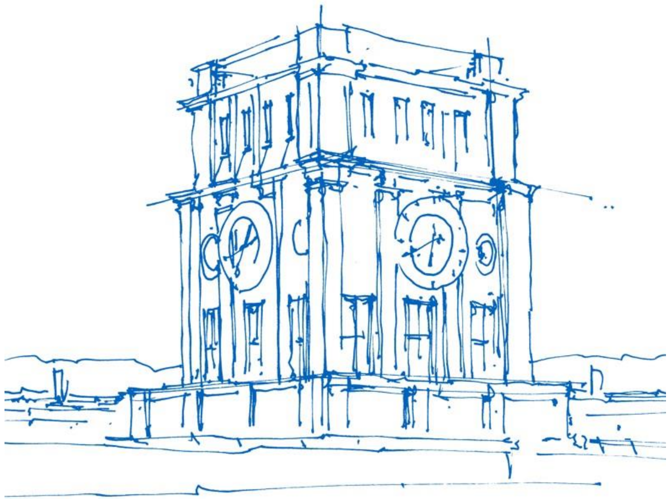
Department of Informatics
Technical University of Munich

Adversarial 3D Shape Reconstruction using Neural Fields

Zhuolun Zhou

Tutors: Lukas Koestler, Tarun Yenamandra

March 14, 2023



Uhrenturm der TUM

Motivation

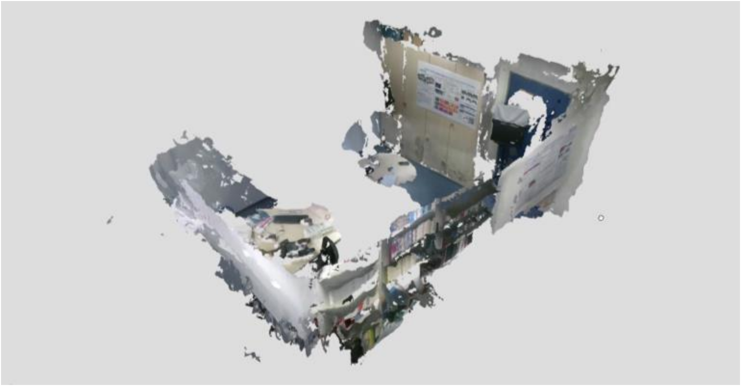
3D generation with GAN

☺ generates photo-realistic images indistinguishable from real objects
☹ not conditioned on existing objects

Source: π-GAN [1]

3D reconstruction

☺ geometrically accurate reconstruction of existing objects
☹ results are noisy, not photo-realistic to human perception



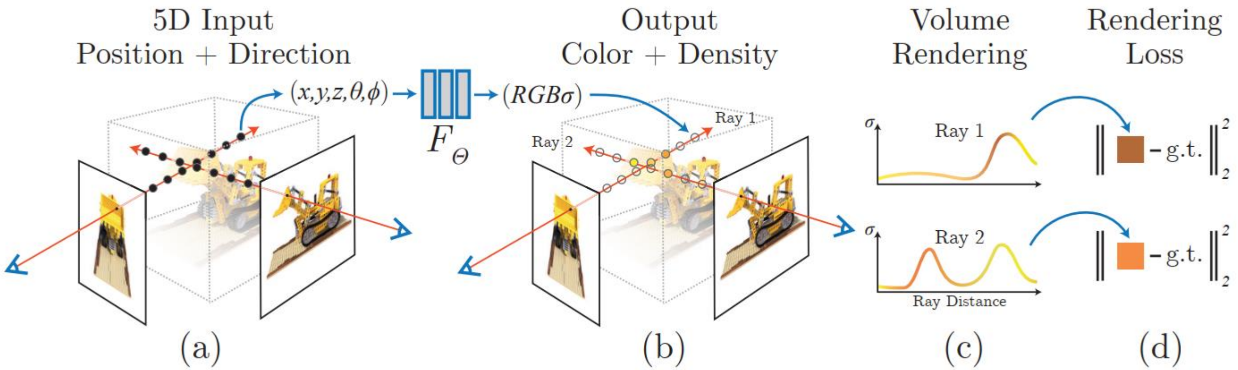
Source: TANDEM [2]

Intuition:

Improve the visual fidelity of 3D reconstruction results with GAN (“adversarial shape reconstruction”)

Background: neural fields (NeRF)

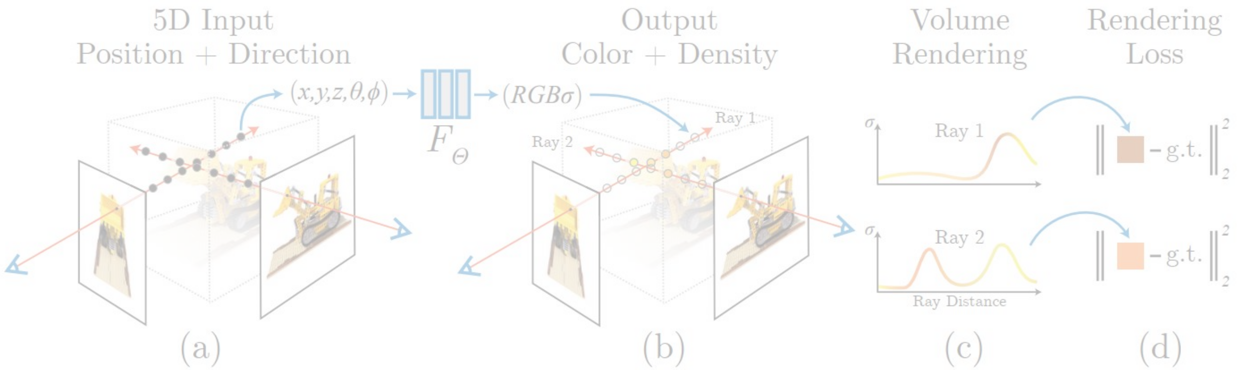
Neural radiance field



Source: NeRF [3]

Background: 3D-GAN with NeRF

Neural radiance field



Source: NeRF [3]

3D-GAN with NeRF

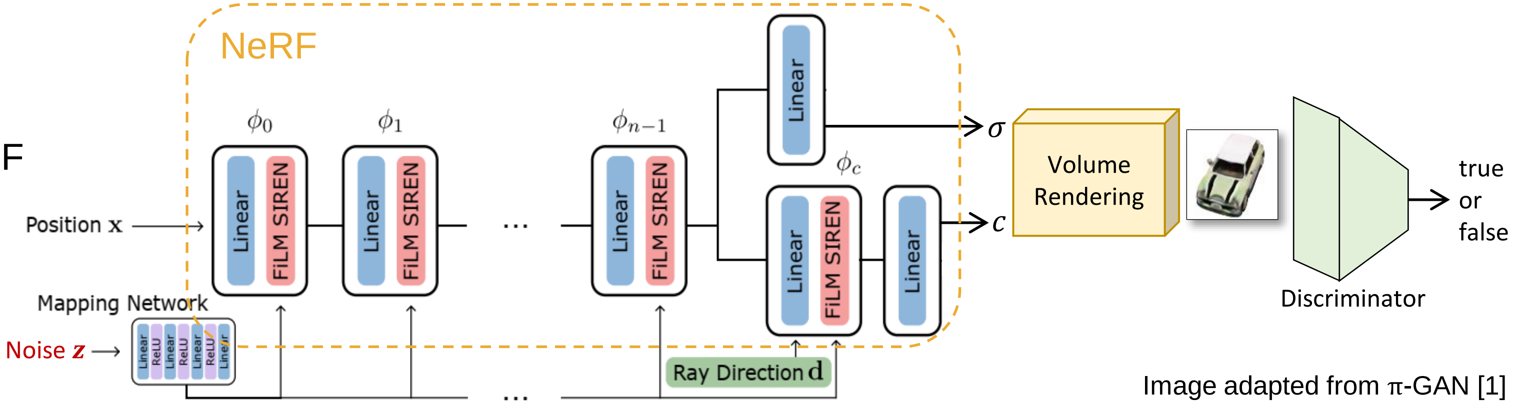
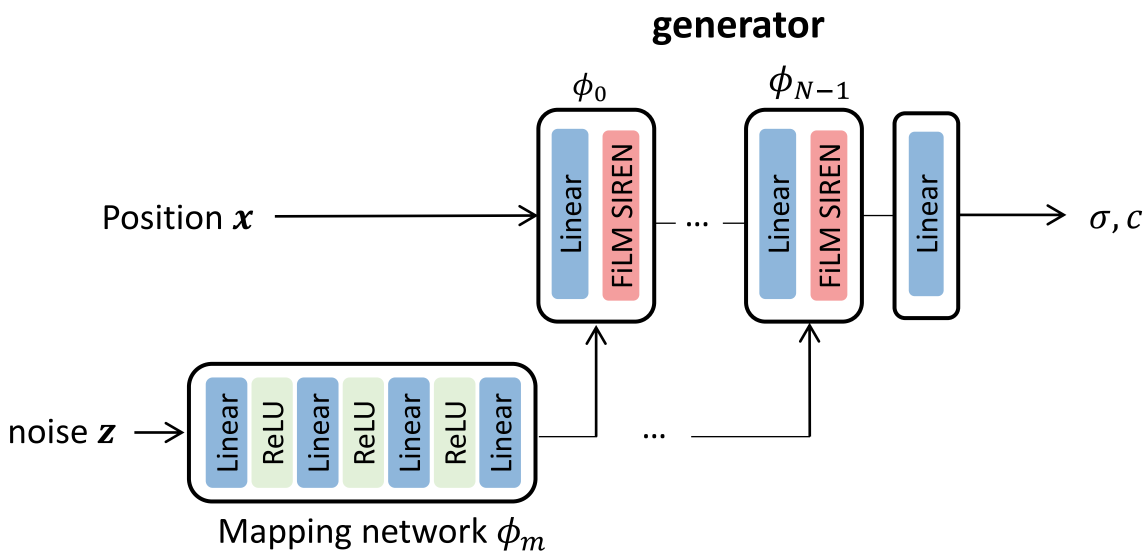


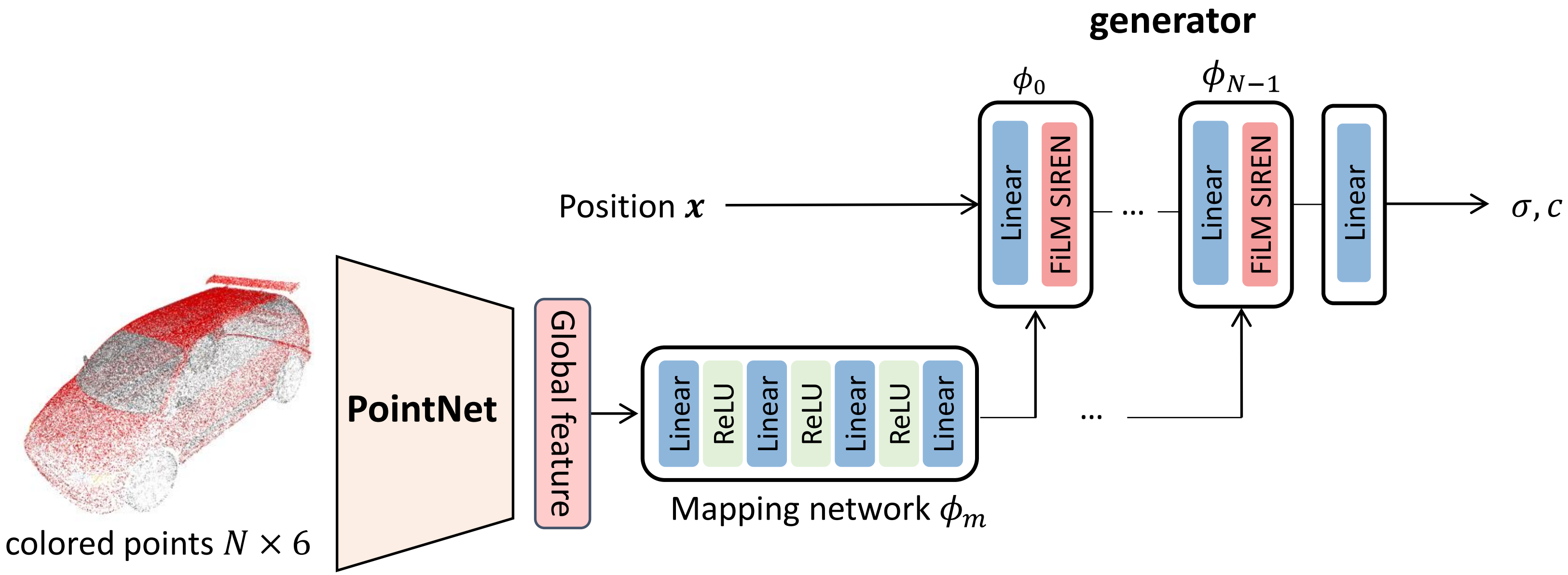
Image adapted from π-GAN [1]

Methods: point cloud encoding



Our goal:
Improve the visual fidelity of 3D reconstruction results (e.g. point clouds or coarse voxel grids)

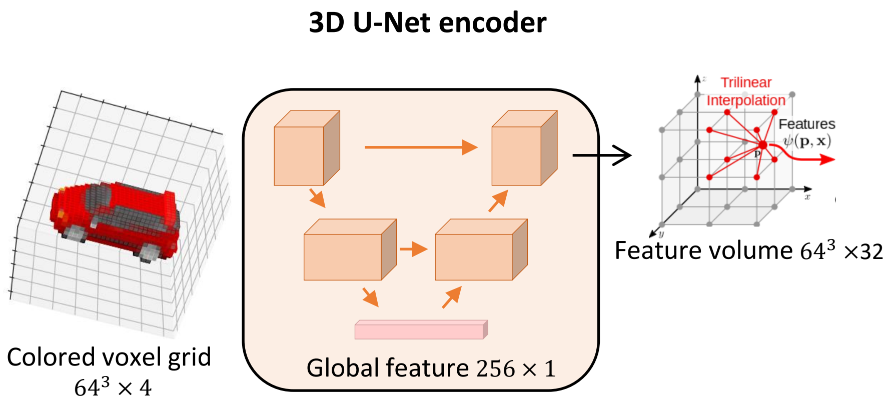
Methods: point cloud encoding



Our goal:

Improve the visual fidelity of 3D reconstruction results (e.g. point clouds or coarse voxel grids)

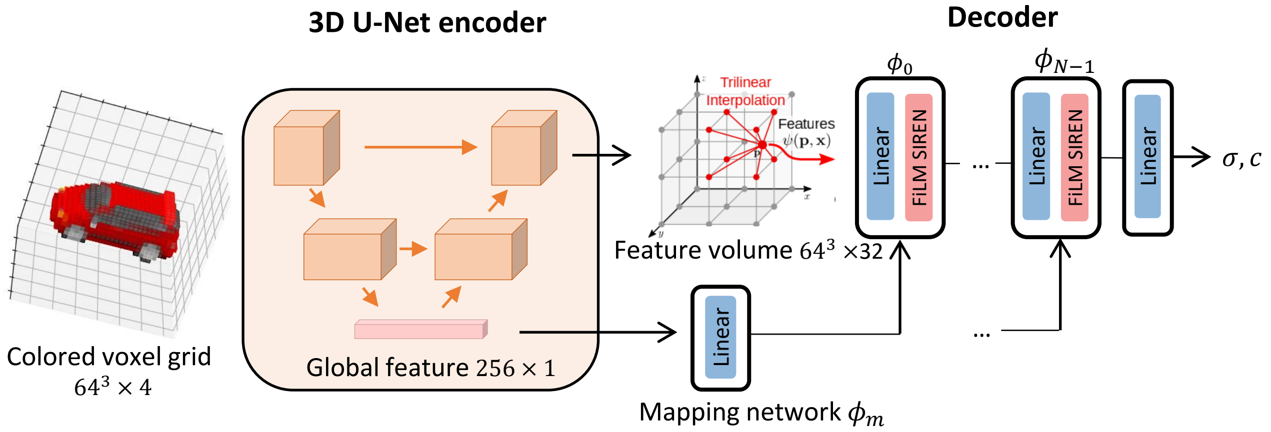
Methods: feature volume



Our goal:

Improve the visual fidelity of 3D reconstruction results (e.g. point clouds or coarse voxel grids)

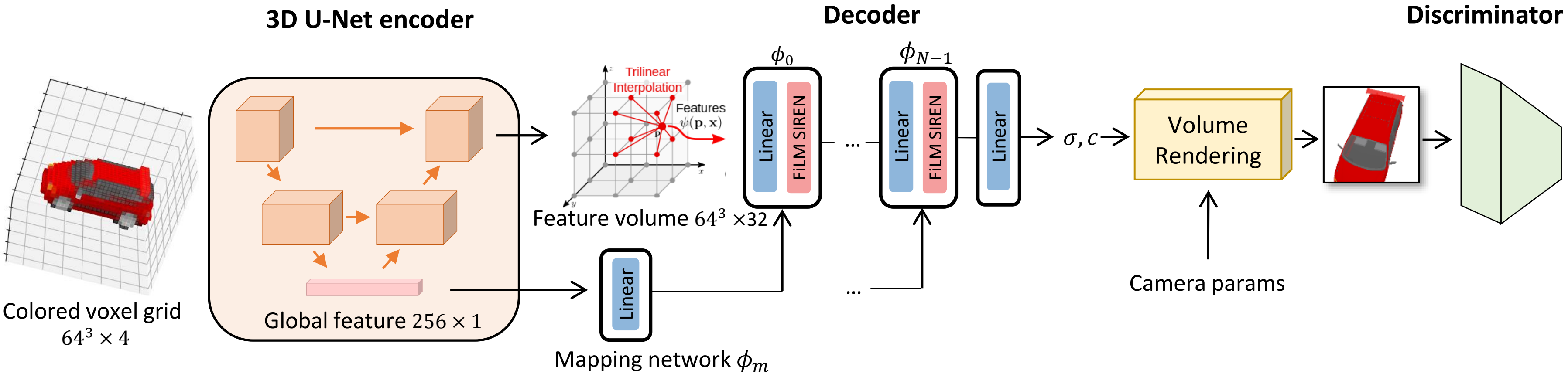
Methods: feature volume



Our goal:

Improve the visual fidelity of 3D reconstruction results (e.g. point clouds or coarse voxel grids)

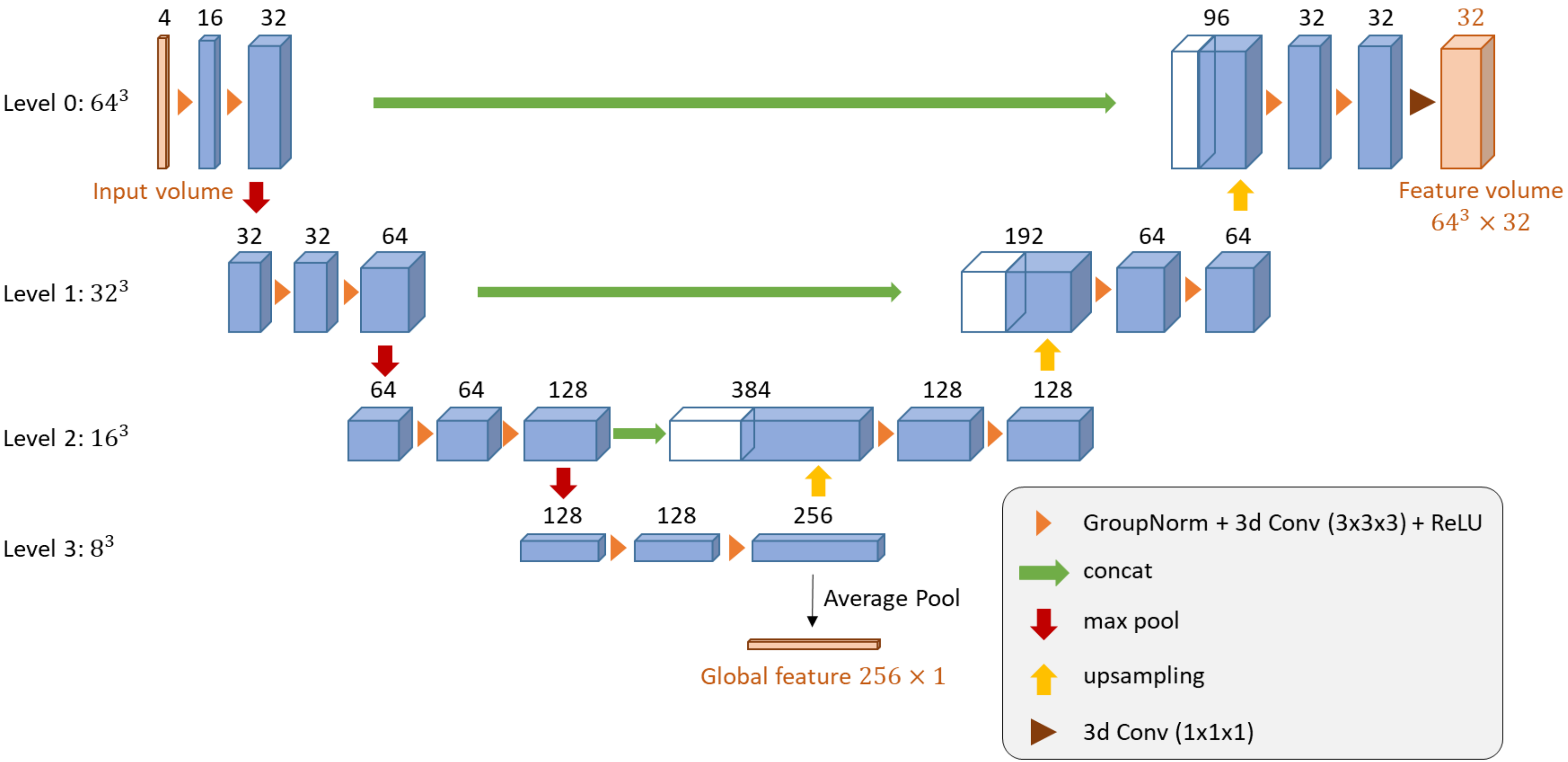
Methods: feature volume



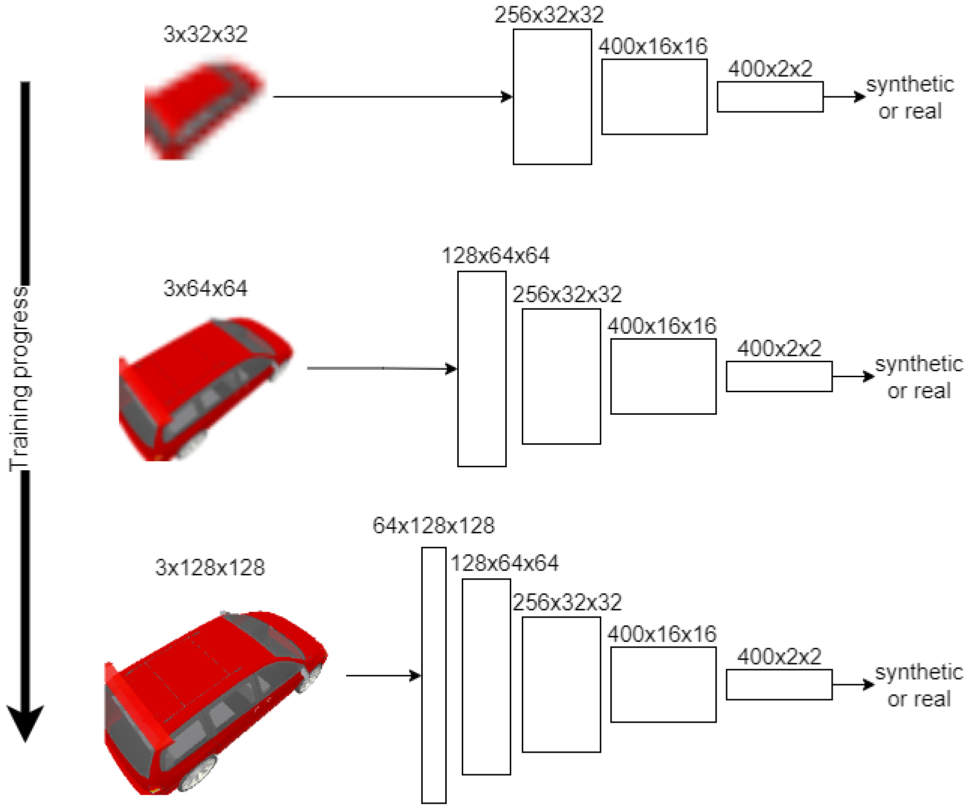
Our goal:

Improve the visual fidelity of 3D reconstruction results (e.g. point clouds or coarse voxel grids)

Methods: 3D U-Net encoder



Methods: progressive discriminator



Methods: losses

- GAN loss: $$\mathcal{L}(\theta_D, \theta_\Phi, \theta_U) = \mathbb{E}_{V \sim p_V, \xi \sim p_\xi} [f(D(\Phi(U(\mathbf{V}), \xi)))] + \mathbb{E}_{I \sim p_I} \left[f(-D(I)) + \lambda |\nabla D(I)|^2 \right]$$

  where $f(u) = -\log(1 + \exp(-u))$
  U, Φ, D: encoder, decoder, discriminator
  $\boldsymbol{V}$: input voxel grids
  ξ: camera parameters
  I: real image
  $\Phi(U(\boldsymbol{V}), \xi)$: generated image at pose ξ
  λ: weight of R1 regularization

- Photometric loss: $$\mathcal{L}(\theta_\Phi, \theta_U; \mathbf{V}, \xi, I_\xi) = \frac{1}{H \times W \times 3} \|\Phi(U(\mathbf{V}), \xi) - I_\xi\|_F^2$$

  where I_ξ: real image of the object at pose ξ
  $\Phi(U(\boldsymbol{V}), \xi)$: generated image at pose ξ

Experiments: dataset and metrics

Dataset:

ShapeNet car, plane and chair

image and depth map point cloud voxel grid

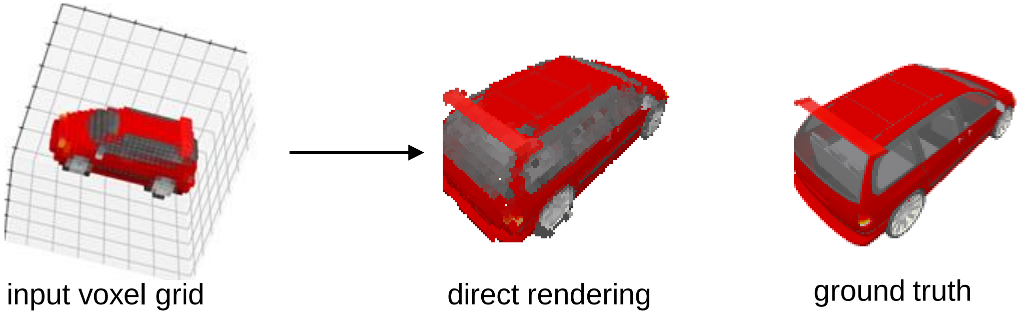
Metrics:

- Fréchet Inception Distance (FID) ↓ [7]
- object FID (oFID) ↓
- LPIPS [8] ↓
- Peak Signal-to-Noise Ratio (PSNR)↑

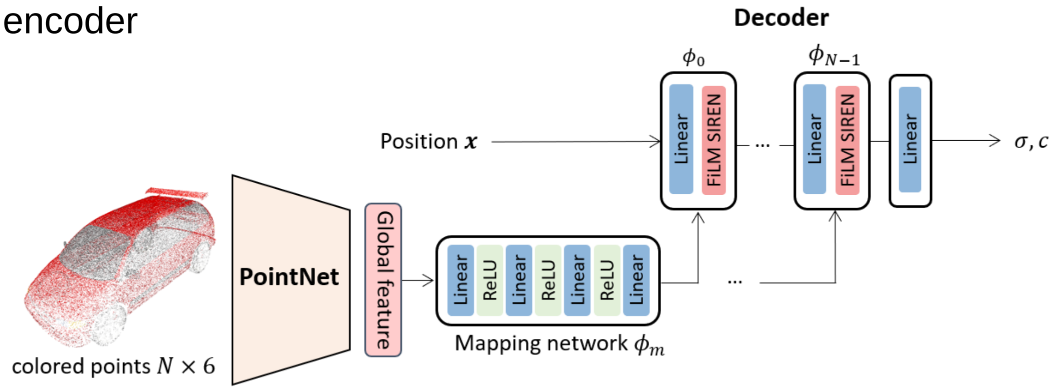
Perceptual similarity

Experiments: baseline methods

- Voxel surface rendering



- PointNet [4] encoder



Experiments: quantitative results

	FID↓	oFID↓	LPIPS↓	PSNR↑
Voxel Surface Rendering	75.75	3.88	0.167	17.68
PointNet Encoder	181.95	6.24	0.357	17.14
Ours w/ discri.	**46.27**	**3.81**	0.138	20.26
Ours w/o discri.	56.11	3.84	**0.123**	**23.64**

(a) cars

	FID↓	oFID↓	LPIPS↓	PSNR↑
Voxel Surface Rendering	50.37	**4.22**	0.198	19.44
PointNet Encoder	191.96	7.05	0.437	19.97
ours w/ discri.	29.87	4.71	0.151	23.82
ours w/o discri.	**26.66**	**4.22**	**0.095**	**28.02**

(b) chairs

	FID↓	oFID↓	LPIPS↓	PSNR↑
Voxel Surface Rendering	45.04	**3.88**	0.166	20.54
PointNet Encoder	190.76	6.29	0.248	24.81
ours w/ discri.	44.14	4.32	0.128	25.60
ours w/o discri.	**31.24**	3.93	**0.078**	**29.90**

(c) planes

Results on test set (unseen objects), 64^3 input voxel resolution

	FID↓	oFID↓	LPIPS↓	PSNR↑
Voxel Surface Rendering	126.65	**4.61**	0.246	14.80
ours w/o discri.	**112.90**	4.75	**0.197**	**20.72**

Results on test set (unseen objects) of cars, 32^3 input voxel resolution

Experiments: qualitative results

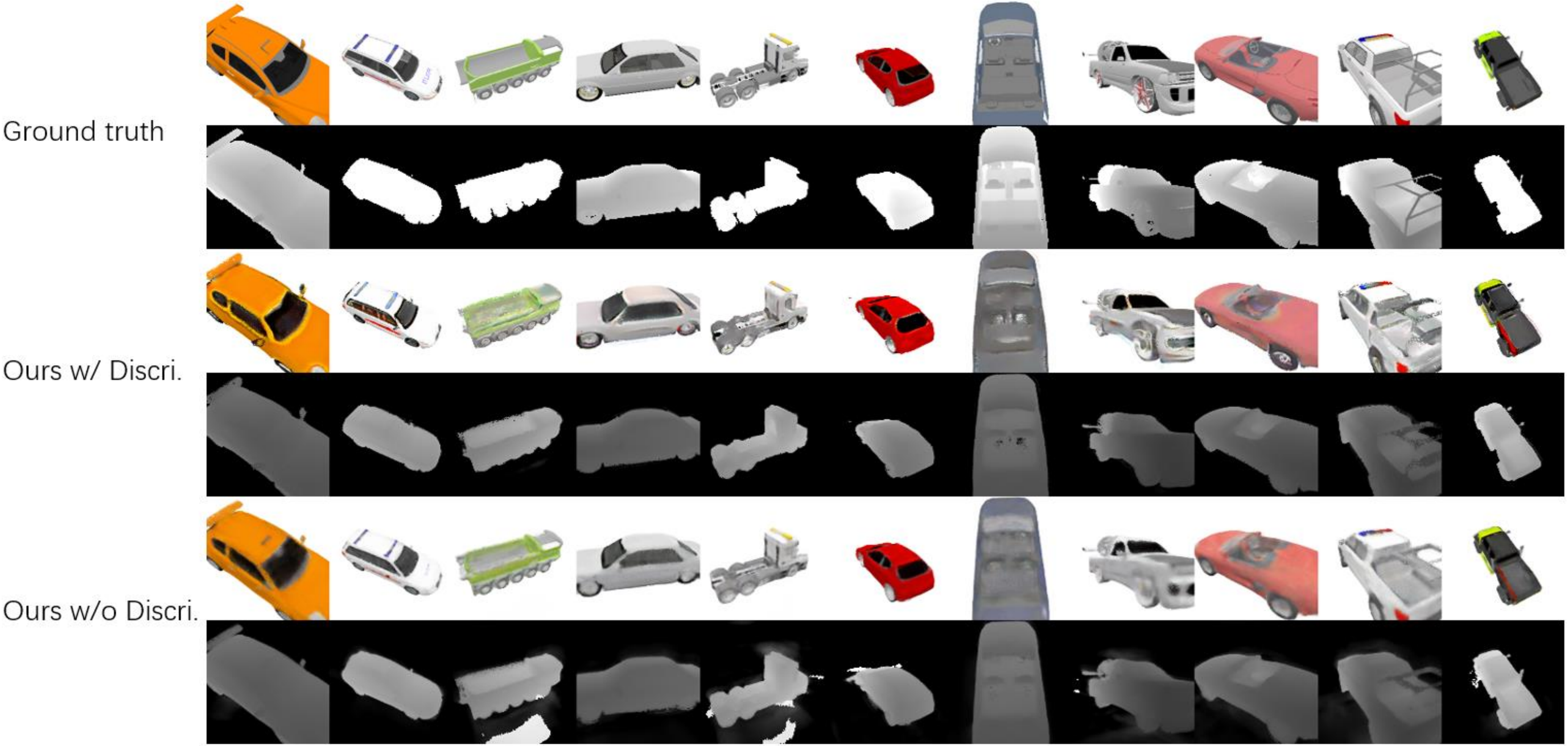
Experiments: qualitative results

Experiments: qualitative results

Experiments: more results on test set

Experiments: randomly chosen results on test set

Experiments: effects of discriminator



Experiments: geometry

Input voxel

Output geometry

Output geometry

Experiments: geometry

Input voxel

Output geometry

Output geometry

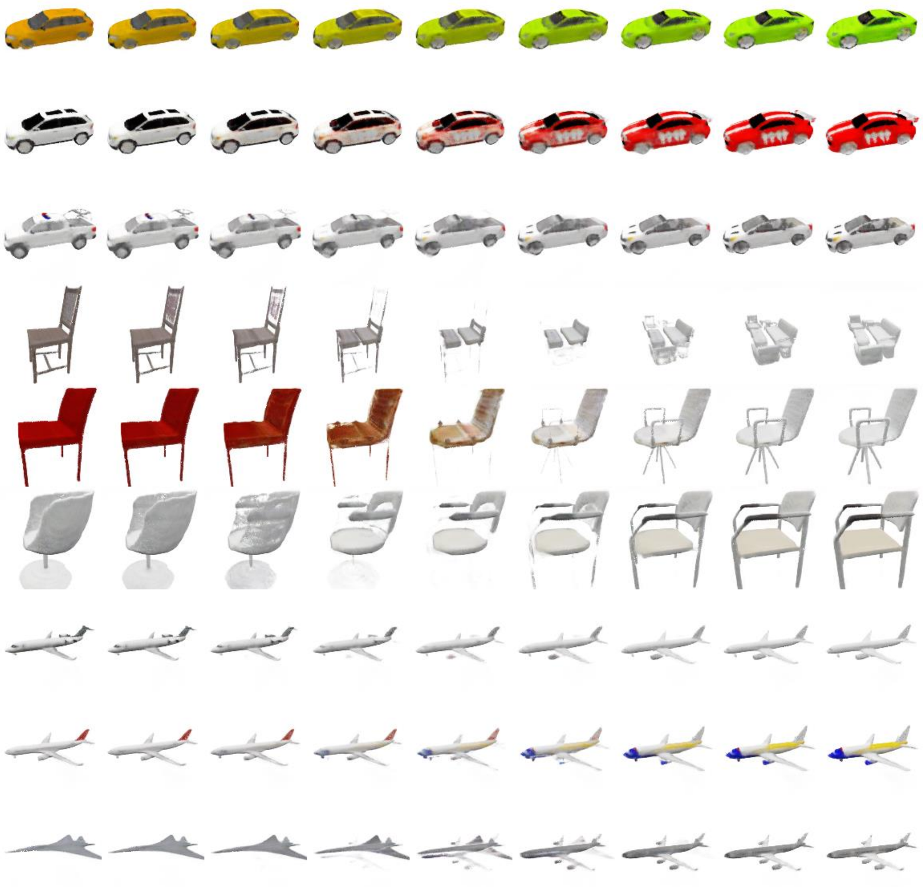
Experiments: geometry

Input
voxel

Output
geometry

Output
geometry

Experiments: interpolating the latent space



Experiments: ablation study

each column: a design choice

each row: one experiment

Feature encoding	3D U-Net level	Depth loss	Voxel Res.	# SIREN	FID↓	oFID↓	LPIPS↓	PSNR↑
FP + g	4	✗	64	4	72.56	4.18	0.130	**23.40**
FV	4	✗	64	4	122.06	5.26	0.194	21.71
FV w/ skip-layer	4	✗	64	4	92.71	4.50	0.153	22.70
FV + g	5	✗	64	4	74.30	4.20	0.130	23.24
FV + g	4	✓	64	4	75.48	4.22	0.131	23.26
FV + g	4	✗	32	4	132.06	5.07	0.201	20.96
FV + g	4	✗	32 → 128	4	190.02	7.32	0.481	15.35
FV + g	4	✗	64	2	75.19	4.22	0.131	23.30
FV + g	4	✗	64	8	**69.00**	4.17	**0.128**	23.35
FV + g	4	✗	64	4	71.30	**4.09**	**0.128**	23.36

adopted setting →

FV: feature volume
g: global feature
FP: feature pyramid

input voxel resolution

number of layers in the decoder

Experiments: ablation study

discriminator style	FID↓	oFID↓	LPIPS↓	PSNR↑
no conditioning	**53.06**	4.10	0.144	21.42
Input concat	60.60	4.18	0.140	21.30
Projection	53.86	**4.09**	**0.137**	21.45
✗	72.70	4.40	0.157	**22.95**

Ablation study on conditioning the discriminator

Contributions

- We proposed a feature volume for local encoding and a feature vector for global encoding of 3D objects to condition the neural radiance field
- We introduced the adversarial loss in a GAN framework into 3D reconstruction
- We implemented a conditioned neural radiance field to render realistic images from low-quality geometry input

Future work

- Experiment on real-world dataset (e.g. CO3D [5]) without canonical poses
- Global + local encoding for point cloud

Thanks for your attention!

Questions?

References

[1] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. “pi-GAN: Periodic implicit generative adversarial networks for 3d-aware image synthesis.” In: *CVPR* 2021

[2] L. Koestler, N. Yang, N. Zeller, and D. Cremers. “TANDEM: Tracking and Dense Mapping in Real-time using Deep Multi-view Stereo.” In: CoRL 2021

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.” In: *ECCV* 2020

[4] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. “PointNet: Deep learning on point sets for 3d classification and segmentation.” In: *CVPR* 2017

[5] J. Reizenstein, R. Shapovalov, P. Henzler, L. Sbordone, P. Labatut, and D. Novotny. “Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction.” In: *ICCV* 2021.

[6] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. “Rethinking the inception architecture for computer vision.” In: CVPR 2016

[7] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. “GANs trained by a two time-scale update rule converge to a local nash equilibrium.” In: NeurIPS 2017

[8] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric.” In: CVPR 2018

Backup: metrics

- FID (Frechnet Inception Distance) [7]:

$$\begin{aligned}\mathrm{FID}(S, S') &= d_F(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu', \Sigma')) \\ &= \|\mu - \mu'\|_2^2 + \mathrm{trace}\left(\Sigma + \Sigma' - 2(\Sigma\Sigma')^{\frac{1}{2}}\right)\end{aligned}$$

where S and S' are two image datasets, μ and Σ are the mean and covariance of the pool3 layer of the Inceptionv3 [6] model over S.

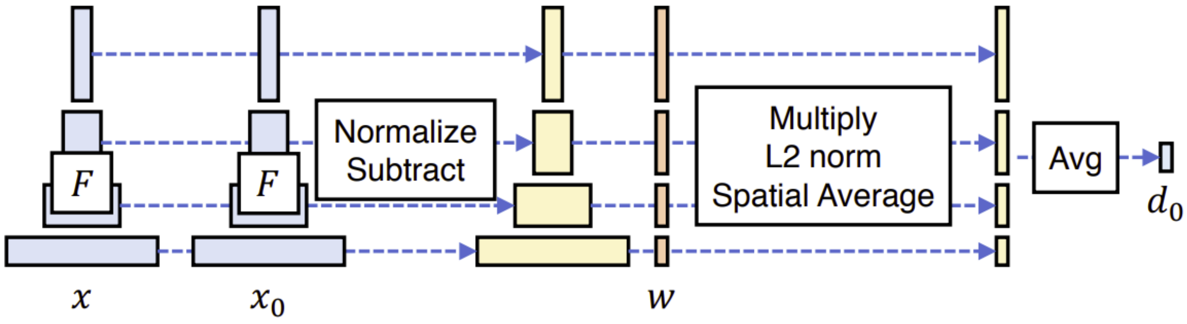
- oFID (object FID): averaged FID for each object

$$\mathrm{oFID}(S, S') = \frac{1}{|\mathcal{Y}|} \sum_{y \in \mathcal{Y}} \mathrm{FID}(S_y, S'_y)$$

where S_y denotes the image subset of object y.

Backup: metrics

- LPIPS [8]: the similarity between the activations of two image patches for a pre-trained neural network



Source: [8]

- PSNR (Peak Signal-to-Noise Ratio):

$$\mathrm{PSNR}(I, I') = 10 \log_{10} \frac{\mathrm{MAX}}{\mathrm{MSE}(I, I')} = -10 \log_{10} \frac{\|I - I'\|_F^2}{H \times W \times 3}$$

Backup: discriminator conditioning

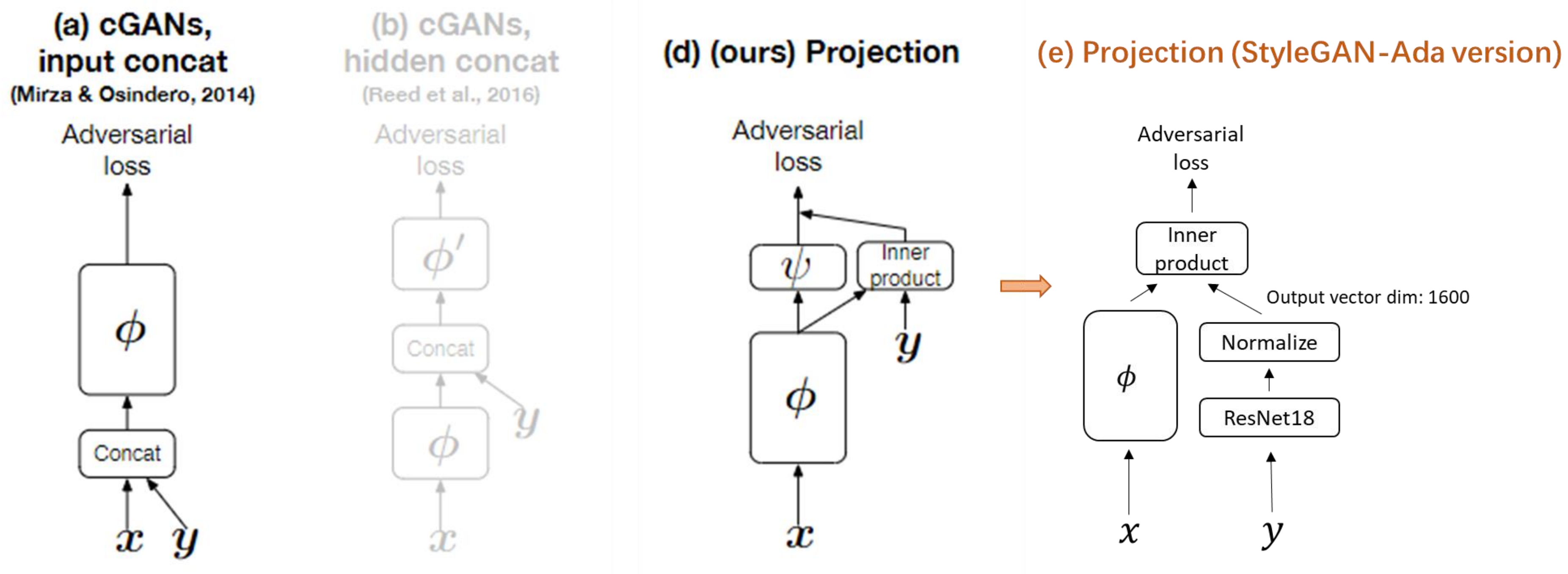
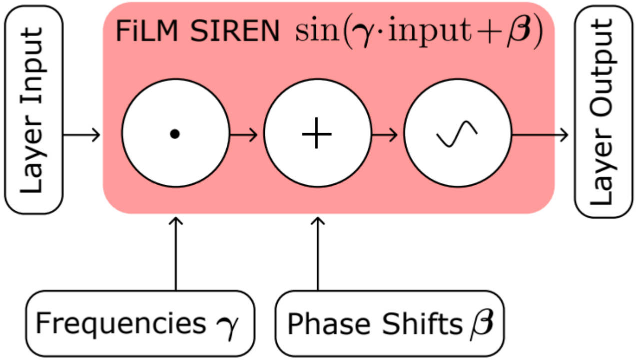


Image taken from *cGANS with projection discriminator, Miyato et.al.*

Backup: FiLM-ed SIREN



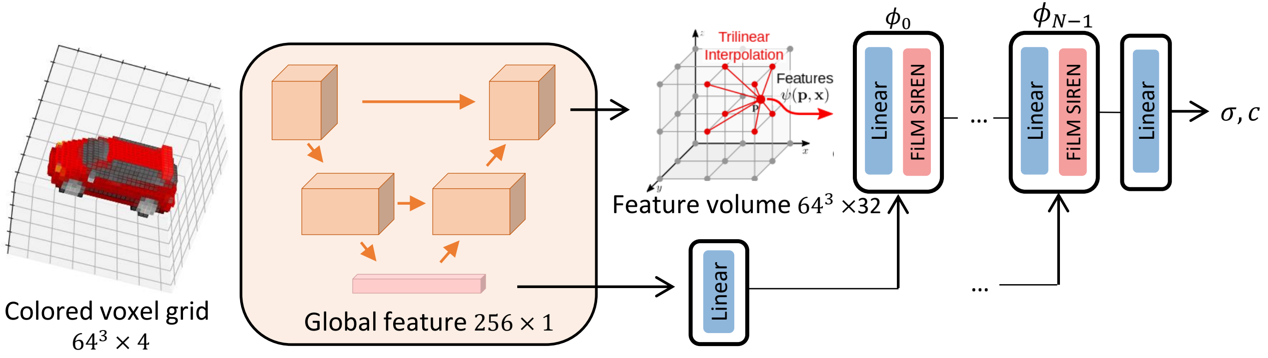
Source: [1]

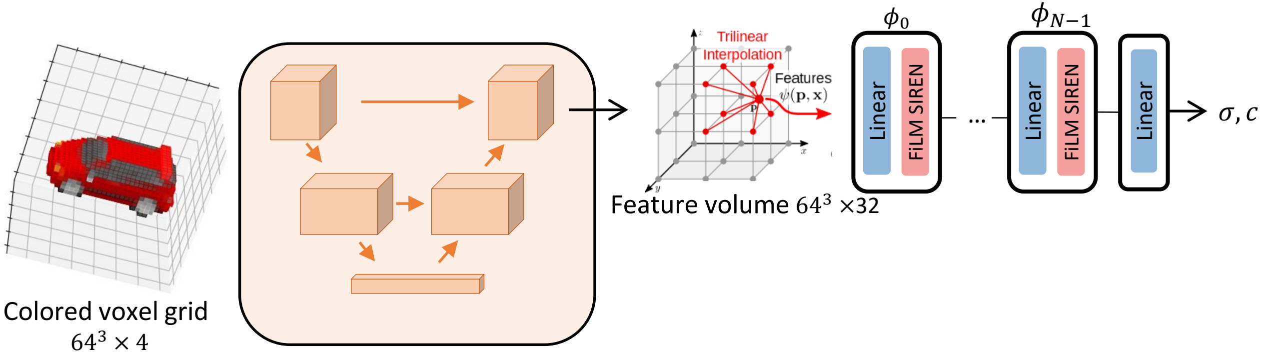
Conditioning	Architecture	
	ReLU P.E.	Sine
Concatenation	32.0	21.6
Mapping Network	26.8	**5.15**

Table 2: FID scores on CelebA @ 64×64, when comparing network architectures with different activation functions and conditioning methods.

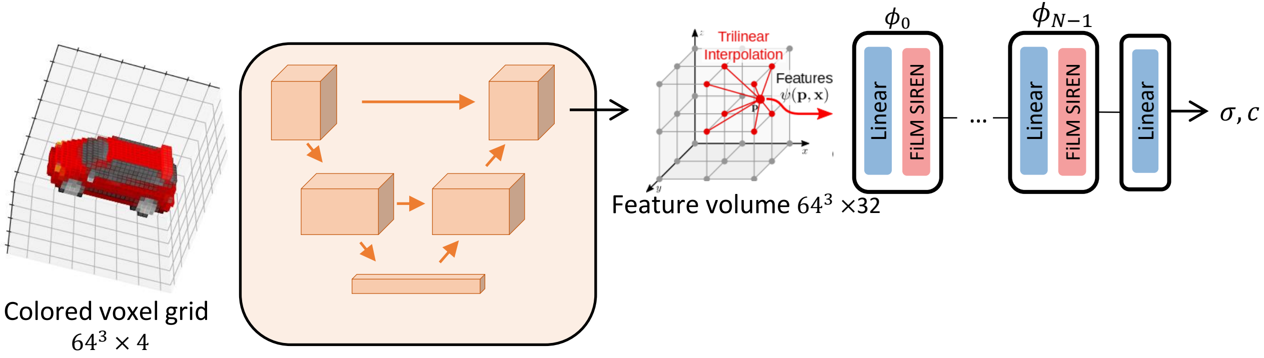
Source: [1]

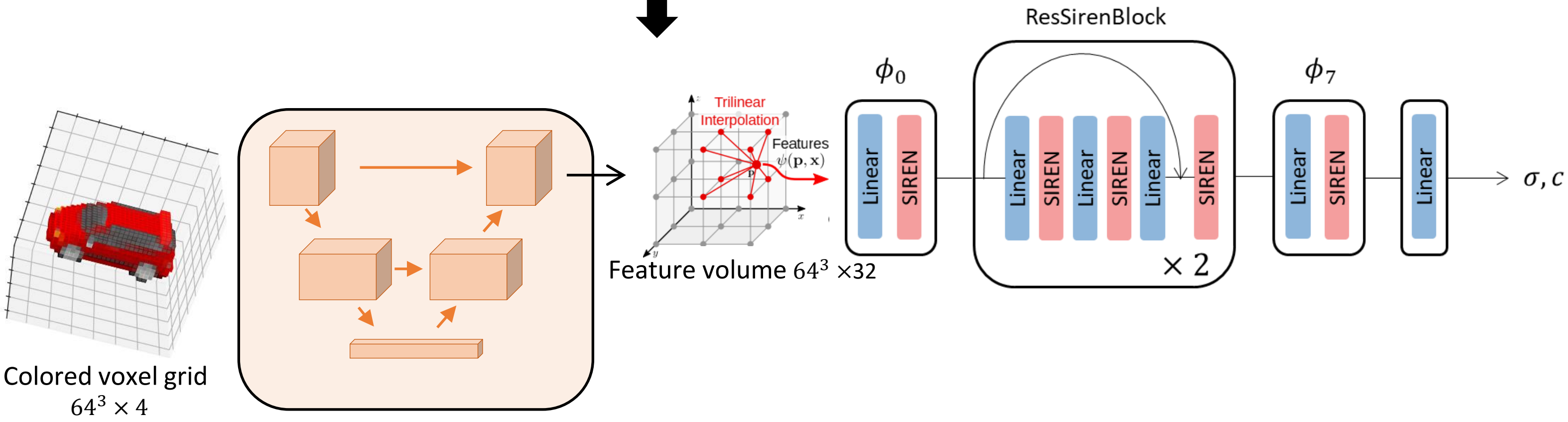
Backup: FV w/ global feature



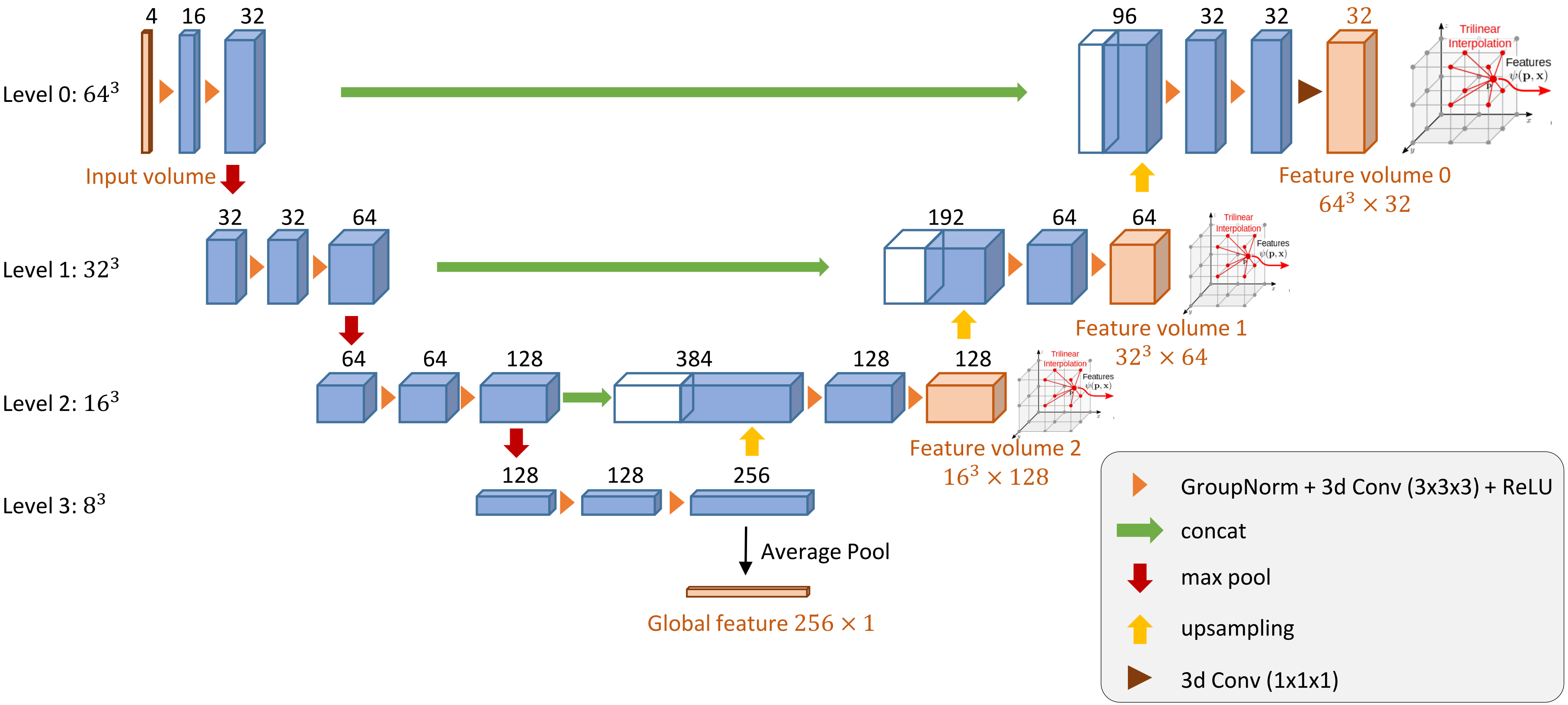


Backup: FV w/ global feature, skip-layer





Back up: feature pyramid



Backup: adversarial loss

Mi, Lu, et al. "im2nerf: Image to neural radiance field in the wild." *arXiv preprint arXiv:2209.04061* (2022).

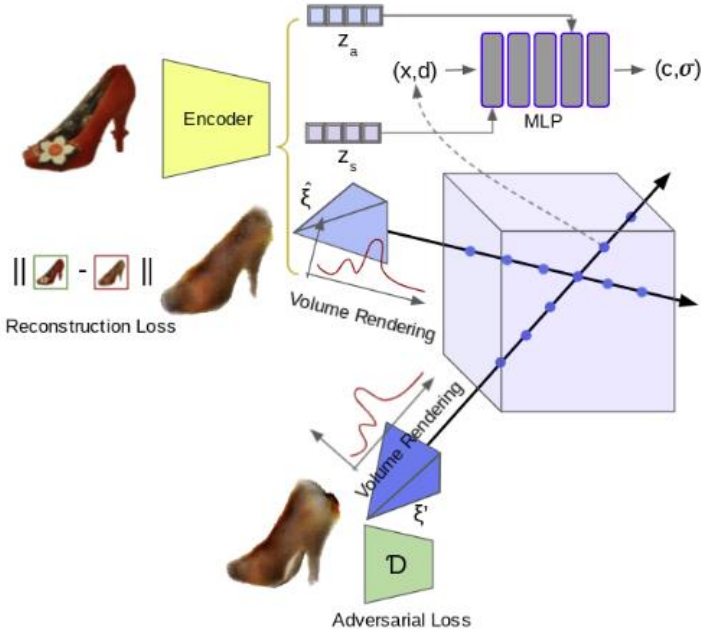
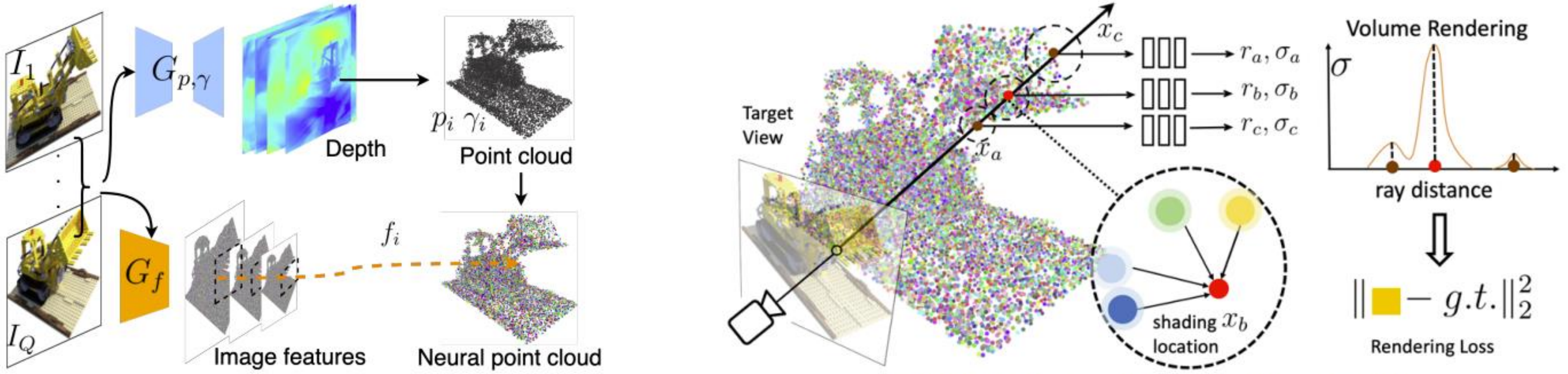


Figure 2. **Overview of our method.** Given an input image, the encoder predicts a shape z_s and an appearance code z_a and estimates the pose of the camera $\hat{\xi}$ that captures the input image. The decoder conditions a NeRF on the predicted shape and appearance representations and uses volume rendering to generate images from novel views. In addition to using a photometric reconstruction loss for input view, we apply an adversarial loss on rendered images from novel views. In addition, we further constrain the problem by using a scene box, cycle camera pose consistency and object symmetry (for symmetric object categories).

Backup: point-nerf



(a) Neural Point Generation. (b) Point-NeRF Representation with Volume Rendering.

Figure 2. Overview of Point-NeRF. (a) From multi-view images, our model generates depth for each view by using a cost volume-based 3D CNNs $G_{p,\gamma}$ and extract 2D features from the input images by a 2D CNN G_f. After aggregating the depth map, we obtain a point-based radiance field in which each point has a spatial location p_i, a confidence γ_i and the unprojected image features f_i. (b) To synthesize a novel view, we conduct differentiable ray marching and compute shading only nearby the neural point cloud (e.g., x_a, x_b, x_c). At each shading location, Point-NeRF aggregates features from its K neural point neighbors and compute radiance r and volume density σ then accumulate r using σ. The entire process is end-to-end trainable and the point-based radiance field can be optimized with the rendering loss.

Xu, Qiangeng, et al. "Point-nerf: Point-based neural radiance fields." *CVPR* 2022

Backup: control-nerf

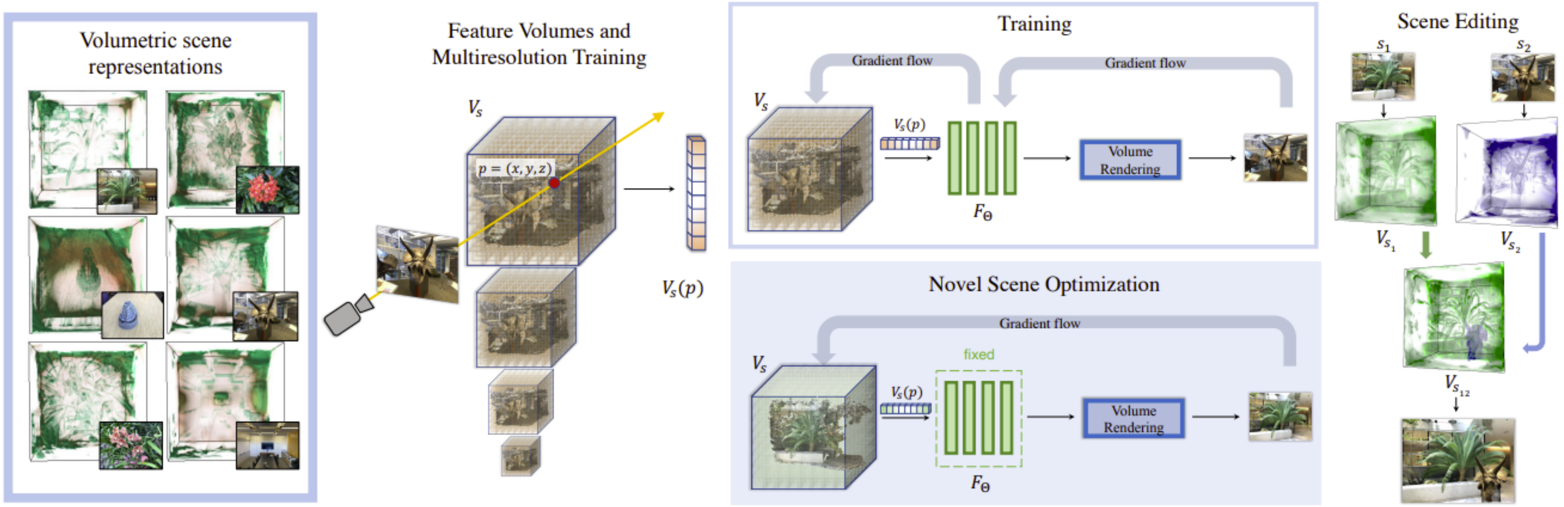


Figure 2. Our method learns a volumetric representations for multiple scenes simultaneously. Left in the figure we show visualizations of the learned feature volumes. We query the volume along the ray and predict color and density based on the obtained features. The pixel color is derived using volume rendering, similar to [23]. At training time the volume and the rendering network are trained jointly. For novel scenes, the rendering network is fixed and only the scene volume is optimized. As shown on the right, these volumes can be edited and mixed and for the purpose of scene editing.

Lazova, Verica, et al. "Control-nerf: Editable feature volumes for scene rendering and manipulation." *WACV* 2023