GIRAFFE: Representing Scenes as Compositional Generative Neural Feature Fields

Michael Niemeyer, Andreas Geiger

CVPR 2021 Best Paper

STRUCT Group Seminar
Presenter: Wenjing Wang
2020.07.13

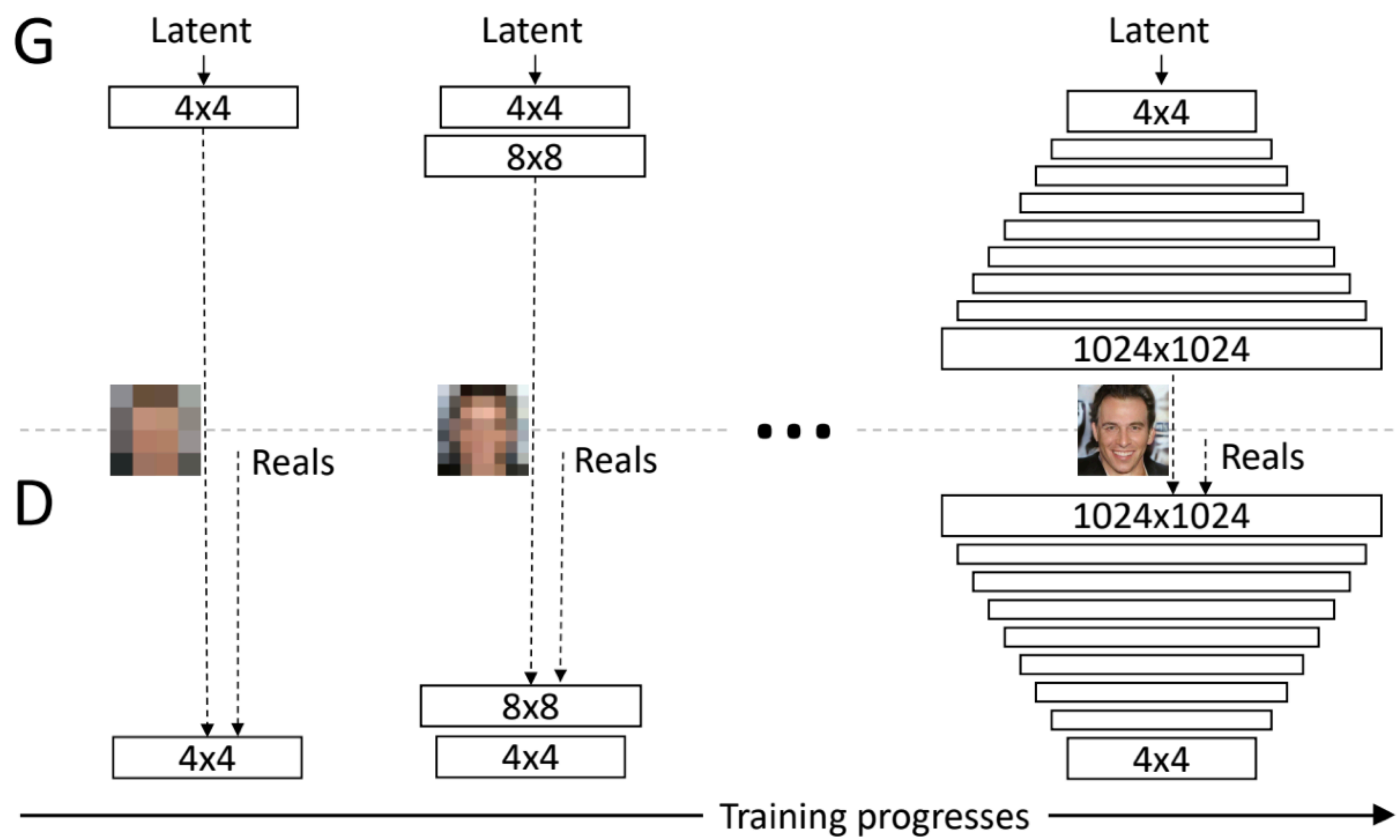
OUTLINE

- Authorship
- Background
- Proposed Method
- Experimental Results
- Conclusion

BACKGROUND

➤ GAN generator $\mathbf{I} = G(\mathbf{z})$, z: d-dimensional latent; I: image

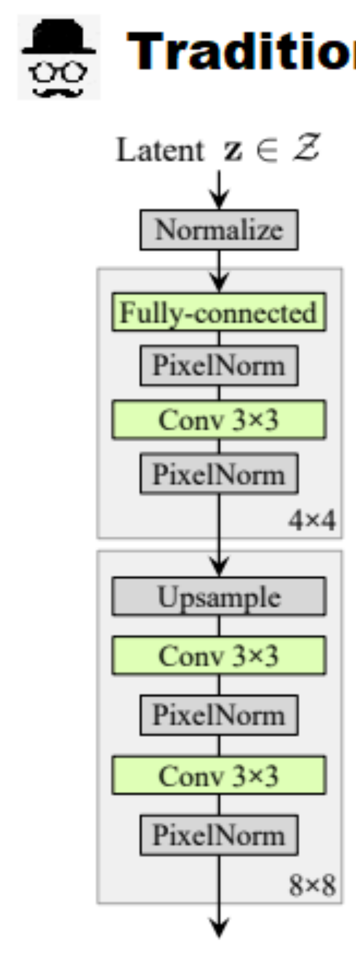
- PGGAN (ICLR-18)

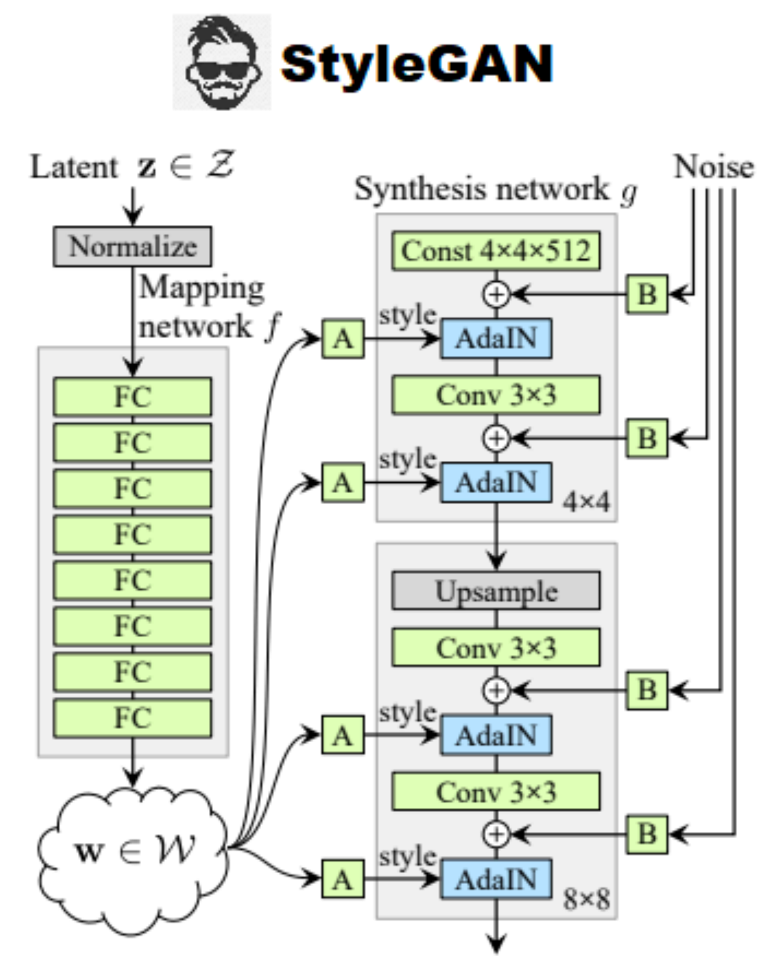


BACKGROUND

➤ Controlable image generation

- StyleGAN (CVPR-19)





Our generator thinks of an image as a collection of "styles", where each style controls the effects at a particular scale

- Coarse styles → pose, hair, face shape
- Middle styles → facial features, eyes
- Fine styles → color scheme

BACKGROUND

➤ Controlable image generation

- Closed-Form Factorization of Latent Semantics in GANs (CVPR21)

- GAN generator $\mathbf{I} = G(\mathbf{z})$, z: d-dimensional latent; I: image
- Manipulation/Editing only consider the first projection step

$$\begin{aligned}\mathbf{y}' \triangleq G_1(\mathbf{z}') &= G_1(\mathbf{z} + \alpha\mathbf{n}) \\ &= \mathbf{A}\mathbf{z} + \mathbf{b} + \alpha\mathbf{A}\mathbf{n} = \mathbf{y} + \alpha\mathbf{A}\mathbf{n}\end{aligned}$$

BACKGROUND

- ➤ Controlable image generation
- • Closed-Form Factorization of Latent Semantics in GANs (CVPR21)

Posture (Left & Right)

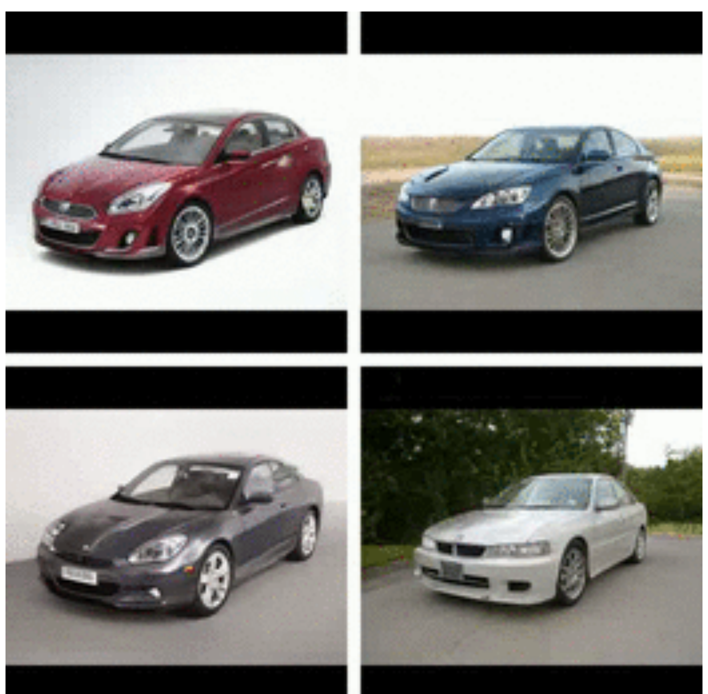
Posture (Up & Down)

Zoom

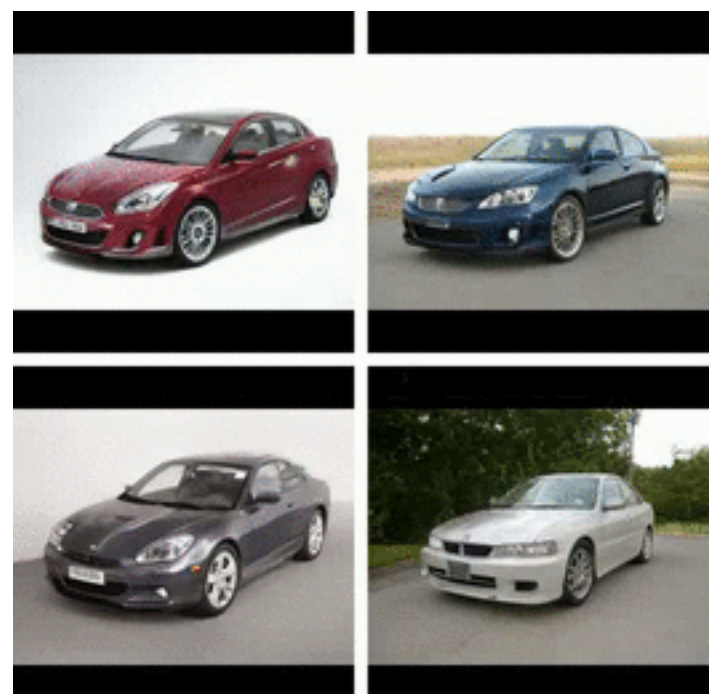
BACKGROUND

- ➤ Controlable image generation
- • Closed-Form Factorization of Latent Semantics in GANs (CVPR21)

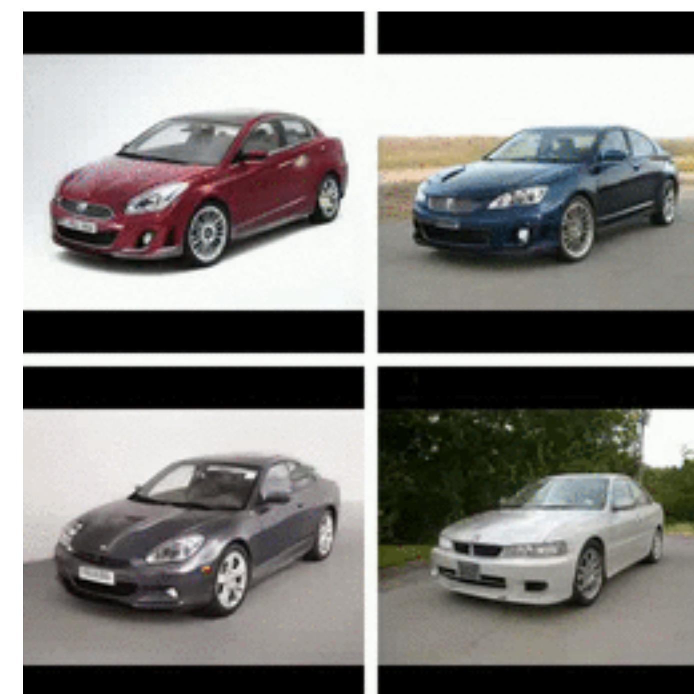
Orientation



Vertical Position



Shape



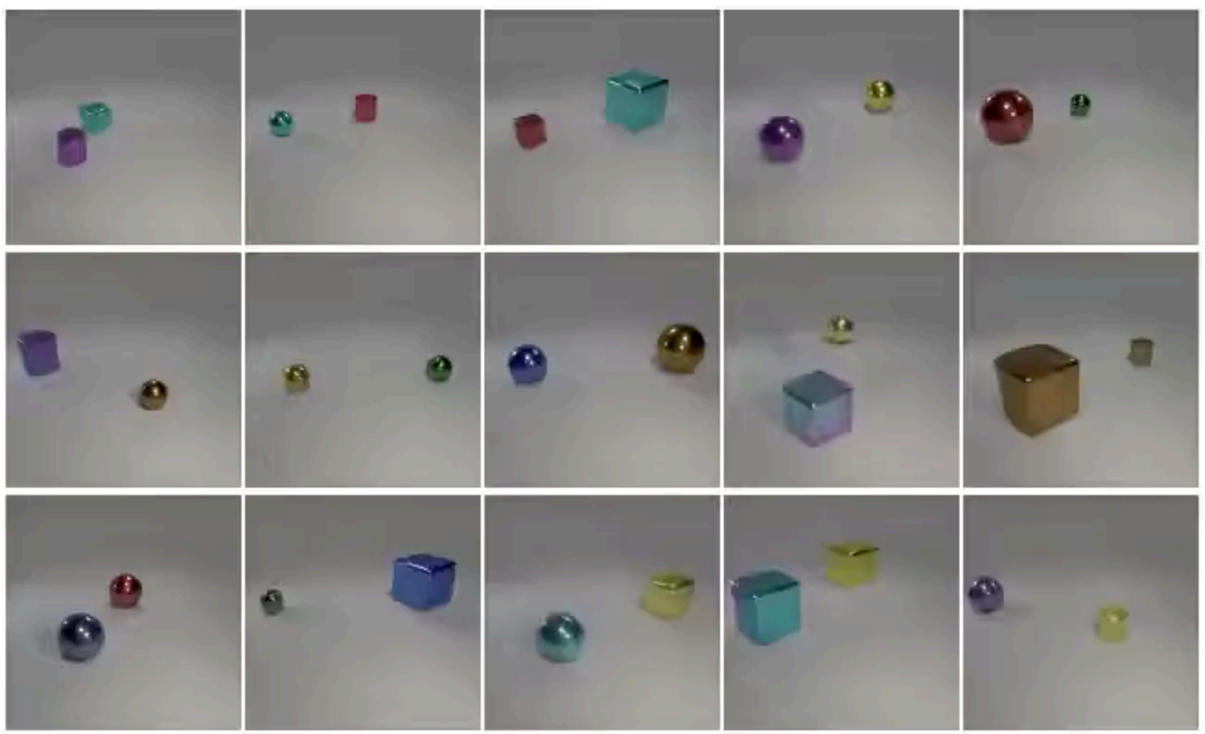
OUTLINE

- Authorship
- Background
- Proposed Method
- Experimental Results
- Conclusion

PROPOSED METHOD

➤ Controlable image generation

- Single-object translation



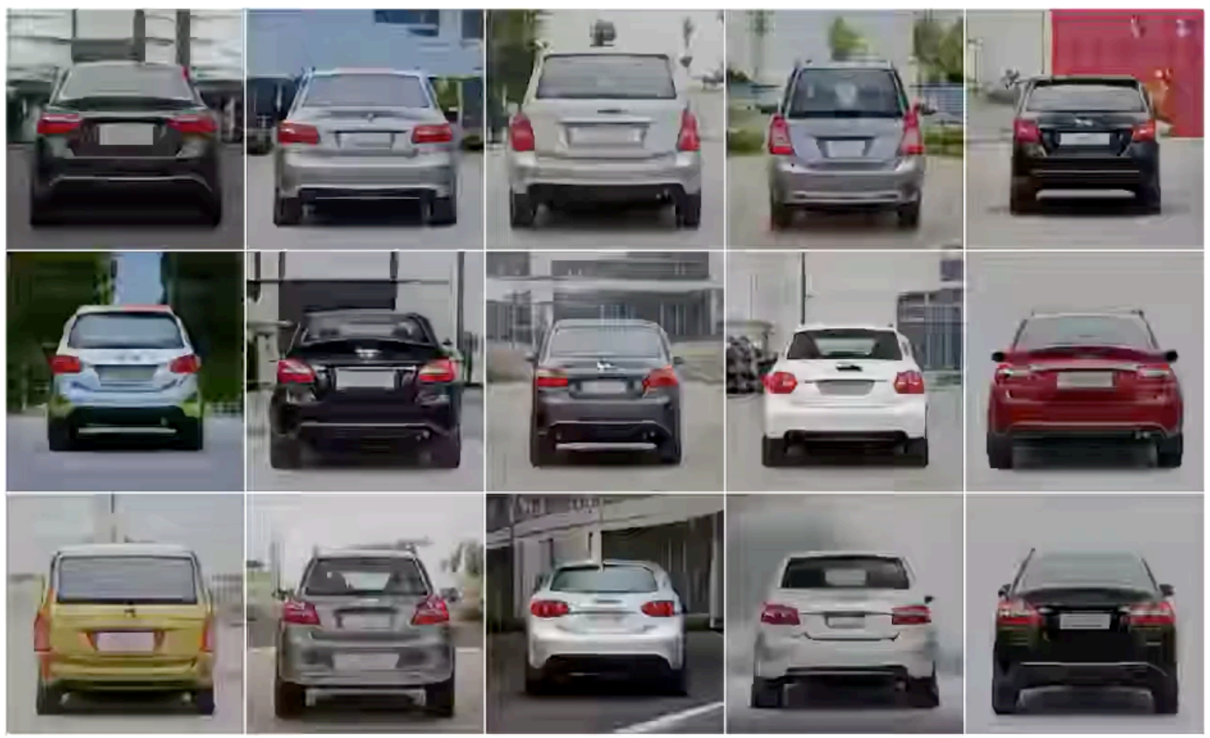
2D-based GAN

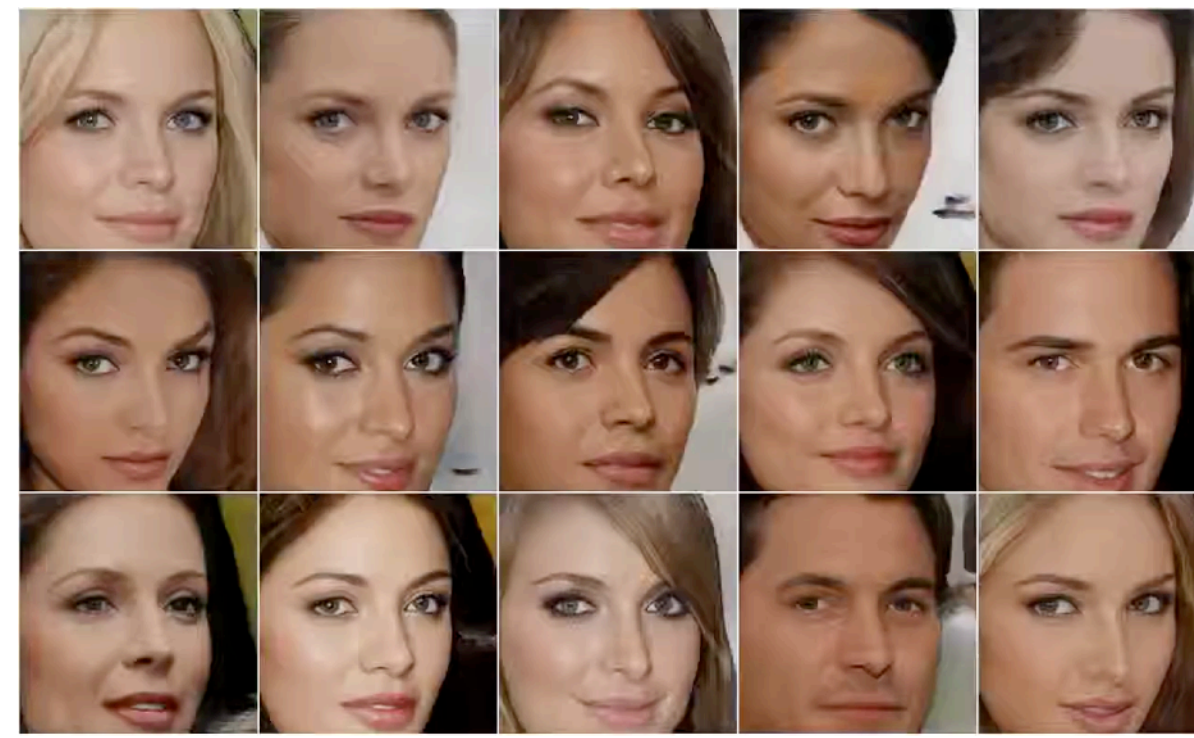
Our Method

PROPOSED METHOD

➤ Controlable image generation

- Rotate object

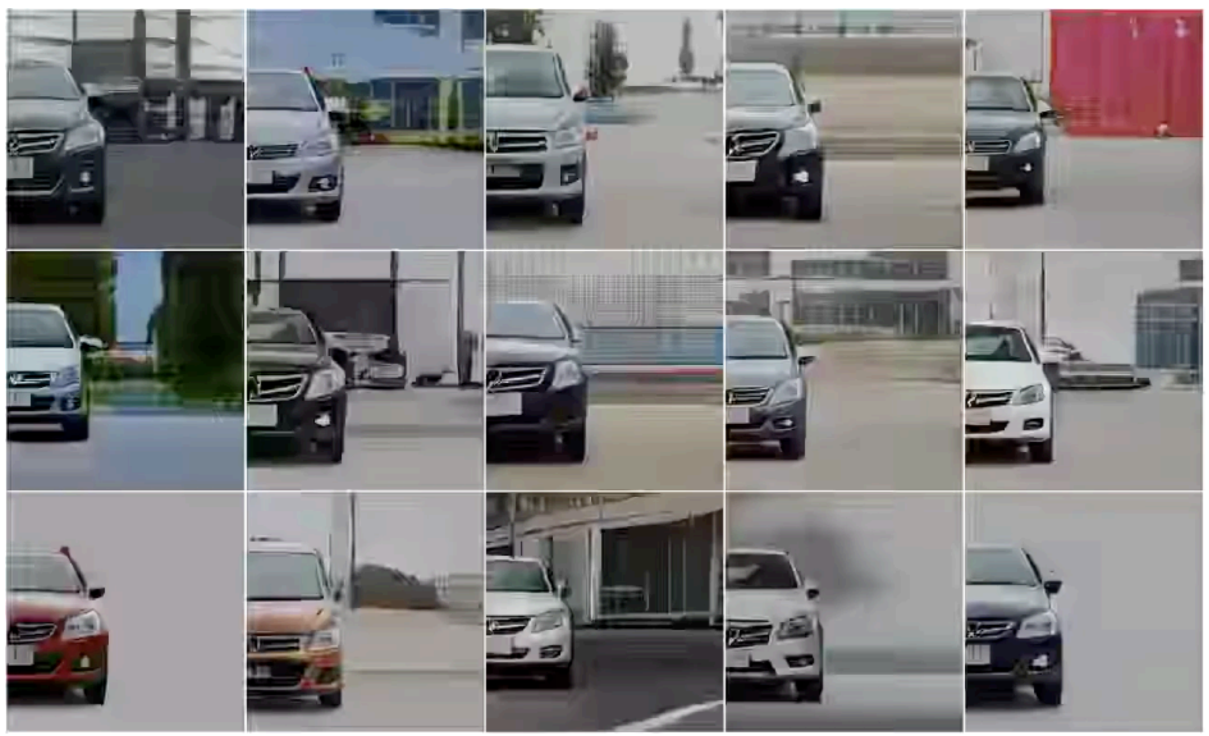




PROPOSED METHOD

➤ Controlable image generation

- Horizontal/Vertical translation



PROPOSED METHOD

➤ Controlable image generation

- Change object/background appearance

PROPOSED METHOD

- Controlable image generation

Change Background Appearance

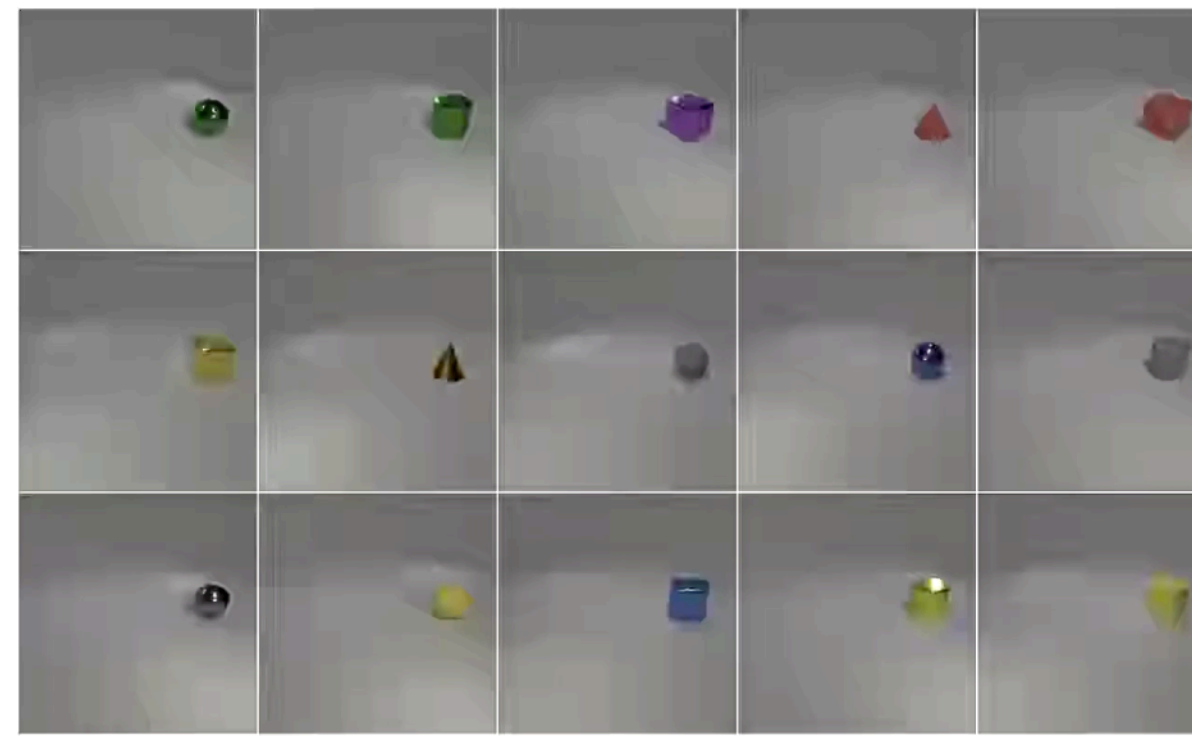
Circular Translation

PROPOSED METHOD

➤ Controlable image generation

- Out-of-Distribution Generalization

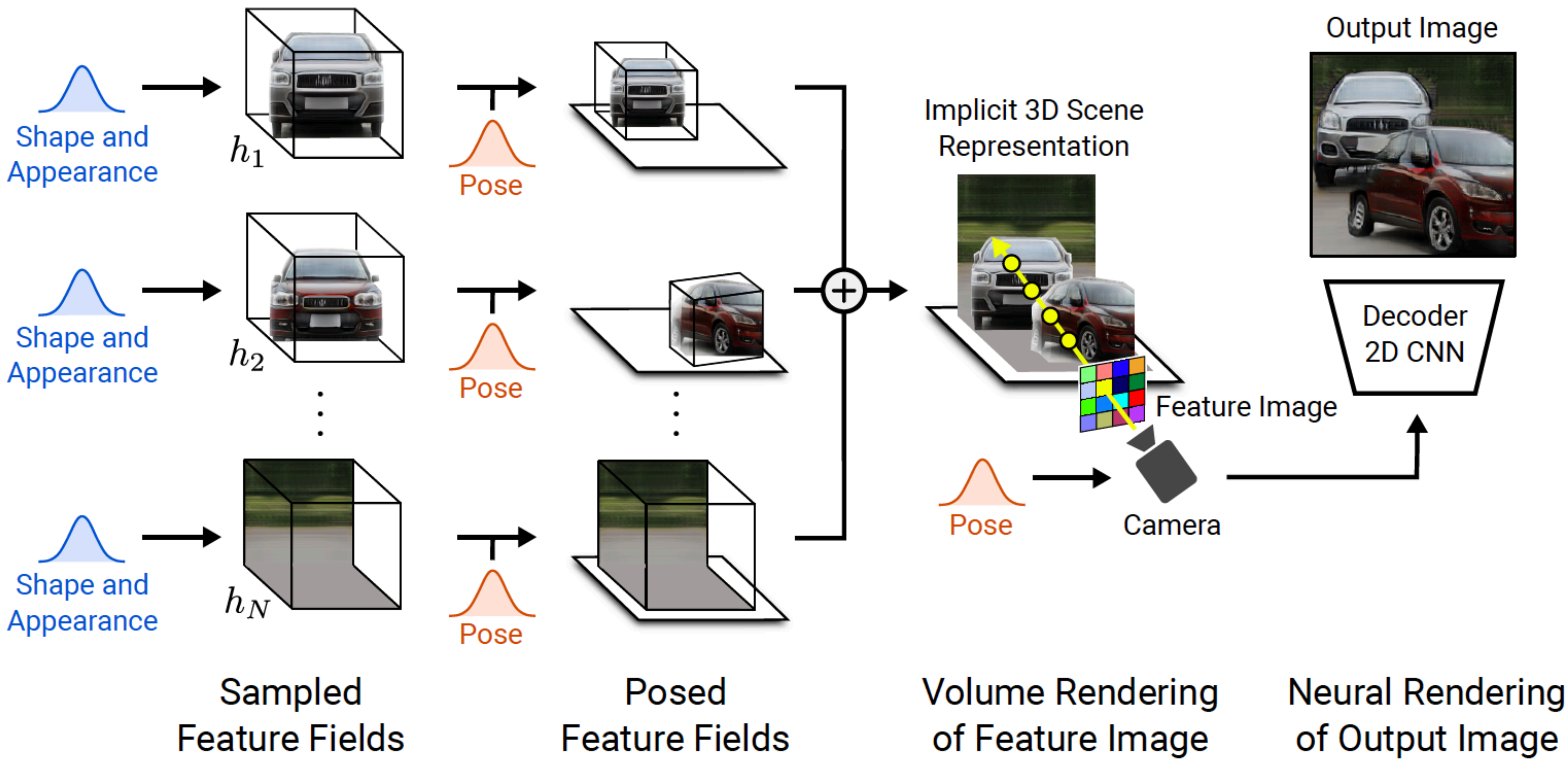
Trained On One-Object Scenes



Trained On Two-Object Scenes

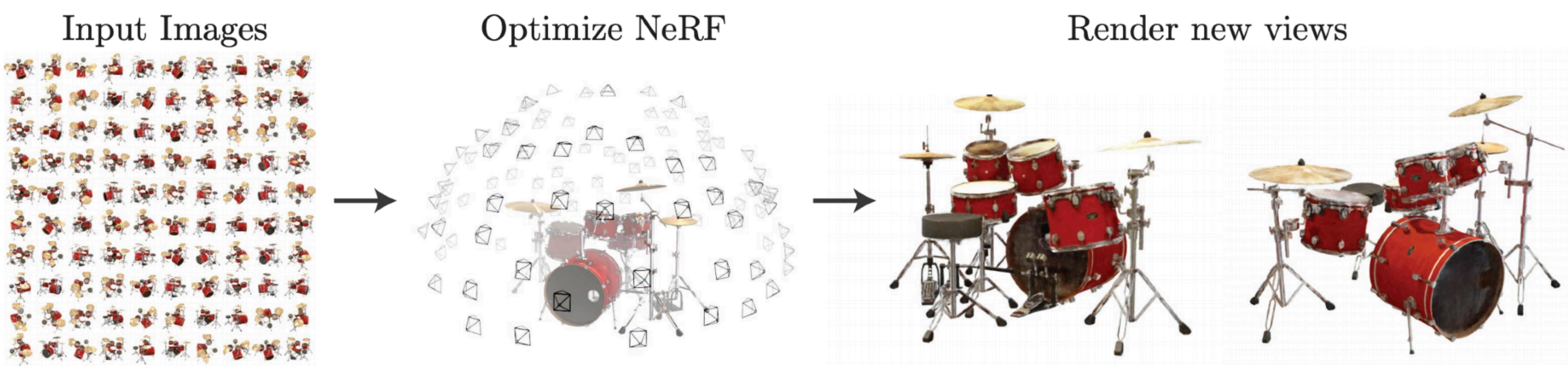
PROPOSED METHOD

➤ Compositional Generative Neural Feature Fields



PROPOSED METHOD

- NeRF (ECCV20, Best Paper Honorable Mention)
- Task: optimizes a continuous 5D neural radiance field representation



PROPOSED METHOD

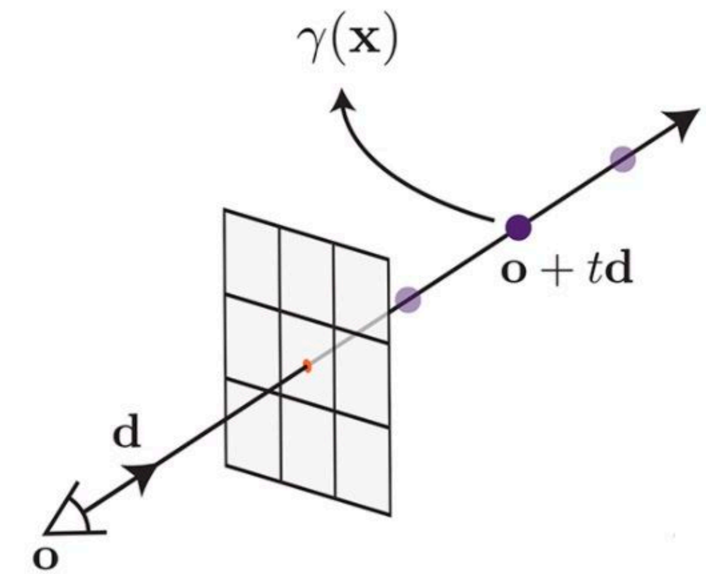
➤ NeRF (ECCV20, Best Paper Honorable Mention)

- Neural Radiance Field
- Input:
  - 3D point, $(x,y,z) \in R^3$ (R^{Lx})
  - Viewing direction, $(\theta,\varphi) \in R^2$ (R^{Ld})
- Output:
  - Volume density, $\sigma \in R^+$
  - RGB color value, $(r,g,b) \in R^3$

$$\boxed{\begin{array}{c} f_\theta : \mathbb{R}^{L_{\mathbf{x}}} \times \mathbb{R}^{L_{\mathrm{d}}} \to \mathbb{R}^+ \times \mathbb{R}^3 \\ (\gamma(\mathbf{x}), \gamma(\mathbf{d})) \mapsto (\sigma, \mathbf{c}) \end{array}}$$

PROPOSED METHOD

➤ NeRF (ECCV20, Best Paper Honorable Mention)



- Volume Rendering

The volume density σ(x) can be interpreted as the differential probability of a ray terminating at an infinitesimal particle at location x.

Color: $$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt\,, \text{ where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$$

Camera ray: $$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

PROPOSED METHOD

- ➤ NeRF (ECCV20, Best Paper Honorable Mention)

$$\boxed{\begin{gathered} f_\theta : \mathbb{R}^{L_{\mathbf{x}}} \times \mathbb{R}^{L_{\mathrm{d}}} \to \mathbb{R}^+ \times \mathbb{R}^3 \\ (\gamma(\mathbf{x}), \gamma(\mathbf{d})) \mapsto (\sigma, \mathbf{c}) \end{gathered}}$$

- Volume Rendering

Discrete Version:

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i \delta_i))\mathbf{c}_i \,, \text{ where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$$

$$\delta_i = t_{i+1} - t_i$$

PROPOSED METHOD

➤ NeRF (ECCV20, Best Paper Honorable Mention)

- Positional encoding
  - Mapping the inputs to a higher dimensional space using high frequency functions before passing them to the network

    $\rightarrow$ Better fitting of data that contains high frequency variation

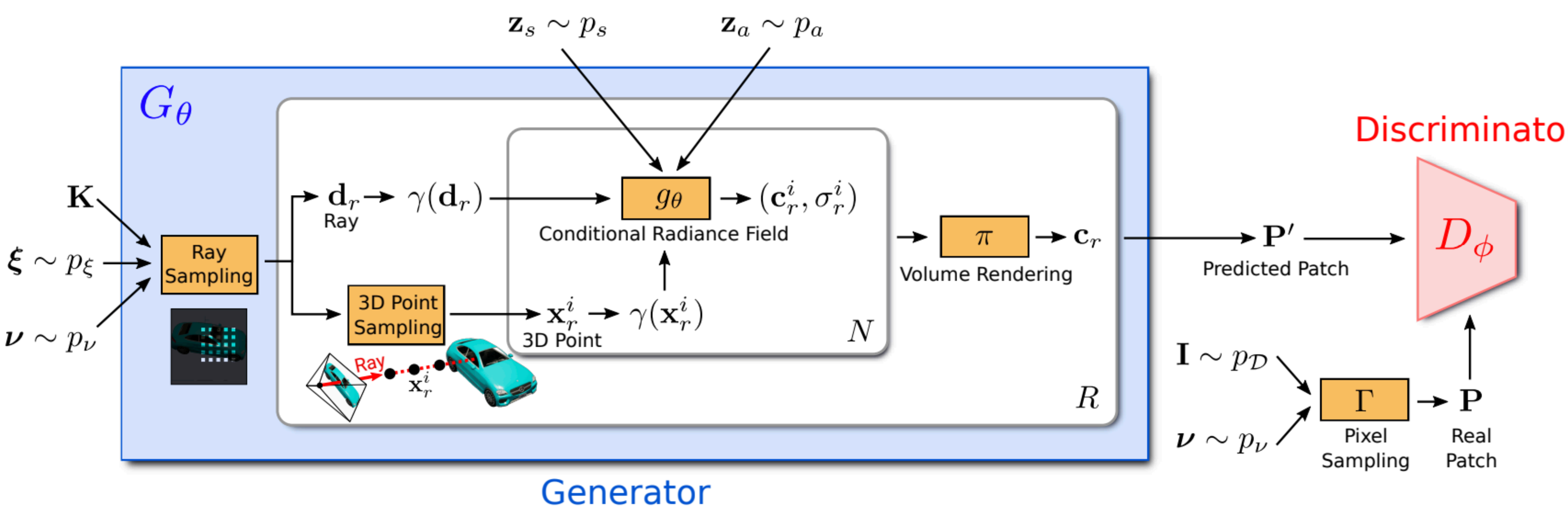
$$\gamma(p) = \left(\sin\left(2^0 \pi p\right), \cos\left(2^0 \pi p\right), \cdots , \sin\left(2^{L-1} \pi p\right), \cos\left(2^{L-1} \pi p\right) \right)$$

  - $\gamma(\cdot)$ applied separately to each values in x and d
  - x and d normalized to $[-1,1]$

PROPOSED METHOD

➤ GRAF (NeurIPS20)

$$g_\theta : \mathbb{R}^{L_\mathbf{x}} \times \mathbb{R}^{L_\mathbf{d}} \times \mathbb{R}^{M_s} \times \mathbb{R}^{M_a} \rightarrow \mathbb{R}^+ \times \mathbb{R}^3$$
$$(\gamma(\mathbf{x}), \gamma(\mathbf{d}), \mathbf{z}_s, \mathbf{z}_a) \mapsto (\sigma, \mathbf{c}) \qquad \mathbf{z}_s, \mathbf{z}_a \sim \mathcal{N}(\mathbf{0}, I)$$



PROPOSED METHOD

- NeRF (ECCV20, Best Paper Honorable Mention)

$$f_\theta : \mathbb{R}^{L_\mathbf{x}} \times \mathbb{R}^{L_\mathrm{d}} \rightarrow \mathbb{R}^+ \times \mathbb{R}^3 \qquad (\gamma(\mathbf{x}), \gamma(\mathbf{d})) \mapsto (\sigma, \mathbf{c})$$

PROPOSED METHOD

- NeRF (ECCV20, Best Paper Honorable Mention)

$$f_\theta : \mathbb{R}^{L_\mathbf{x}} \times \mathbb{R}^{L_\mathrm{d}} \to \mathbb{R}^+ \times \mathbb{R}^3 \qquad (\gamma(\mathbf{x}), \gamma(\mathbf{d})) \mapsto (\sigma, \mathbf{c})$$

- GRAF (NeurIPS20)

$$g_\theta : \mathbb{R}^{L_\mathbf{x}} \times \mathbb{R}^{L_\mathrm{d}} \times \boxed{\mathbb{R}^{M_s} \times \mathbb{R}^{M_a}} \to \mathbb{R}^+ \times \mathbb{R}^3$$

$$(\gamma(\mathbf{x}), \gamma(\mathbf{d}), \boxed{\mathbf{z}_s, \mathbf{z}_a}) \mapsto (\sigma, \mathbf{c}) \quad \mathbf{z}_s, \mathbf{z}_a \sim \mathcal{N}(\mathbf{0}, I)$$

Appearance control

PROPOSED METHOD

- NeRF (ECCV20, Best Paper Honorable Mention)

$$f_\theta : \mathbb{R}^{L_{\mathbf{x}}} \times \mathbb{R}^{L_{\mathrm{d}}} \rightarrow \mathbb{R}^{+} \times \mathbb{R}^{3} \qquad (\gamma(\mathbf{x}), \gamma(\mathbf{d})) \mapsto (\sigma, \mathbf{c})$$

- GRAF (NeurIPS20)

$$g_\theta : \mathbb{R}^{L_{\mathbf{x}}} \times \mathbb{R}^{L_{\mathrm{d}}} \times \boxed{\mathbb{R}^{M_s} \times \mathbb{R}^{M_a}} \rightarrow \mathbb{R}^{+} \times \mathbb{R}^{3}$$

$$(\gamma(\mathbf{x}), \gamma(\mathbf{d}), \boxed{\mathbf{z}_s, \mathbf{z}_a}) \mapsto (\sigma, \mathbf{c}) \quad \mathbf{z}_s, \mathbf{z}_a \sim \mathcal{N}(\mathbf{0}, I)$$

Appearance control

- This paper

$$h_\theta : \mathbb{R}^{L_{\mathbf{x}}} \times \mathbb{R}^{L_{\mathrm{d}}} \times \mathbb{R}^{M_s} \times \mathbb{R}^{M_a} \rightarrow \mathbb{R}^{+} \times \boxed{\mathbb{R}^{M_f}}$$

$$(\gamma(\mathbf{x}), \gamma(\mathbf{d}), \mathbf{z}_s, \mathbf{z}_a) \mapsto (\sigma, \boxed{\mathbf{f}})$$

Output feature vector

PROPOSED METHOD

➤ Object Representation

- NeRF and GRAF: the entire scene is represented by a single model
- This paper: control pose, shape and appearance of individual objects

Each object has a feature field + affine transformation

$$\mathbf{T} = \{\mathbf{s}, \mathbf{t}, \mathbf{R}\}$$

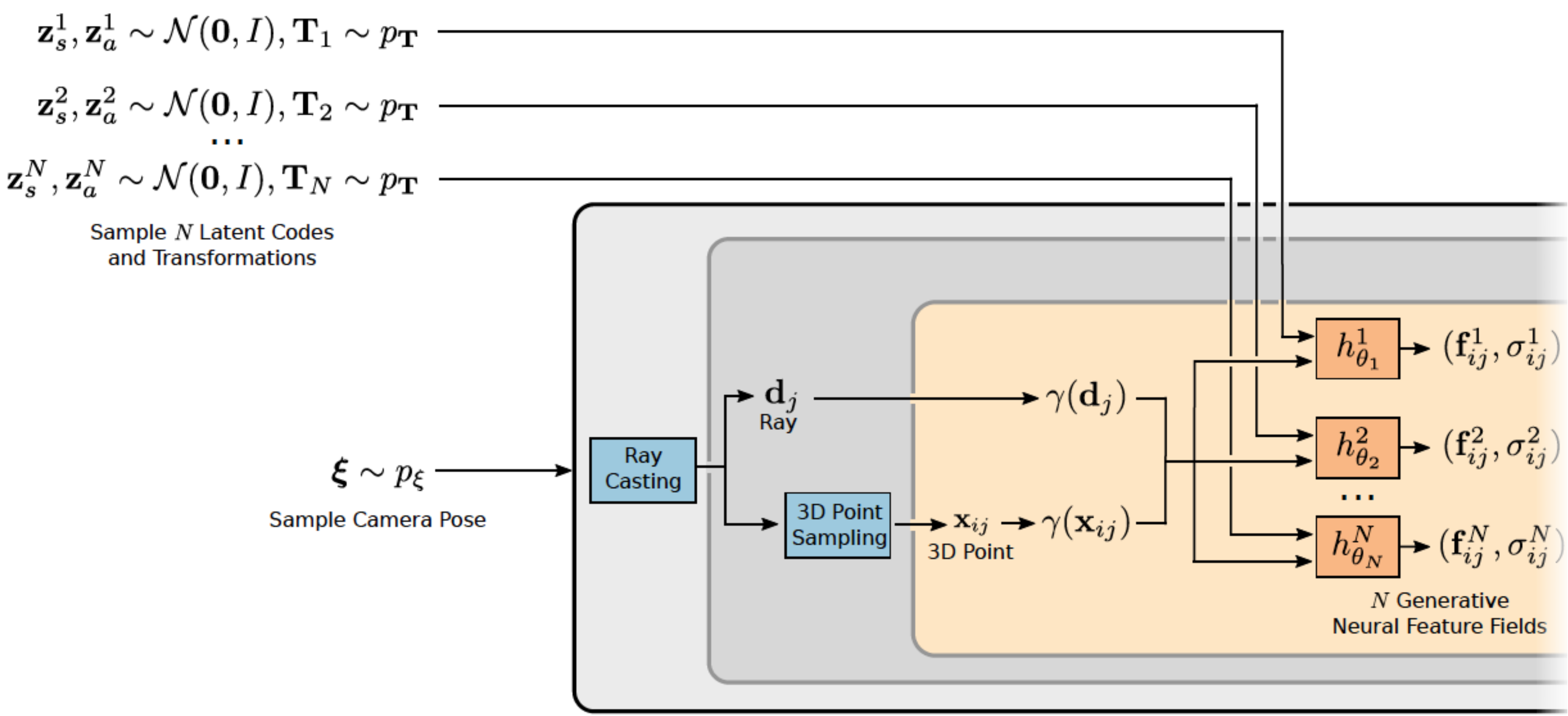
(scale, translation, rotation)

$$k(\mathbf{x}) = \mathbf{R} \cdot \begin{bmatrix} s_1 & & \\ & s_2 & \\ & & s_3 \end{bmatrix} \cdot \mathbf{x} + \mathbf{t}$$

Volume render in scene space and evaluate the feature field in its canonical object space

$$(\sigma, \mathbf{f}) = h_\theta(\gamma(k^{-1}(\mathbf{x})), \gamma(k^{-1}(\mathbf{d})), \mathbf{z}_s, \mathbf{z}_a)$$

PROPOSED METHOD



PROPOSED METHOD

➤ Scene Compositions

- N entities: N−1 objects + background
- Sum up the individual densities and to use the density-weighted mean to combine all features at (x, d)

$$C(\mathbf{x}, \mathbf{d}) = \left(\sigma, \frac{1}{\sigma} \sum_{i=1}^{N} \sigma_i \mathbf{f}_i \right), \text{where} \quad \sigma = \sum_{i=1}^{N} \sigma_i$$

Density $\sigma_i \in \mathbb{R}^+$

- Additional benefit: ensure gradient flow to all entities with a density greater than 0

PROPOSED METHOD

➤ Scene Rendering, two steps:

- 3D Volume Rendering

$$\pi_{\text{vol}} : (\mathbb{R}^+ \times \mathbb{R}^{M_f})^{N_s} \to \mathbb{R}^{M_f}, \quad \{\sigma_j, \mathbf{f}_j\}_{j=1}^{N_s} \mapsto \mathbf{f}$$

  - Numerical integration in NeRF

$$\mathbf{f} = \sum_{j=1}^{N_s} \tau_j \alpha_j \mathbf{f}_j \quad \tau_j = \prod_{k=1}^{j-1} (1 - \alpha_k) \quad \alpha_j = 1 - e^{-\sigma_j \delta_j}$$

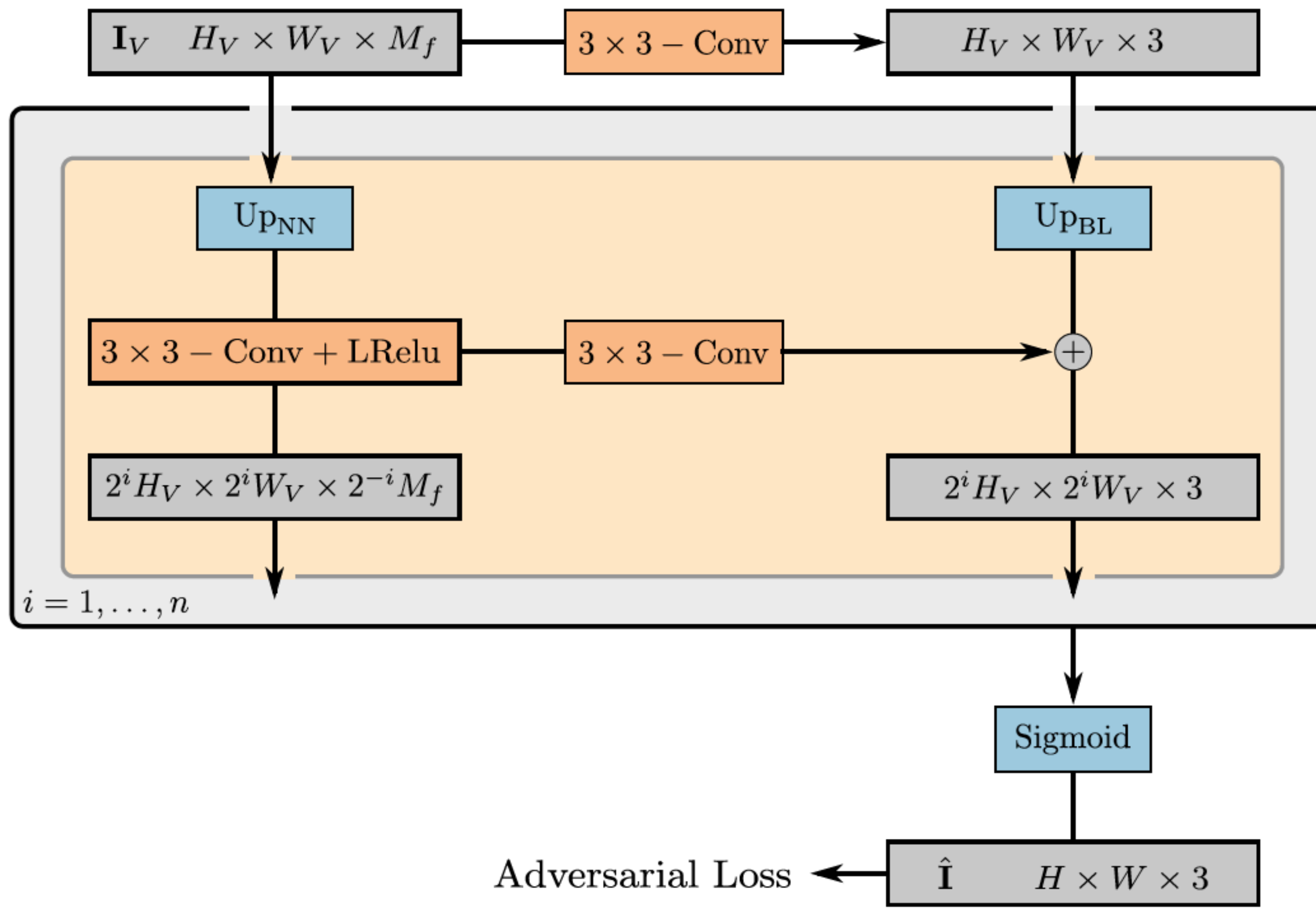
  - For efficiency, render feature at resolution 16×16

PROPOSED METHOD

- Scene Rendering, two steps:
- 2D Neural Rendering $\pi_\theta^{\text{neural}} : \mathbb{R}^{H_V \times W_V \times M_f} \to \mathbb{R}^{H \times W \times 3}$
- Design a 2D CNN
  - Small kernel sizes and no intermediate layers: only allow for spatially small refinements, avoid entangling global scene properties.
  - Map the feature image to an RGB image at every spatial resolution, and add the previous output to the next via bilinear upsampling.
  - Sigmoid activation to the last RGB layer.

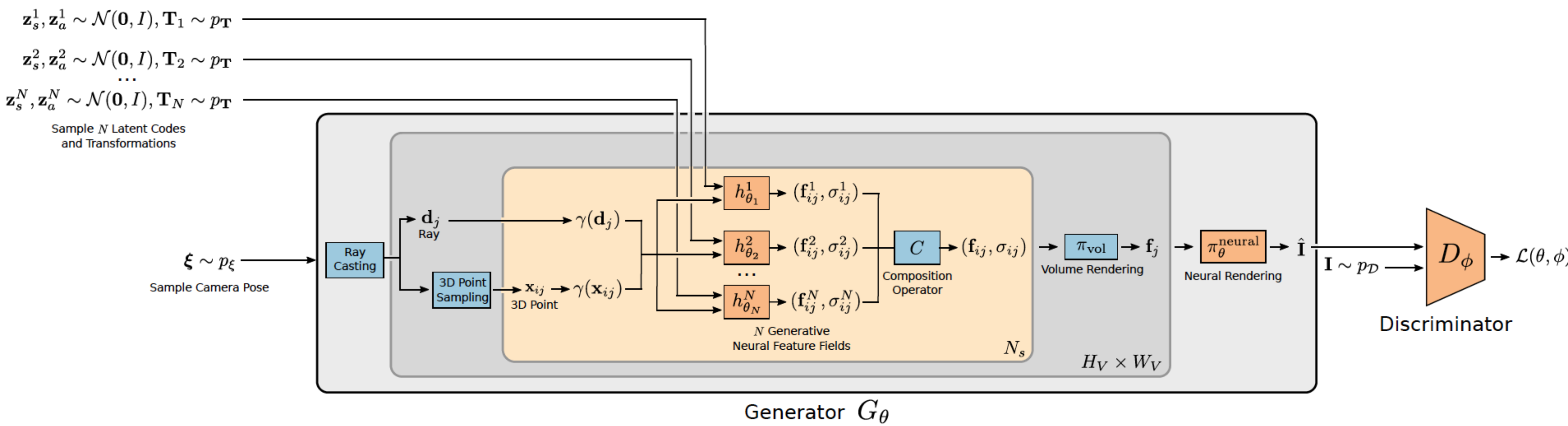
PROPOSED METHOD

- ➤ Scene Rendering, two steps:
- 2D Neural Rendering



PROPOSED METHOD

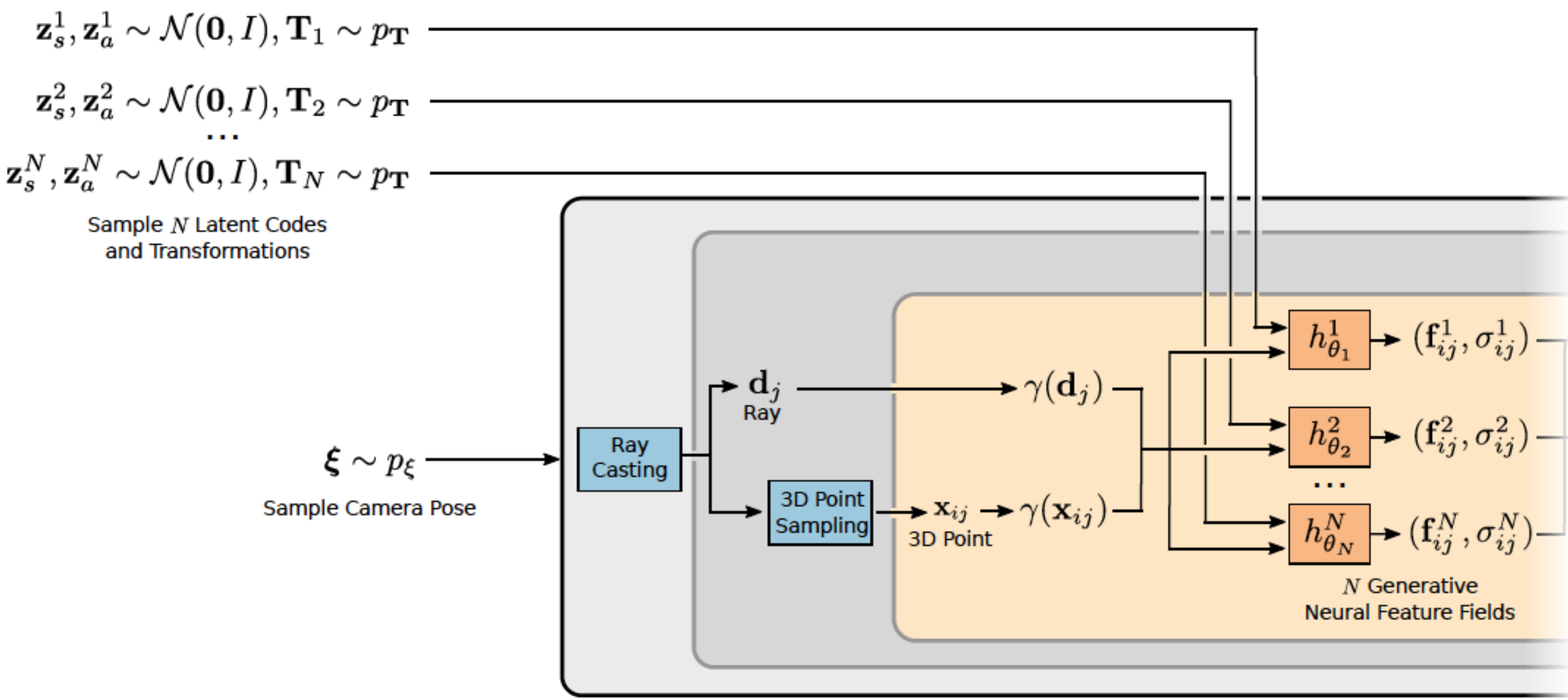
➤ Framework



Orange indicates learnable and blue non-learnable operations.

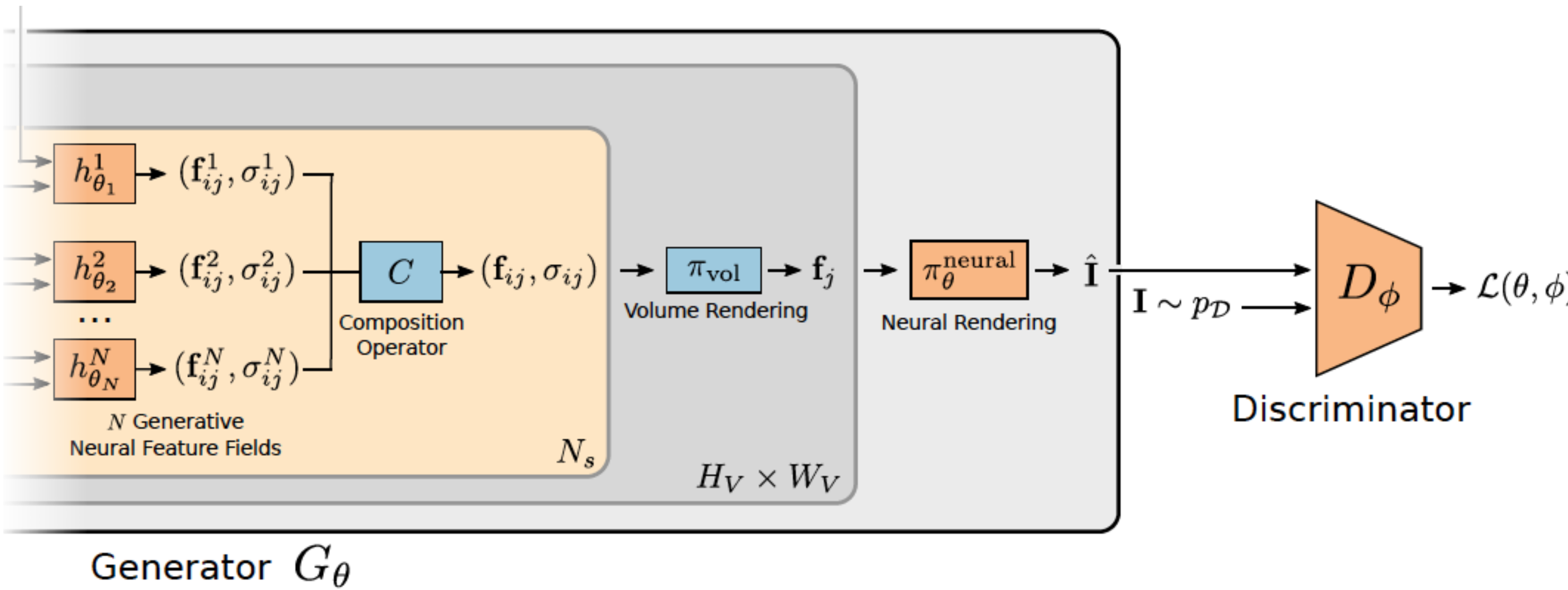
PROPOSED METHOD

➤ Framework



PROPOSED METHOD

➤ Framework



PROPOSED METHOD

- Generator

$$G_\theta(\{\mathbf{z}_s^i, \mathbf{z}_a^i, \mathbf{T}_i\}_{i=1}^N, \boldsymbol{\xi}) = \pi_\theta^{\text{neural}}(\mathbf{I}_V)$$

$$\text{where} \quad \mathbf{I}_V = \{\pi_{\text{vol}}(\{C(\mathbf{x}_{jk}, \mathbf{d}_k)\}_{j=1}^{N_s})\}_{k=1}^{H_V \times W_V}$$

- Discriminator

Layer Type	Kernel Size	Stride	Padding	Activation	Feature Dimension	Spatial Output Dimensions
Conv	4×4	2	1	LReLU	16	128×128
Conv	4×4	2	1	LReLU	32	64×64
Conv	4×4	2	1	LReLU	64	32×32
Conv	4×4	2	1	LReLU	128	16×16
Conv	4×4	2	1	LReLU	256	8×8
Conv	4×4	2	1	LReLU	512	4×4
Conv	4×4	1	0	-	1	1×1

(b) 256^2 Pixel Resolution.

PROPOSED METHOD

- Loss: GAN + R1 gradient penalty

$$
\begin{aligned}
&\mathcal{V}(\theta, \phi) = \\
&\mathbb{E}_{\mathbf{z}_s^i, \mathbf{z}_a^i \sim \mathcal{N}, \boldsymbol{\xi} \sim p_\xi, \mathbf{T}_i \sim p_T} \left[f(D_\phi(G_\theta(\{\mathbf{z}_s^i, \mathbf{z}_a^i, \mathbf{T}_i\}_i, \boldsymbol{\xi}))) \right] \\
&+ \mathbb{E}_{\mathbf{I} \sim p_{\mathcal{D}}} \left[f(-D_\phi(\mathbf{I})) - \lambda \|\nabla D_\phi(\mathbf{I})\|^2 \right]
\end{aligned}
$$

where $f(t) = -\log(1 + \exp(-t)), \lambda = 10$

OUTLINE

- Authorship
- Background
- Proposed Method
- Experimental Results
- Conclusion

EXPERIMENTAL RESULTS

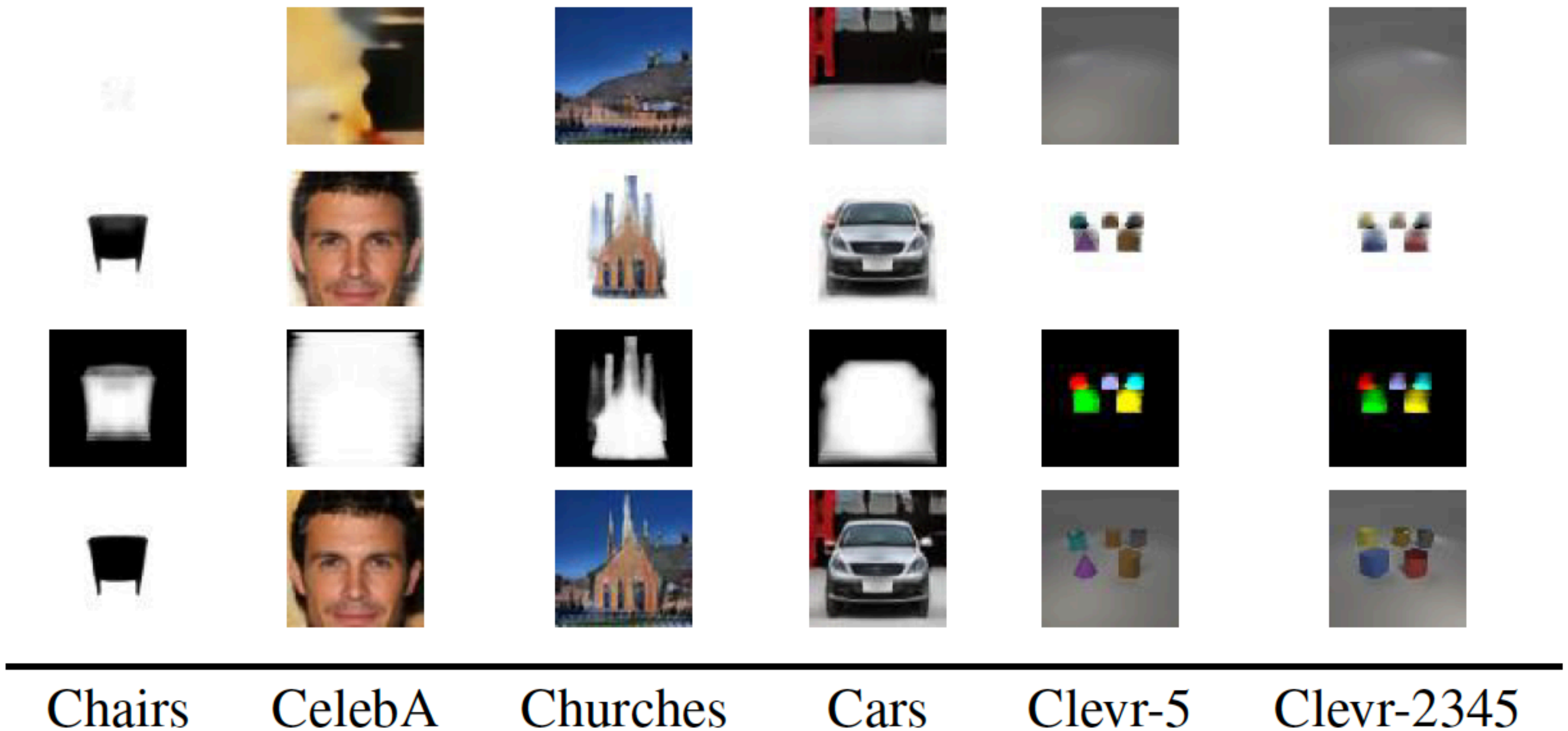
- Single-object datasets
  - Chairs, Cats, CelebA, CelebA-HQ
  - Background is purely white or only takes up a small part of the image

- More challenging single-object, real-world datasets
  - CompCars, LSUN Churches, FFHQ
  - Object is not always in the center, the background is more cluttered

- Multi-object datasets
  - Scenes with 2, 3, 4, or 5 random primitives (Clevr-N)

EXPERIMENTAL RESULTS

➤ Scene Disentanglement



- From top to bottom: only backgrounds, only objects, color-coded object alpha maps, and the final synthesized images (64×64)

EXPERIMENTAL RESULTS

- Training Progression

Figure 6: **Training Progression.** We show renderings of our model on *Clevr-2345* at 256^2 pixels after 0, 1, 2, 3, 10, and 100-thousand iterations. Unsupervised disentanglement emerges already at the very beginning of training.

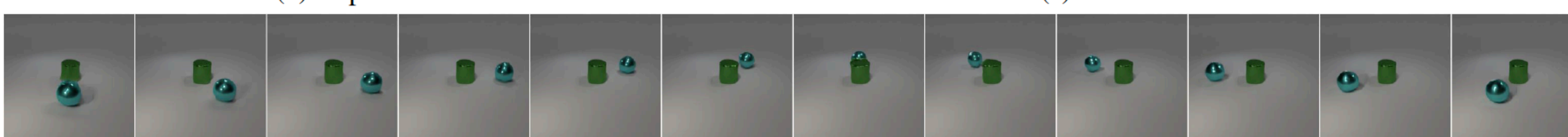
EXPERIMENTAL RESULTS

- Controllable Scene Generation

(a) Object Rotation (b) Camera Elevation

(c) Object Appearance

(d) Depth Translation (e) Horizontal Translation



(f) Circular Translation of One Object Around Another Object

EXPERIMENTAL RESULTS

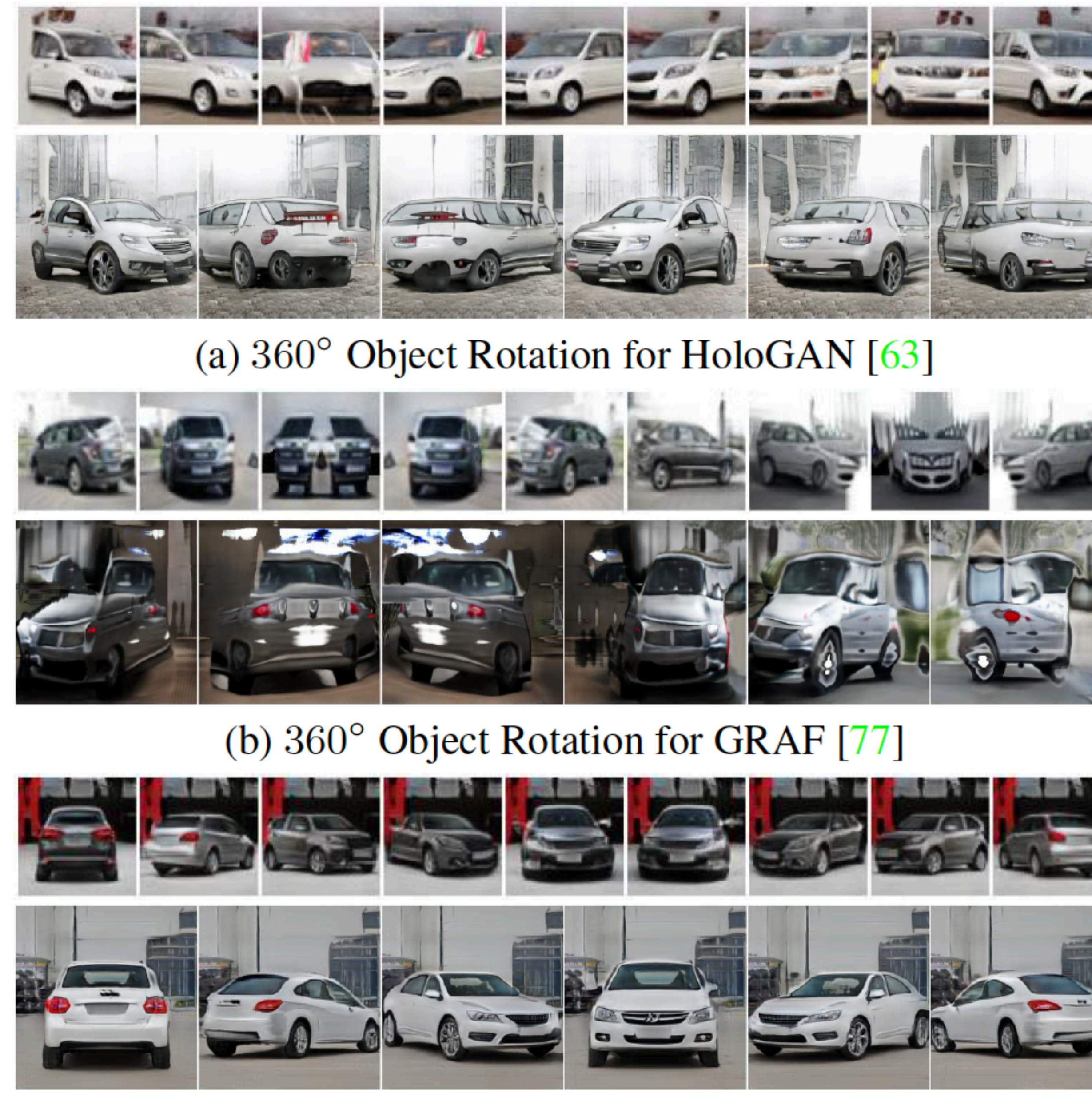
➤ Comparison to Baseline Methods

	Cats	CelebA	Cars	Chairs	Churches
2D GAN [58]	18	15	**16**	59	19
Plat. GAN [32]	318	321	299	199	242
BlockGAN [64]	47	69	41	41	28
HoloGAN [63]	27	25	17	59	31
GRAF [77]	26	25	39	34	38
Ours	**8**	**6**	**16**	**20**	**17**

Table 1: **Quantitative Comparison.** We report the FID score (↓) at 64^2 pixels for baselines and our method.

	CelebA-HQ	FFHQ	Cars	Churches	Clevr-2
HoloGAN [63]	61	192	34	58	241
w/o 3D Conv	33	70	49	66	273
GRAF [77]	49	59	95	87	106
Ours	**21**	**32**	**26**	**30**	**31**

Table 2: **Quantitative Comparison.** We report the FID score (↓) at 256^2 pixels for the strongest 3D-aware baselines and our method.



(a) 360° Object Rotation for HoloGAN [63]

(b) 360° Object Rotation for GRAF [77]

(c) 360° Object Rotation for Our Method

EXPERIMENTAL RESULTS

- Comparison to Baseline Methods

2D GAN	Plat. GAN	BlockGAN	HoloGAN	GRAF	Ours
1.69	381.56	4.44	7.80	0.68	0.41

Table 3: **Network Parameter Comparison.** We report the number of generator network parameters in million.

- Compared to GRAF, total rendering time is reduced from 110.1ms/1595.0ms to 4.8ms/5.9ms for 64×64/256×256 pixels.

EXPERIMENTAL RESULTS

- Ablation Studies

Full	-Skip	-Act.	+NN. RGB Up.	+Bi. Feat. Up.
16.16	16.66	21.61	17.28	20.68

Table 4: **Ablation Study.** We report FID (↓) on *CompCars* without RGB skip connections (-Skip), without final activation (-Act.), with nearest neighbor instead of bilinear image upsampling (+ NN. RGB Up.), and with bilinear instead of nearest neighbor feature upsampling (+ Bi. Feat. Up.).

EXPERIMENTAL RESULTS

➤ Positional Encoding

- Axis-aligned: $(\sin(2^0 t\pi), \cos(2^0 t\pi), \ldots, \sin(2^L t\pi), \cos(2^L t\pi))$

(a) 0° Rotation for Axis-Aligned Positional Encoding [61]

(b) 0° Rotation for Random Fourier Features [82]

Figure 11: **Canonical Pose.** In contrast to random Fourier features [82], axis-aligned positional encoding (1) encourages the model to learn objects in a canonical pose.

EXPERIMENTAL RESULTS

➤ Limitations

Figure 12: **Dataset Bias.** Eye and hair rotation are examples for dataset biases: They primarily face the camera, and our model tends to entangle them with the object rotation.

EXPERIMENTAL RESULTS

➤ Limitations

- Disentanglement failures

OUTLINE

- Authorship
- Background
- Proposed Method
- Experimental Results
- Conclusion

CONCLUSION

- Fast and controllable image synthesis
  - Compositional 3D scene representation
  - Disentangle individual objects without explicit supervision
  - Neural feature fields, neural renderer