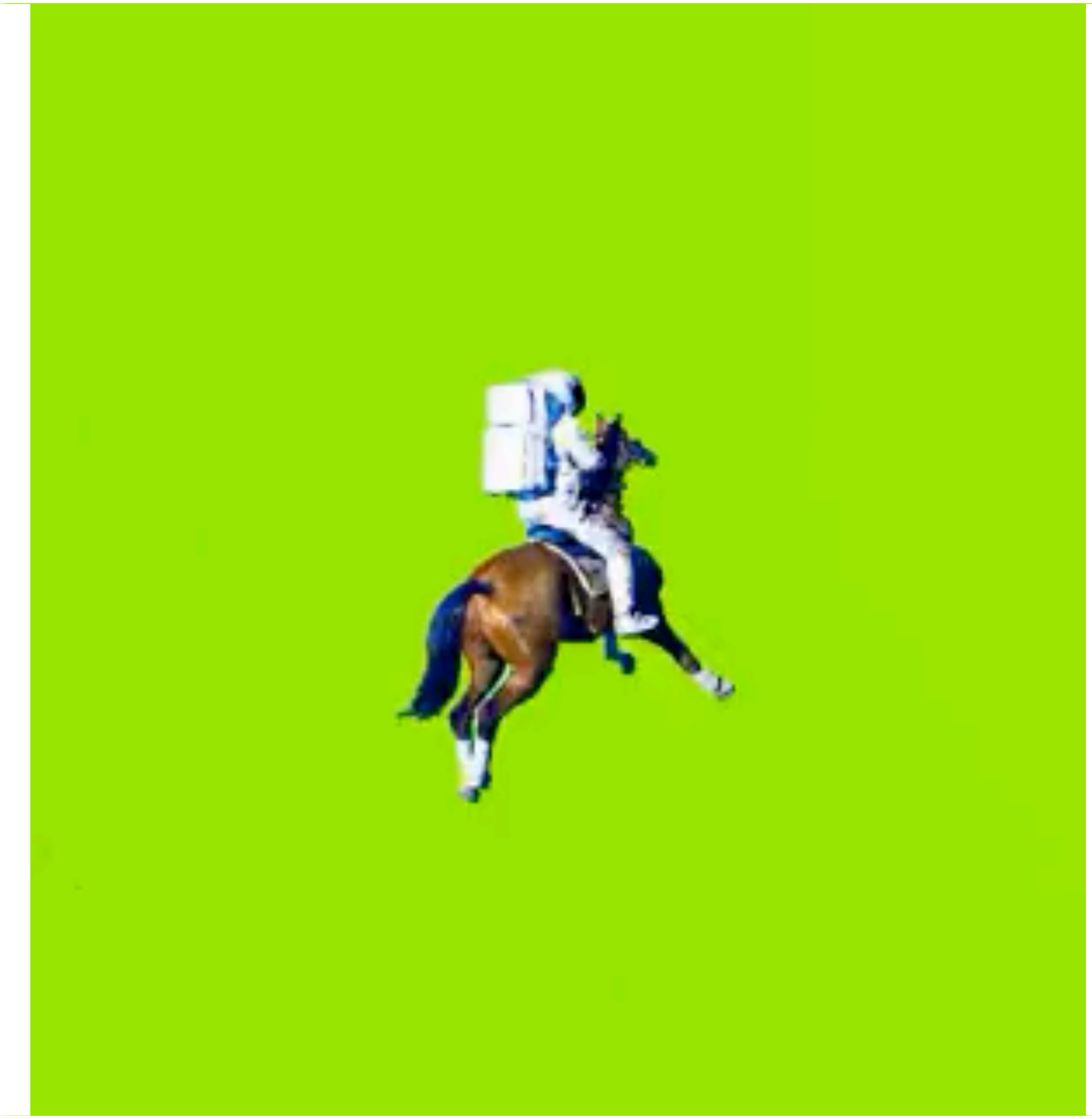
This CVPR: 3D Diffusion Models for 3D Scenes!



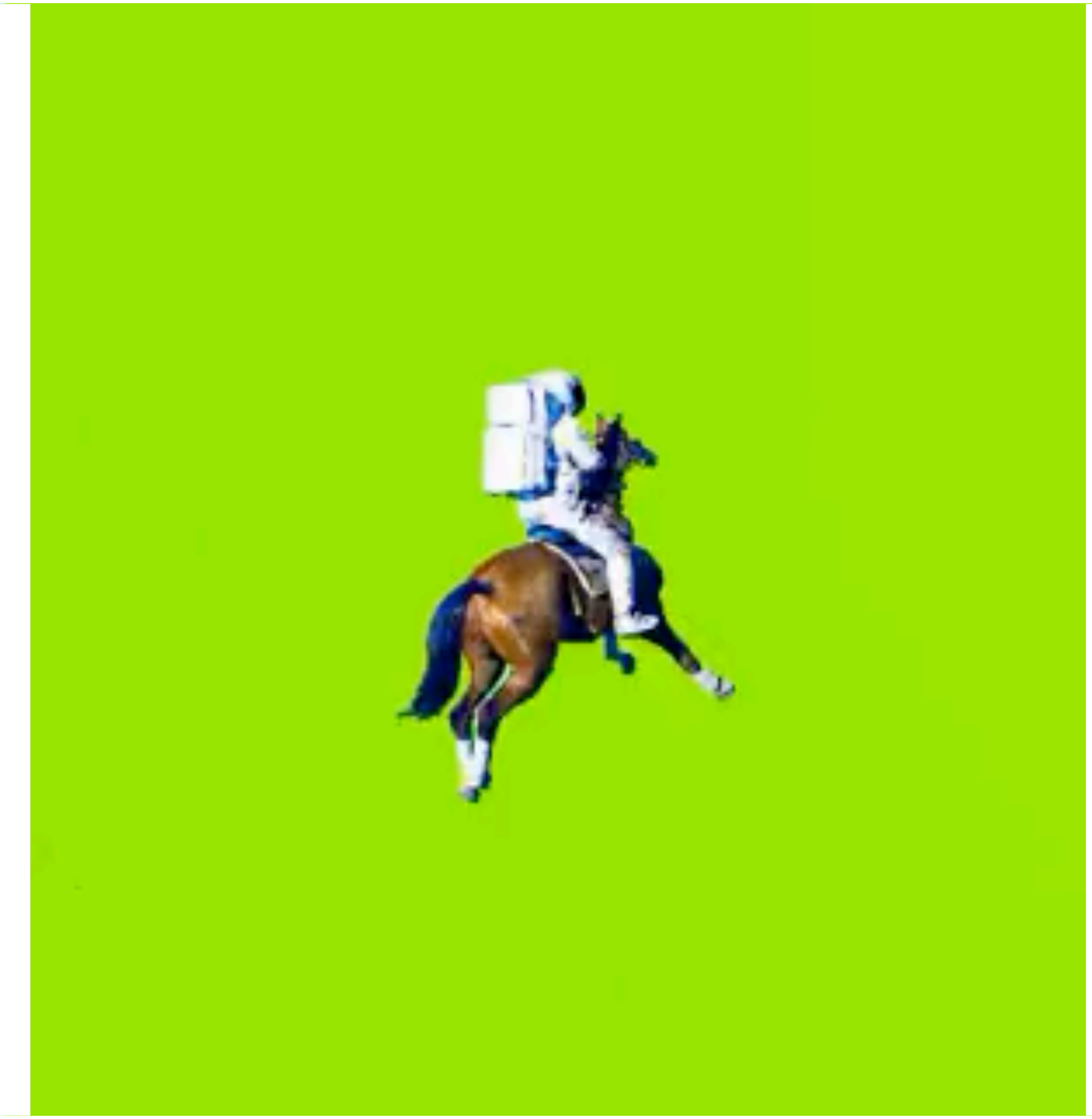
The "Janus Problem"

Multi-face Janus Problem

The “Janus Problem”



The "Janus Problem"

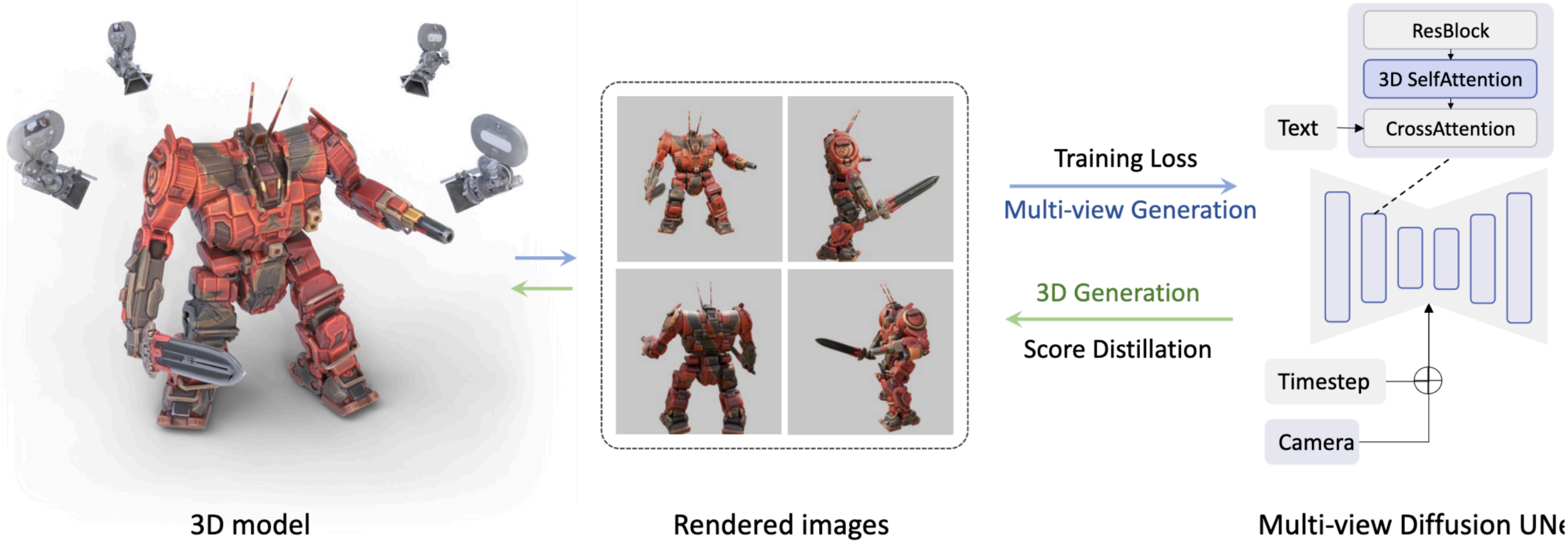


The “Janus Problem”

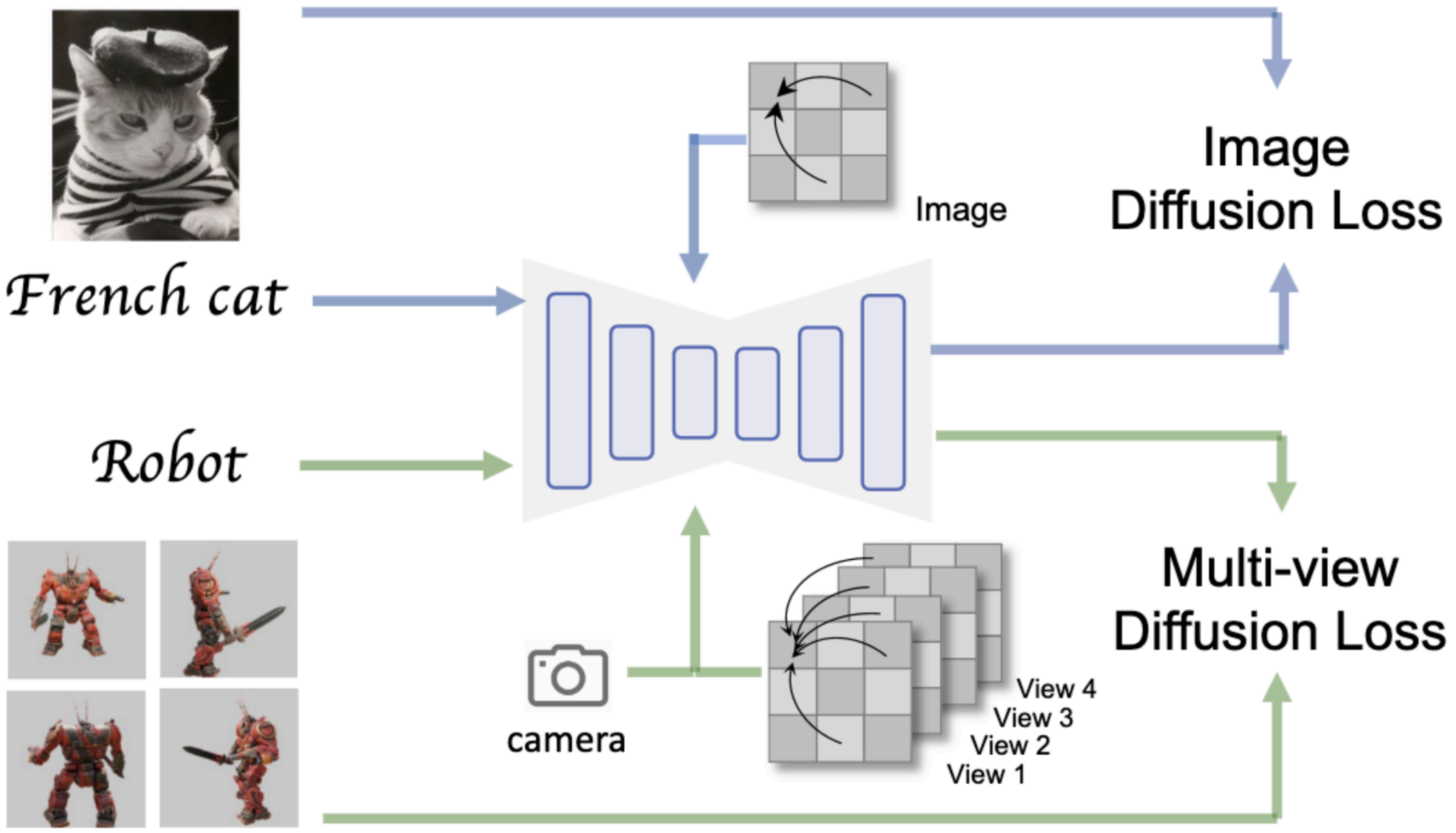
The "Janus Problem"

MVDream: Multi-View Diffusion for 3D Generation

Shi et al. 2023



Multi-view Diffusion Training



MultiView Diffusion Largely Solves the Janus Problem

MultiView Diffusion Largely Solves the Janus Problem