Single-Stage Diffusion NeRF: A Unified Approach to 3D Generation and Reconstruction

Hansheng Chen,[1,*] Jiatao Gu,[2] Anpei Chen,[3] Wei Tian,[1] Zhuowen Tu,[4] Lingjie Liu,[5] Hao Su[4]

[1]Tongji University [2]Apple [3]ETH Zürich
[4]University of California, San Diego [5]University of Pennsylvania

ICCV 2023

Presenter: Hao Wang

Advisor: Prof. Chia-Wen Lin

Outline

- Introduction
- Related Work
- Framework
- Method
- Experiment
- Conclusion

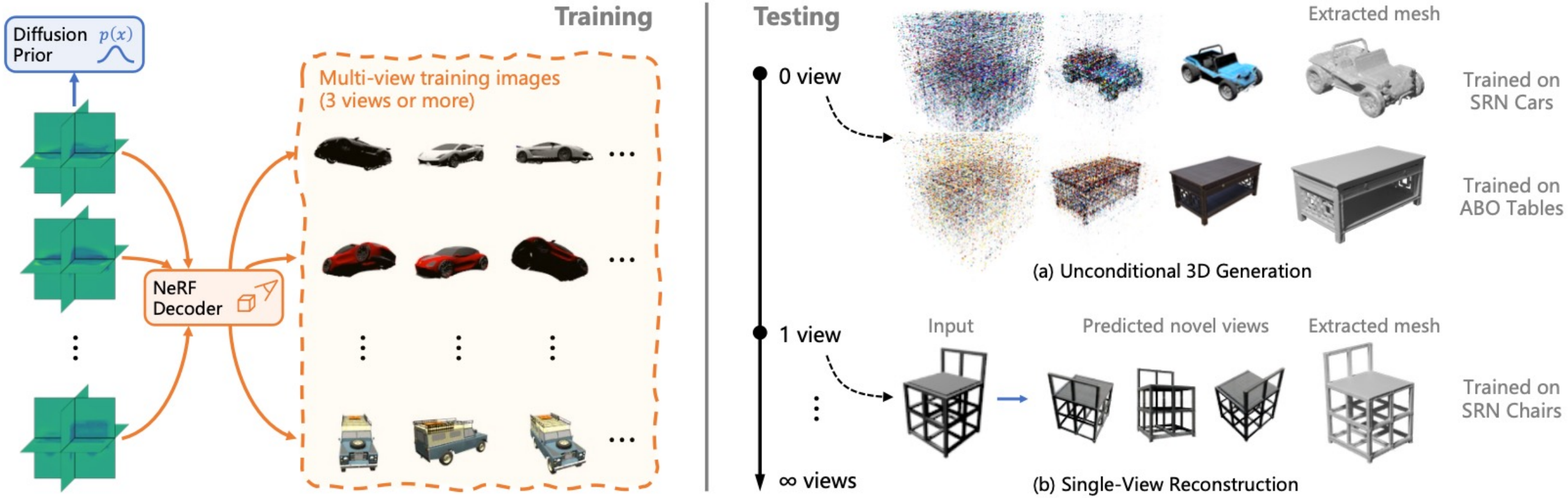
Introduction

- Propose a unified approach to all-round performance in **unconditional 3D generation** and **image-based reconstruction**.

- Propose a novel single-stage training paradigm that **jointly learns NeRF reconstruction** and **diffusion model** from multi-view images of a large number of objects and this enables **training on as sparse as three views per scene**.

- A **guidance-finetuning** sampling scheme is developed to **exploit the learned diffusion priors** for 3D reconstruction from arbitrary number of views at test time.

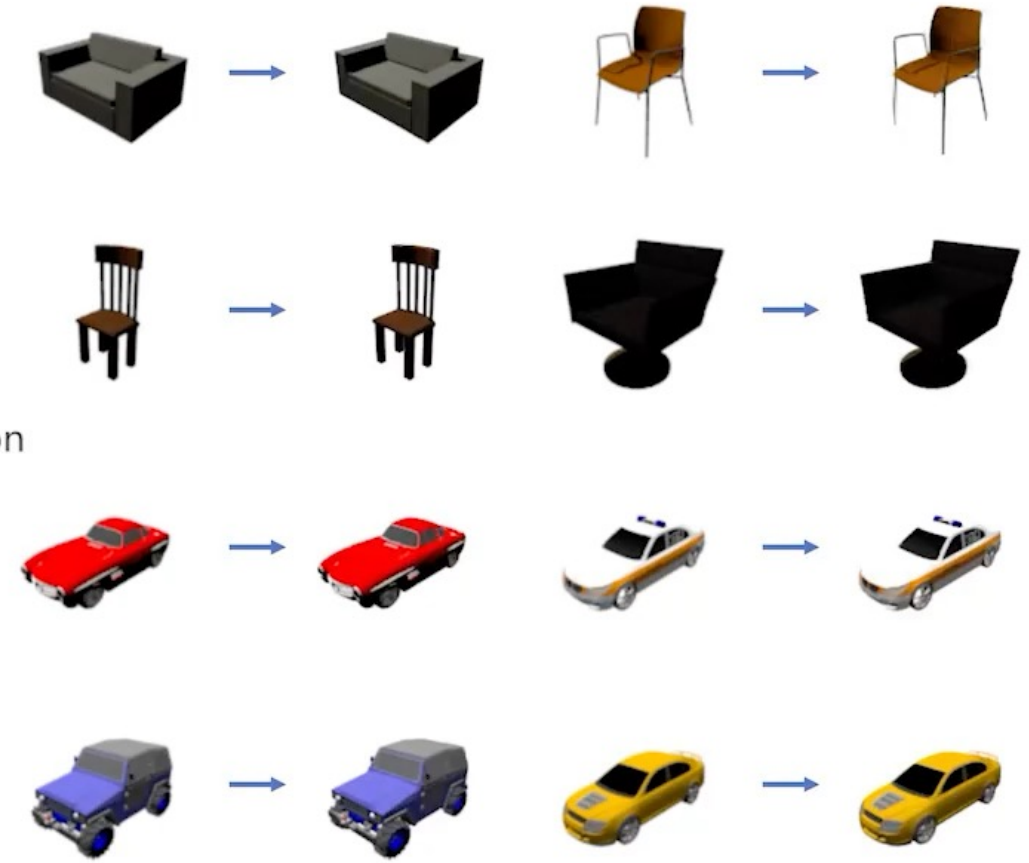
Introduction



Introduction

Unconditional Generation

Single-View Reconstruction



Outline

- Introduction
- Related Work
- Framework
- Method
- Experiment
- Conclusion

Related Work

- CodeNeRF: Disentangled Neural Radiance Fields for Object Categories
  - ICCV 2021

- DiffRF: Rendering-guided 3D Radiance Field Diffusion
  - CVPR 2023 Highlight

Related Work – CodeNeRF

CodeNeRF: Disentangled Neural Radiance Fields for Object Categories

Wonbong Jang Lourdes Agapito
Department of Computer Science
University College London
{ucabwja,l.agapito}@ucl.ac.uk

ICCV 2021

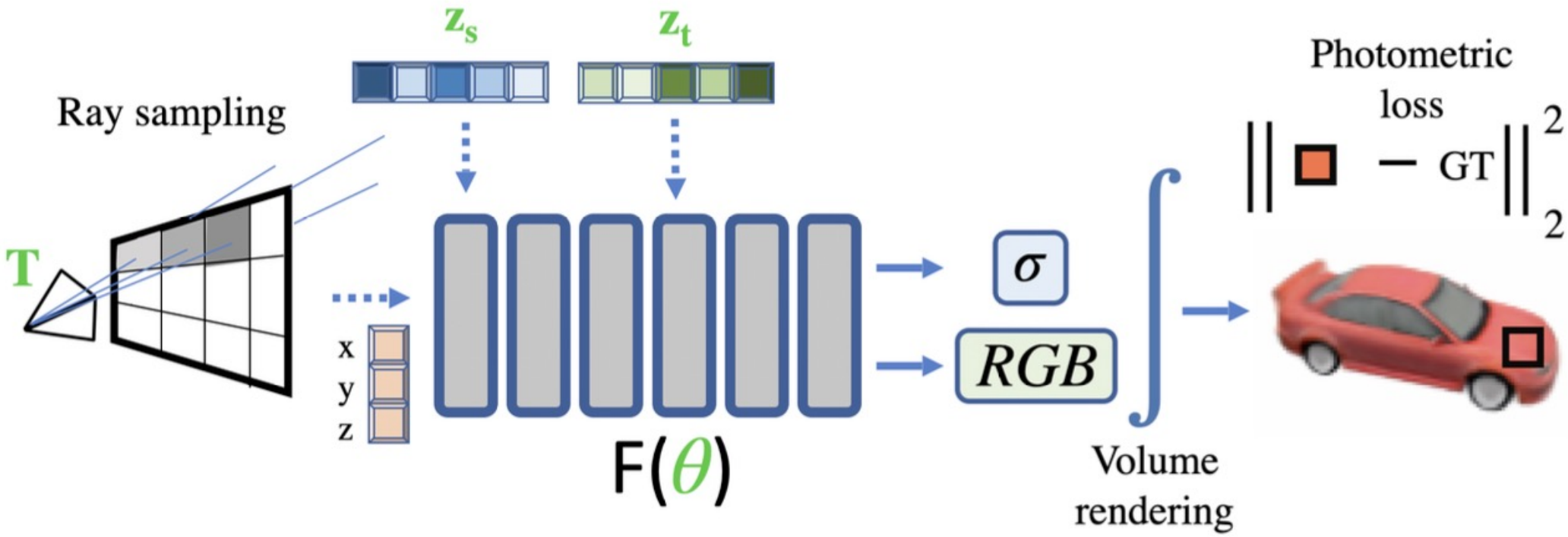
Introduction

- CodeNeRF is an implicit 3D neural representation. Unlike the original NeRF, which is scene specific, CodeNeRF learns to **disentangle shape and texture by learning separate embeddings** across a category .

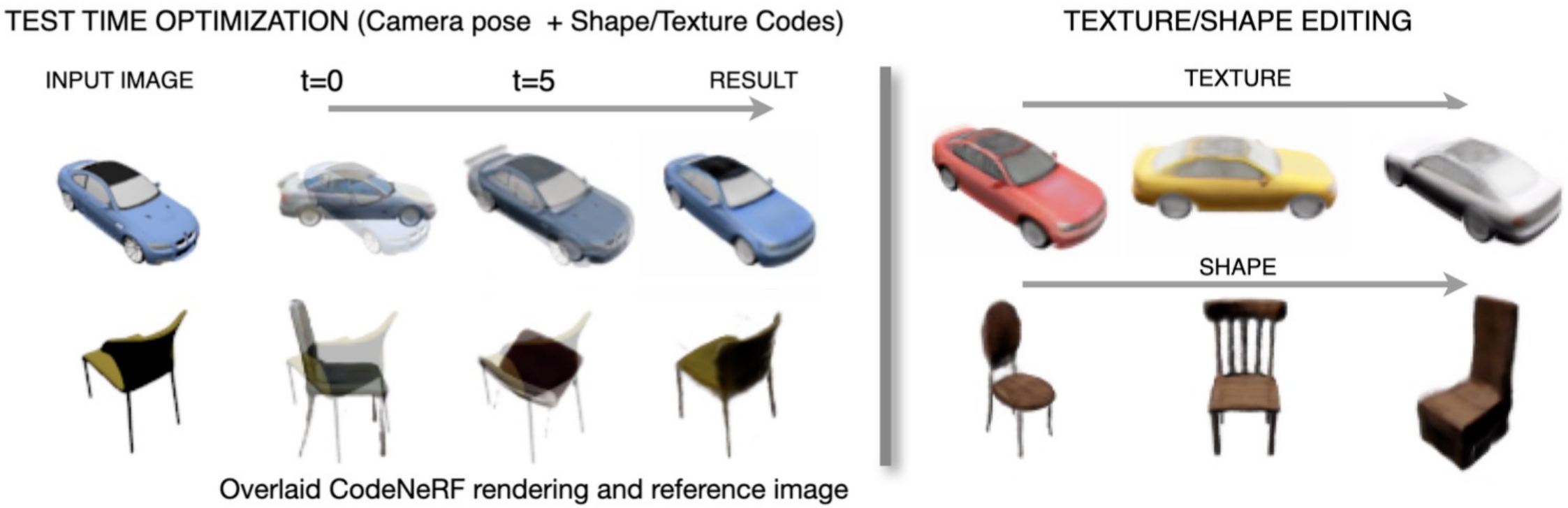
- At test time, given a single unposed image of an unseen object, CodeNeRF jointly **estimates camera viewpoint**, and **shape and appearance codes** via optimization.

- Unseen objects can be **reconstructed from a single image**, and then rendered from new viewpoints or their **shape and texture edited** by varying the latent codes.

Introduction



Introduction



- At inference, given a single unposed reference image of an unseen object, CodeNeRF optimizes shape and texture codes as well as camera pose.
- Our disentangled representation provides full control over the synthesis task, enabling explicit editing of object shape and texture simply by modifying the respective latent codes.

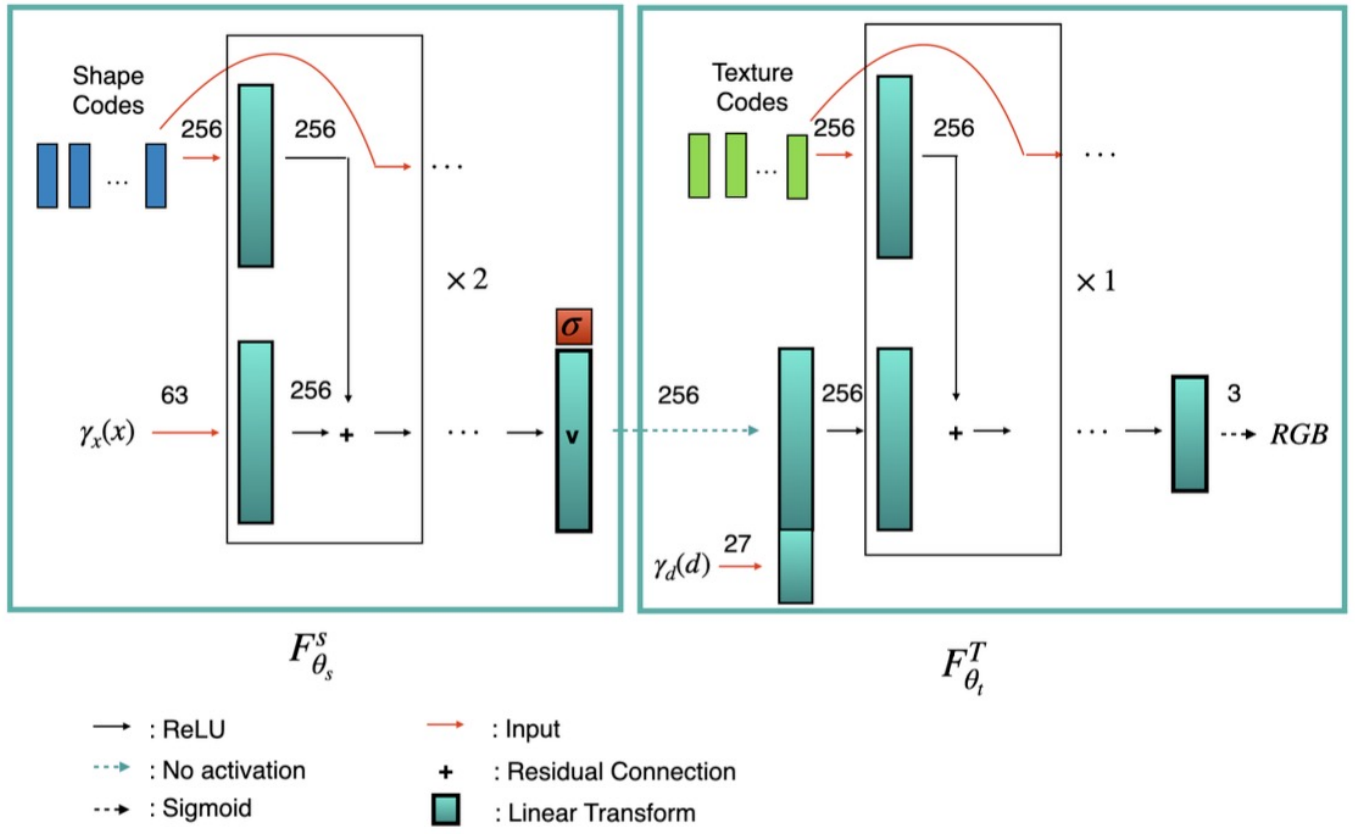
Framework

$$\mathbf{F}_\Theta : (\gamma_x(\mathbf{x}), \gamma_d(\mathbf{d}), \mathbf{z_s}, \mathbf{z_t}) \rightarrow (\sigma, \mathbf{c}) \quad (1)$$

$$\mathbf{F}^s_{\Theta_s} : (\gamma_\mathrm{x}(\mathbf{x}), \mathbf{z_s}) \rightarrow (\sigma, \mathbf{v})$$

$$\mathbf{F}^t_{\Theta_t} : (\mathbf{v}, \gamma_\mathrm{d}(\mathbf{d}), \mathbf{z_t}) \rightarrow (\mathbf{c}) \quad (2)$$

$$\mathbf{F}_\Theta : \mathbf{F}^s_{\Theta_s} \circ \mathbf{F}^t_{\Theta_t}$$

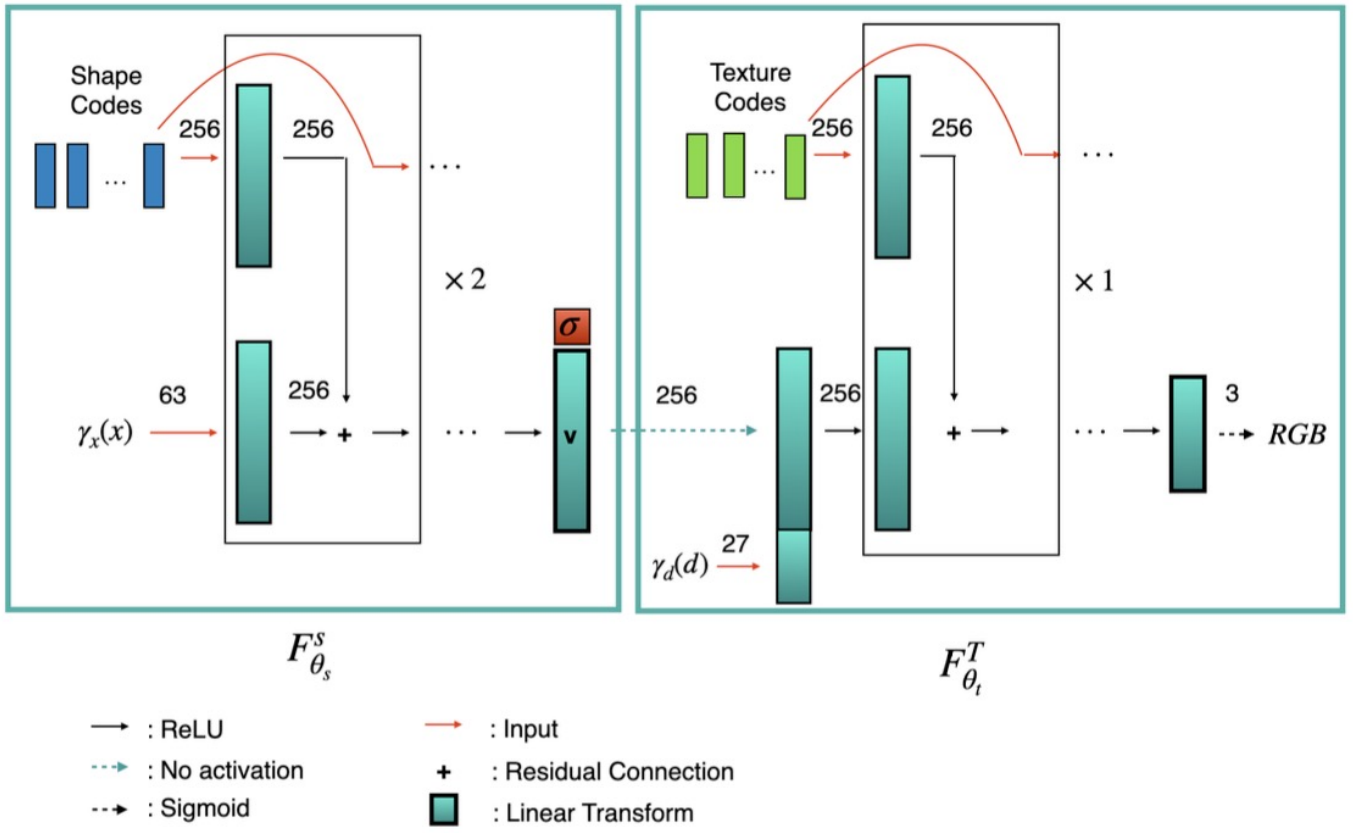


- Training set of N images depicting M objects across a semantic class
- With their respective camera intrinsics and pose parameters $V = \{I_i, K_i, T_i\}_{j=1}^{N}$
- $\{z_s^j, z_t^j\}_{j=1}^{M} \in \mathbb{R}^{256}$

Training CodeNeRF

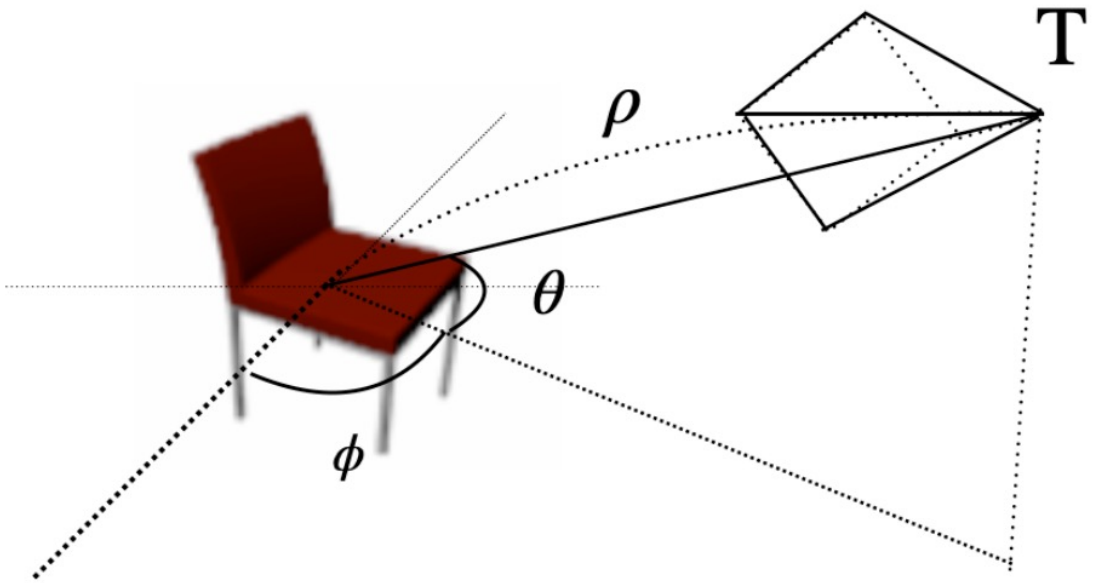
$$\mathcal{L}(\Theta, \{\mathbf{z_s}^i, \mathbf{z_t}^i\}^{\mathrm{M}}) = \sum_{r \in \mathcal{R}} ||\hat{C(r)} - C(r)||_2^2$$

$$\min_{\Theta, \{\mathbf{z_s}^i, \mathbf{z_t}^i\}^{\mathrm{M}}} \mathcal{L}(\Theta, \{\mathbf{z_s}^i, \mathbf{z_t}^i\}) + \frac{1}{\nu^2}(||\mathbf{z_s}^i||_2^2 + ||\mathbf{z_t}^i||_2^2) \quad (4)$$



- Training is supervised using the photometric loss along with a regularization loss.
- Sample a batch of 4094 rays using the intrinsic and extrinsic parameters, then sample 64 points along each ray

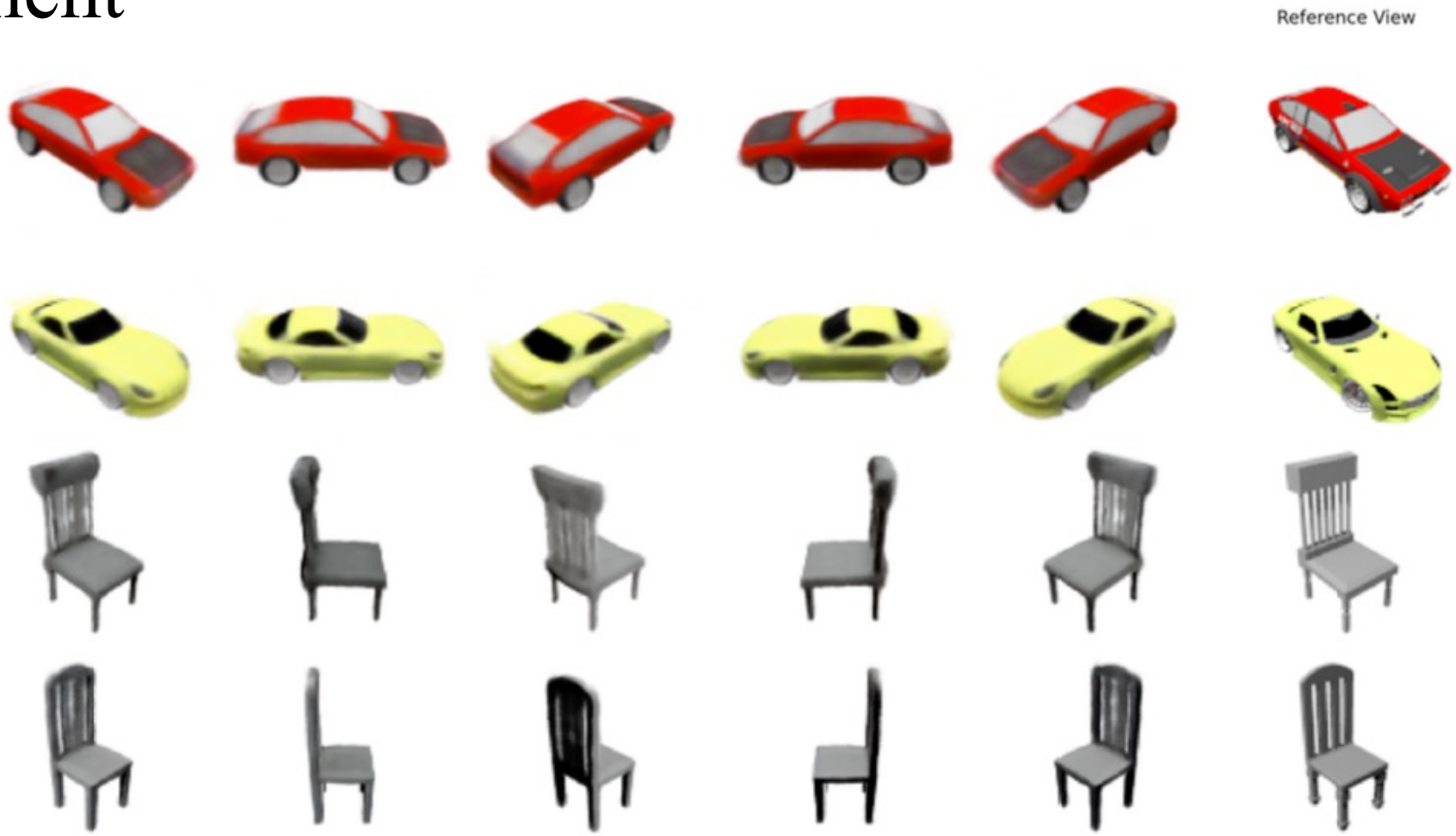
Inference Optimization



$$\min_{\mathbf{z_s}^i, \mathbf{z_t}^i, \rho_i, \theta_i, \phi_i} \mathcal{L}(\mathbf{z_s}^i, \mathbf{z_t}^i, \rho_i, \theta_i, \phi_i) + \frac{1}{\nu^2}(||\mathbf{z_s}^i||_2^2 + ||\mathbf{z_t}^i||_2^2) \tag{5}$$

- At inference time, we do not require known camera viewpoint and we optimize the rotation and translation parameters jointly with the latent embedding vectors.
- Latent vectors are initialized with the mean vector of the trained embeddings

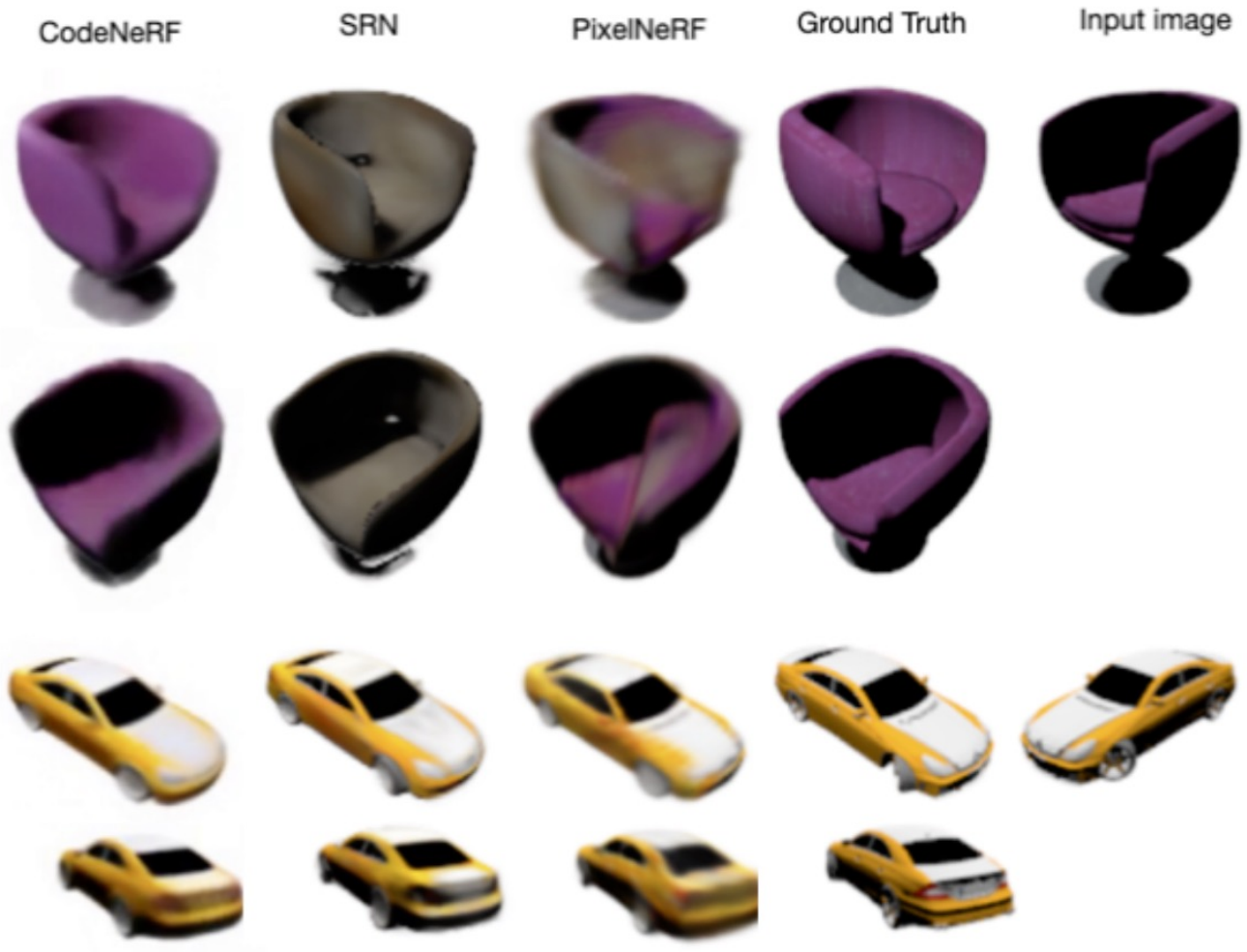
Experiment



- Novel view synthesis of unseen objects from a single input image with CodeNeRF (GT pose), a variant of CodeNeRF which assumes known camera pose at test time.

Experiment

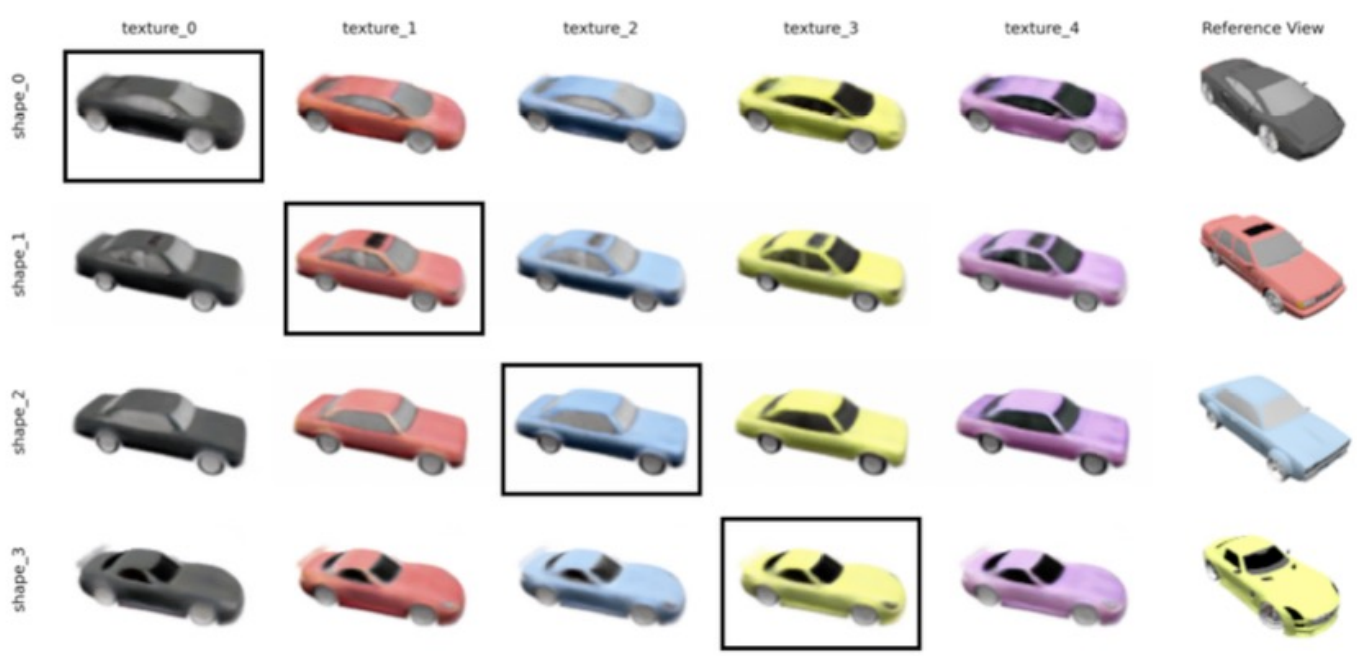
	1-view		2-view	
	PSNR	SSIM	PSNR	SSIM
Chairs				
GRF [28]	21.25	0.86	22.65	0.88
TCO [26]	21.27	0.88	21.33	0.88
dGQN [5]	21.59	0.87	22.36	0.89
ENR [4]	22.83	-	-	-
SRN [24]	22.89	0.89	24.48	0.92
PixelNeRF [37]	**23.72**	**0.91**	**26.20**	**0.94**
CodeNeRF (GT pose)	23.66	0.90	25.63	0.91
CodeNeRF	22.39	0.87	-	-
CodeNeRF (– outliers)	23.11	0.89	-	-
Cars				
SRN [24]	22.25	0.89	24.84	0.92
ENR [4]	22.26	-	-	-
PixelNeRF [37]	23.17	0.90	25.66	**0.94**
CodeNeRF (GT pose)	**23.80**	**0.91**	**25.71**	0.93
CodeNeRF	22.73	0.89	-	-
CodeNeRF (– outliers)	23.17	0.90	-	-

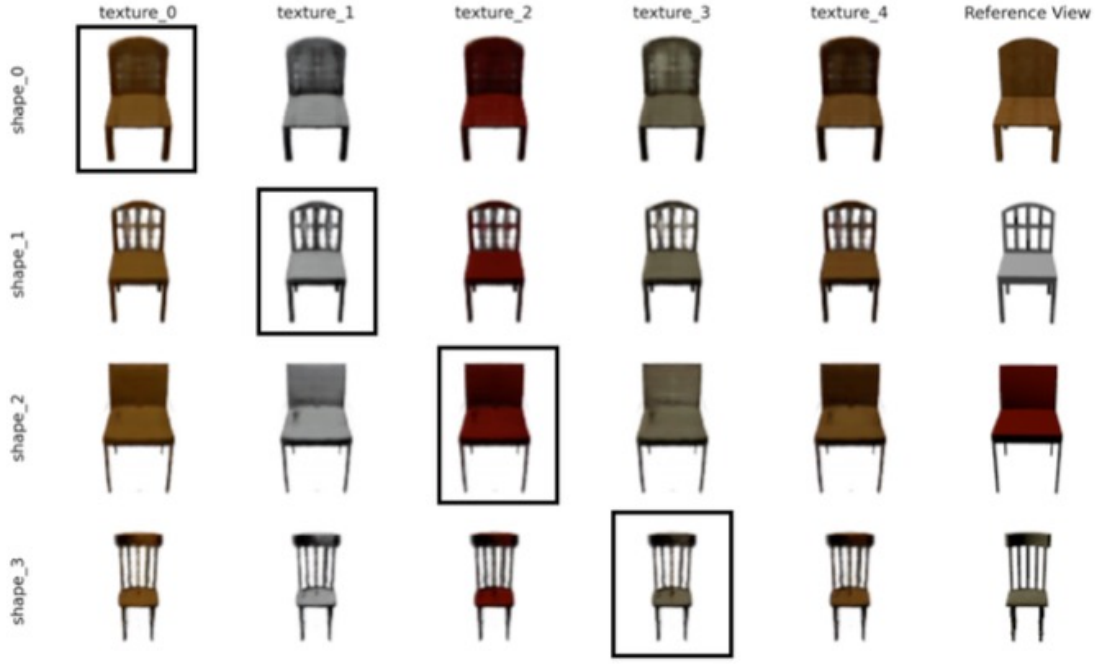


Experiment – test time optimization

- Test-time joint optimization of camera pose and latent codes
- Even when pose and latent codes are initialized far from the ground truth, CodeNeRF converges to good estimates.

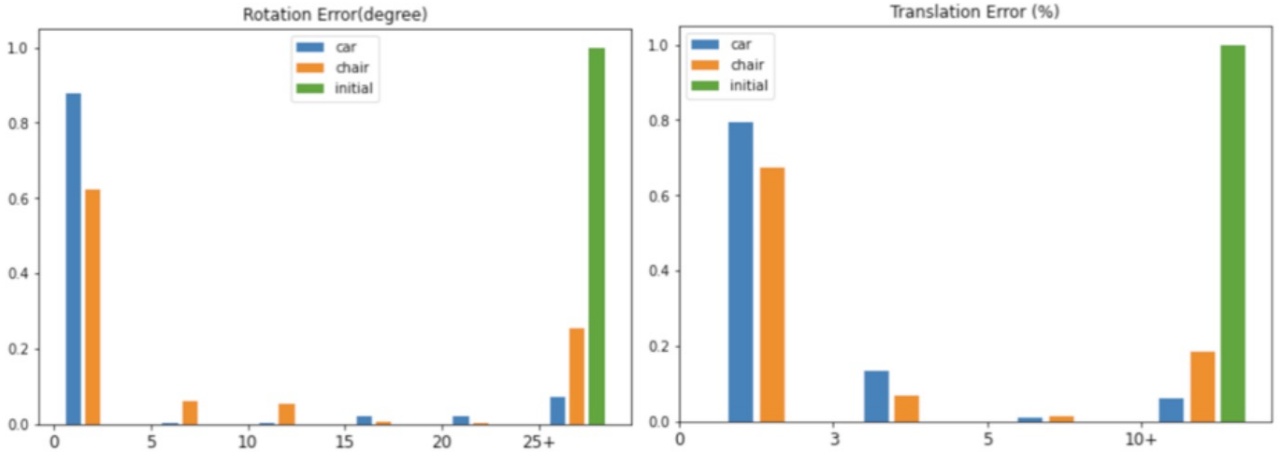
Experiment - Novel Shape/Texture/Pose Synthesis





- CodeNeRF provides full control over the synthesis process as object shapes and textures can be edited simply by varying the corresponding latent codes.

Pose estimation evaluation



		Rotation Error		Translation Error	
		5°	10°	3%	5%
Ours	Cars	87.8%	88.1%	79.3%	92.7%
	Chairs	62.3 %	68.3 %	67.3 %	74.0%

- shows the numerical errors between the estimated and ground-truth camera poses in terms of the percentage of rotation estimates with errors below 5◦ and 10◦, and the percentage of translation estimates with relative errors below 3% and 5%

Related Work – DiffRF

DiffRF: Rendering-Guided 3D Radiance Field Diffusion

Norman Müller[1,2] Yawar Siddiqui[1,2] Lorenzo Porzi[2] Lorenzo Porzi[2] Samuel Rota Bulò[2]
Peter Kontschieder[2] Matthias Nießner[1]

Technical University of Munich[1] Meta Reality Labs Zurich[2]

CVPR 2023 Highlight

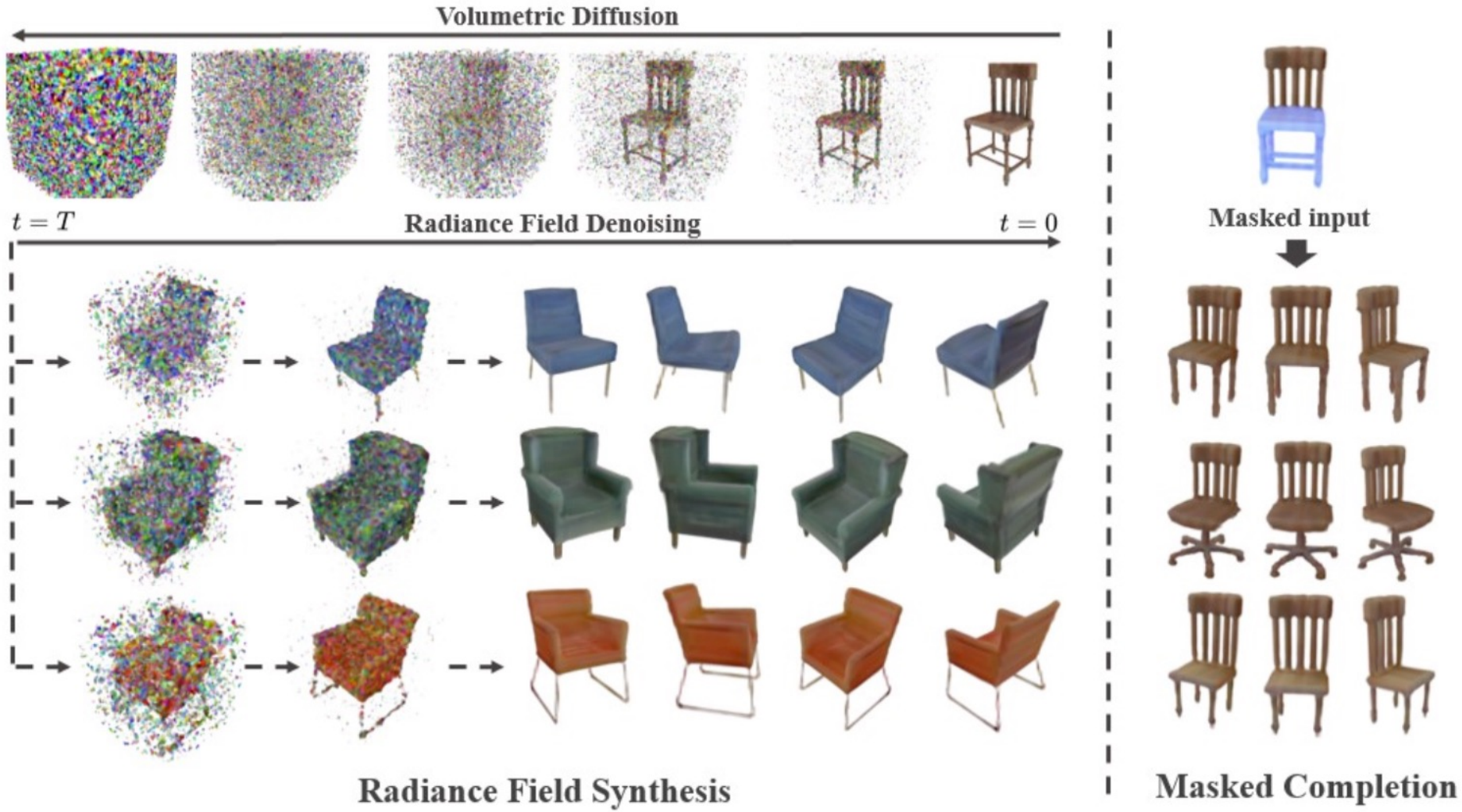
Introduction

- Introduce the first **diffusion model to operate directly on 3D radiance fields**, enabling high-quality, truthful 3D geometry and image synthesis.

- Propose a 3D denoising model which directly operates **on an explicit voxel grid** representation, and the model **learn such a generative model** trained across objects.

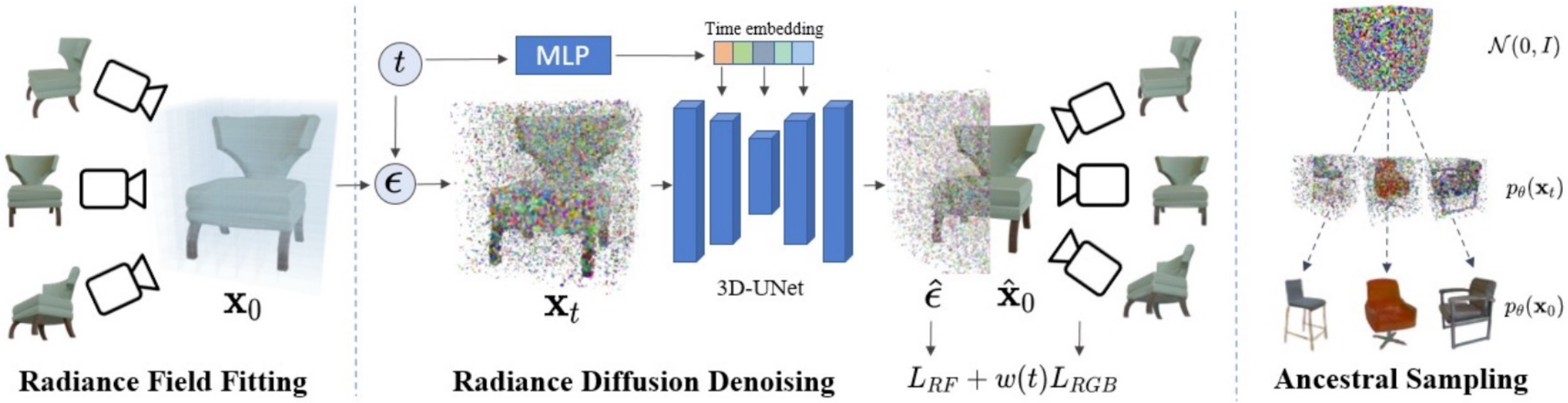
- Our diffusion-based approach naturally enables **conditional generation** such as **masked completion** or **single-view 3D synthesis** at inference time, and show compelling results.

Introduction



- Our method performs denoising of a probabilistic diffusion process applied to 3D radiance fields.
- Novel application of masked completion, solved by our model as conditional inference without task-specific training.

Framework

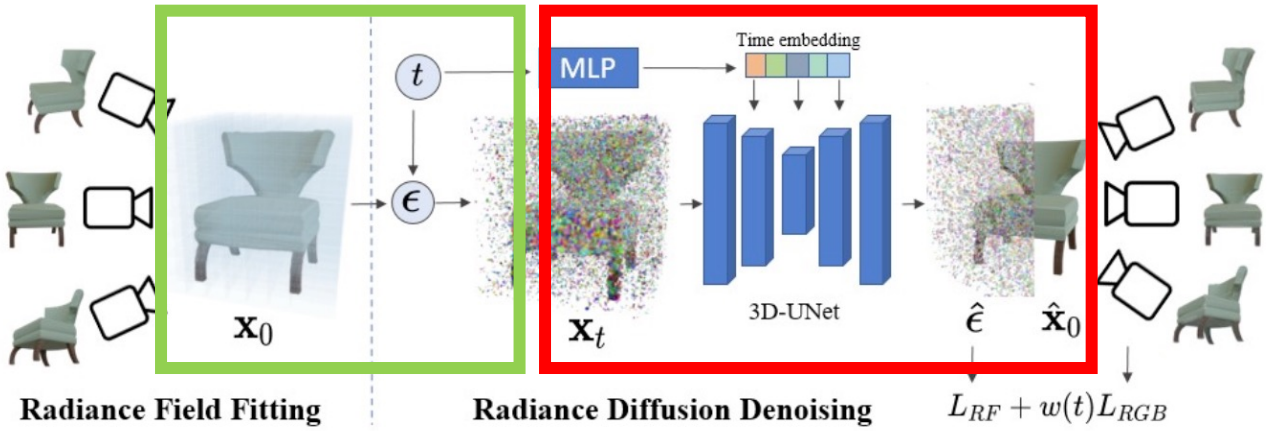


- For a time step t uniformly sampled from $1, \ldots, T$
- we first diffuse an initial radiance field f_0 and the resulting f_t is passed through a time-conditioned 3D-UNet, giving an estimate of the applied noise ϵ.
- We guide the model by the **noise prediction loss** $\mathrm{L_{RF}}$ as well as a **rendering loss** $\mathrm{L_{RGB}}$ on the predicted denoising $\tilde{f}_0$

Framework

Radiance Field Fitting

Generating Radiance Fields



- Generation process

$$p(f_T) := \mathcal{N}(f_T|0, I)$$

$$p_\theta(f_{t-1}|f_t) := \mathcal{N}(f_{t-1}|\mu_\theta(f_t, t), \Sigma_t)\,. \tag{3}$$

$$\mu_\theta(f_t, t) := a_t(f_t - b_t\epsilon_\theta(f_t, t))\,, \tag{4}$$

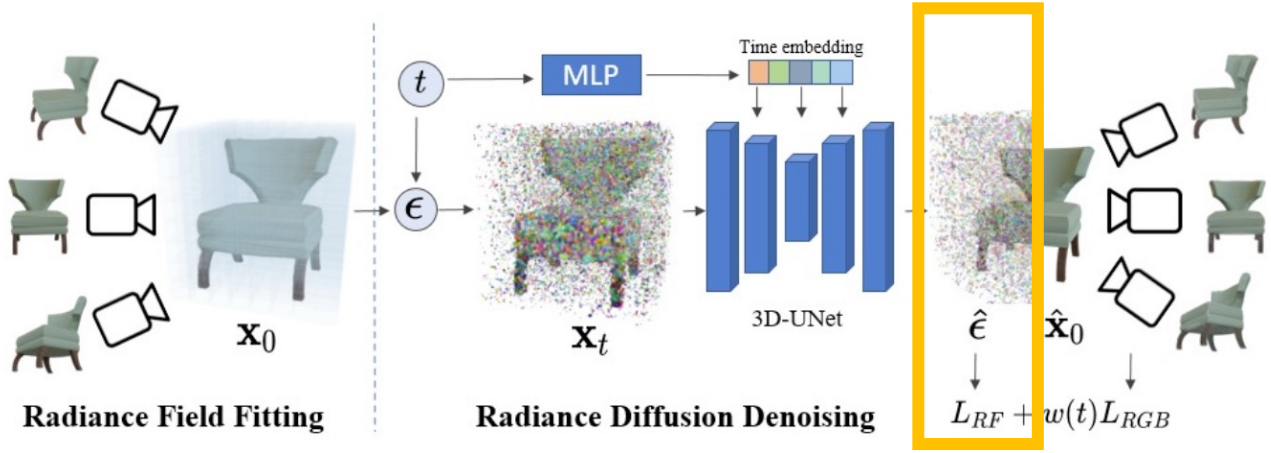
- Diffusion process

$$q(f_t|f_{t-1}) := \mathcal{N}(f_t|\sqrt{\alpha_t}f_{t-1}, \beta_t I)\,, \tag{5}$$

$$q(f_t|f_0) = \mathcal{N}(f_t|\sqrt{\bar{\alpha}_t}f_0, (1-\bar{\alpha}_t)I)\,, \tag{6}$$

Training Objective

- Radiance field generation loss L_{RF}



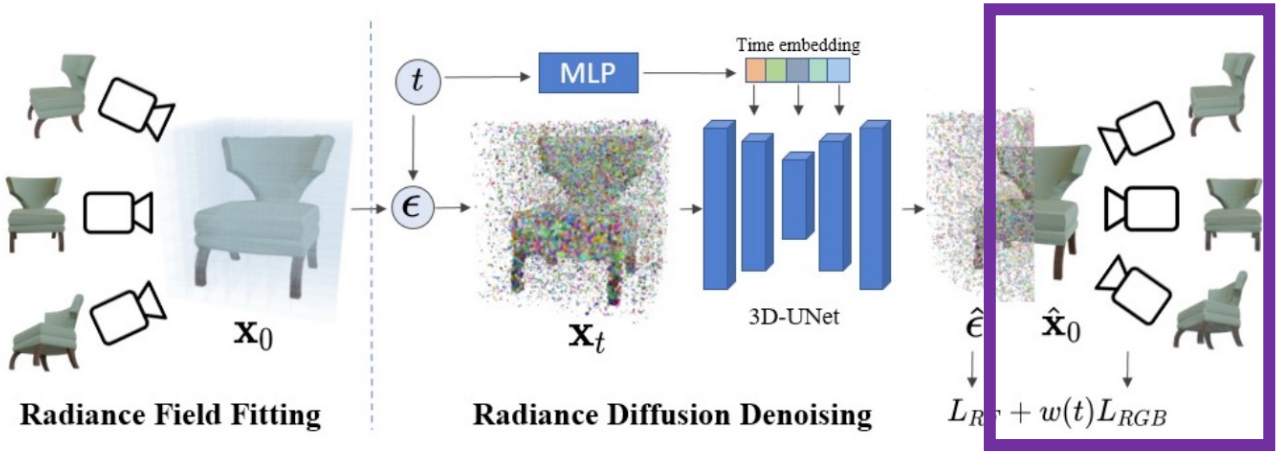
$$-\log p_\theta(f_0) \le \mathbb{E}_q\left[-\log \frac{p_\theta(f_{0:T})}{q(f_{1:T}|f_0)}\right] := L_{\mathtt{RF}}(f_0|\theta)\,, \tag{7}$$

$$L_{\mathtt{RF}}(f_0|\theta) = \sum_{t=1}^{T} L^t_{\mathtt{RF}}(f_0|\theta) + \text{const}\,. \tag{8}$$

$$\begin{aligned} L^t_{\mathtt{RF}}(f_0|\theta) &:= \mathbb{E}_q\left[\|\epsilon - \epsilon_\theta(f_t, t)\|^2\right] \\ &= \mathbb{E}_\phi\left[\left\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} f_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\right\|^2\right]\,, \end{aligned}$$

Training Objective

- Radiance field rendering loss L_{RGB}



$$\tilde{f}_0^t(\epsilon,\theta) \coloneqq f_0 + \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}(\epsilon - \epsilon_\theta(f_t,t))$$

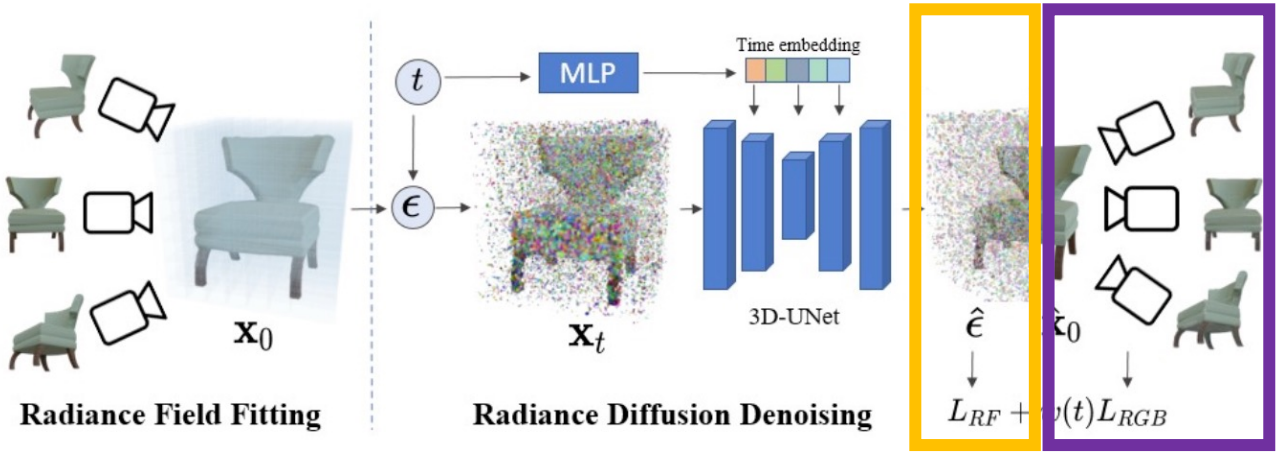
$$L_{\mathtt{RGB}}(f_0|\theta) \coloneqq \sum_{t=1}^{T} L_{\mathtt{RGB}}^t(f_0|\theta)\,. \tag{9}$$

$$\ell_v(f,I) \coloneqq \|I_v - R(v,f)\|^2\,. \tag{10}$$

$$L_{\mathtt{RGB}}^t(f_0|\theta) \coloneqq \omega_t \mathbb{E}_{\phi,\psi}\left[\ell_v(\tilde{f}_0^t(\epsilon,\theta),I)\right]\,, \tag{11}$$

Training Objective

- Final loss

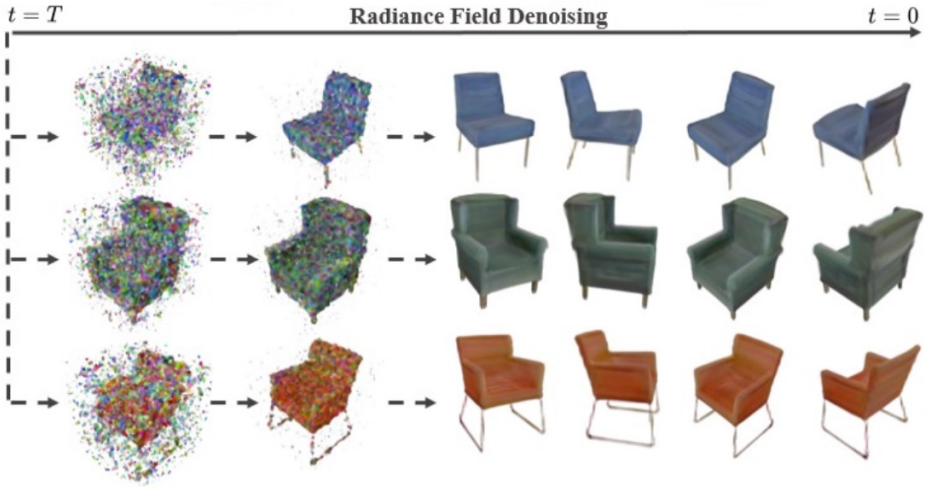


$$L(\theta) := L_{\mathtt{RF}}(f_0|\theta) + \lambda_{\mathtt{RGB}} L_{\mathtt{RGB}}(f_0|\theta)$$
$$\propto \mathbb{E}_{\kappa}\left[L^t_{\mathtt{RF}}(f_0|\theta) + \lambda_{\mathtt{RGB}} L^t_{\mathtt{RGB}}(f_0|\theta)\right].$$

- with a small variation that enables stochastic sampling of the step t from a uniform distribution κ(t)
- $T = 1000$

Experiment

- Unconditional Radiance Field Synthesis



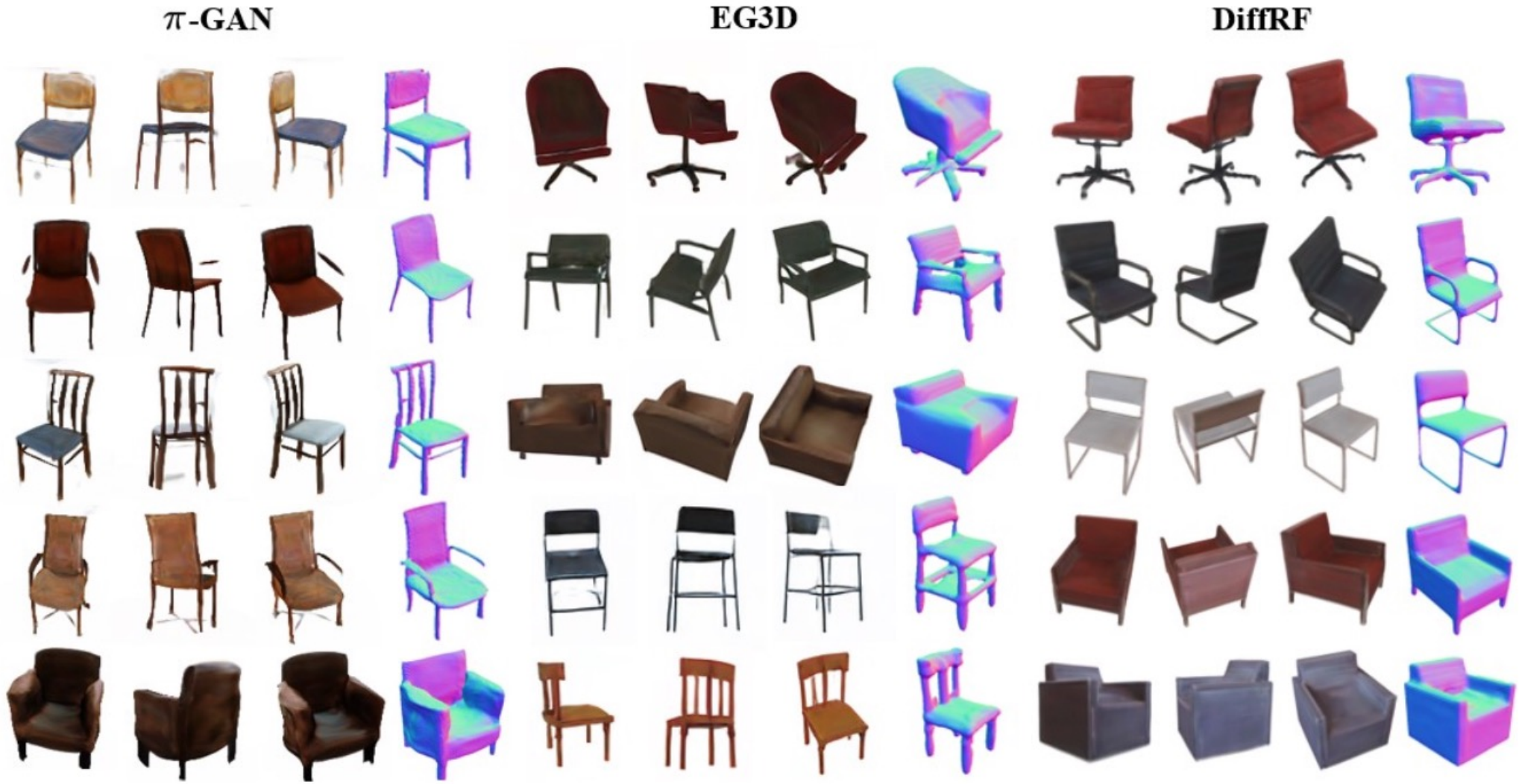
Radiance Field Synthesis

Method	FID ↓	KID ↓	COV ↑	MMD ↓
π-GAN [7]	52.71	13.64	39.92	7.387
EG3D [8]	16.54	8.412	47.55	5.619
DiffRF w/o 2D	18.27	9.263	**59.20**	4.543
DiffRF	**15.95**	**7.935**	58.93	**4.416**

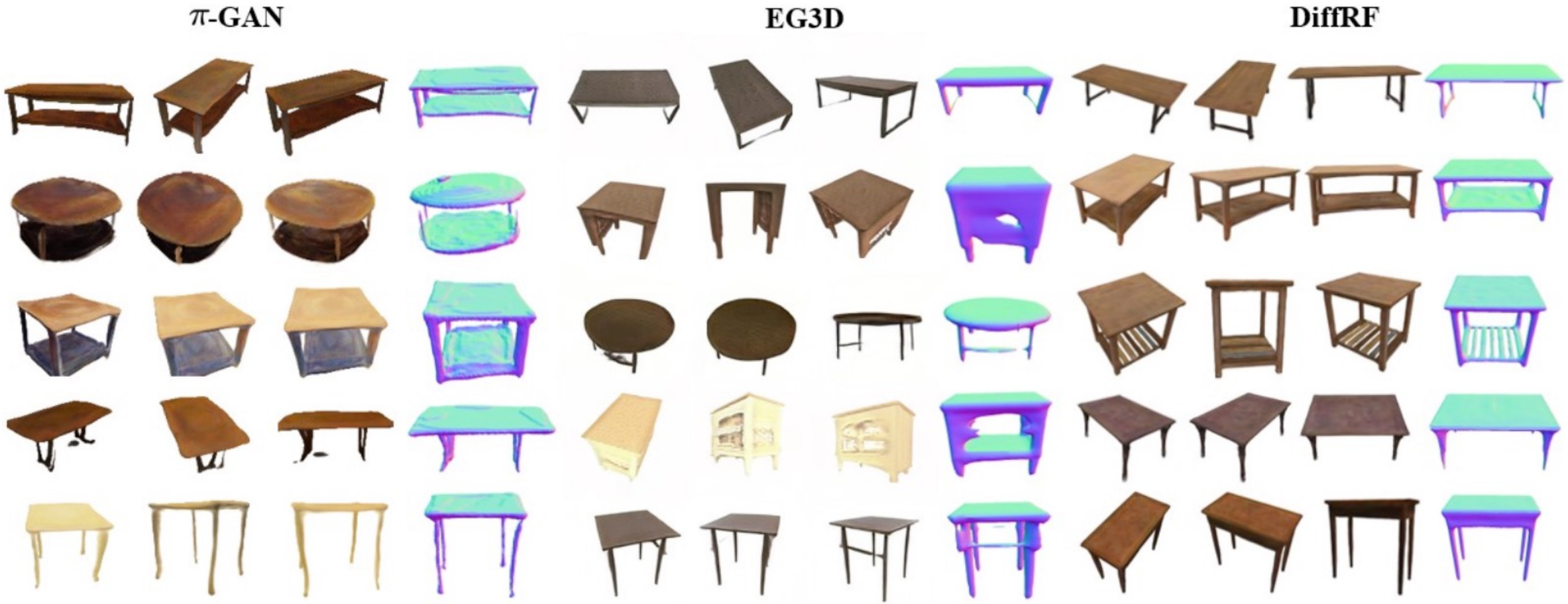
Method	FID ↓	KID ↓	COV ↑	MMD ↓
π-GAN [7]	41.67	13.81	44.23	10.92
EG3D [8]	31.18	11.67	48.15	9.327
DiffRF w/o 2D	35.89	13.94	**63.46**	8.013
DiffRF	**27.06**	**10.03**	61.54	**7.610**

- We run experiments on the PhotoShape Chairs and on the Amazon Berkeley Objects (ABO) Tables dataset.

Unconditional Radiance Field Synthesis



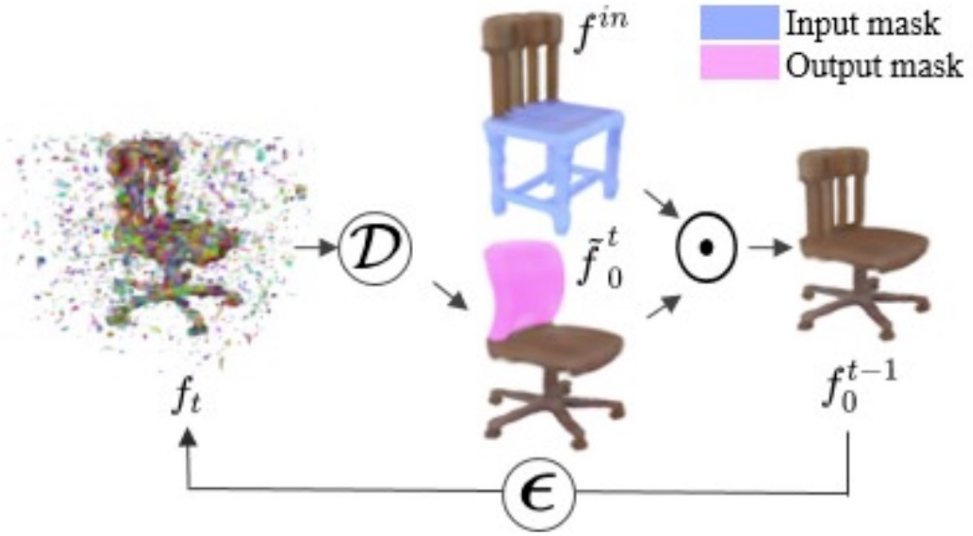
Unconditional Radiance Field Synthesis



Conditional Generation

- Masked Radiance Field Completion

$$f_0^{t-1} = \sqrt{\bar{\alpha}_t}(m \odot \tilde{f}_0^t + (1-m) \odot f^{in})$$
$$f_{t-1} \sim \mathcal{N}(f_0^{t-1}, (1-\bar{\alpha}_t)I)\,,$$

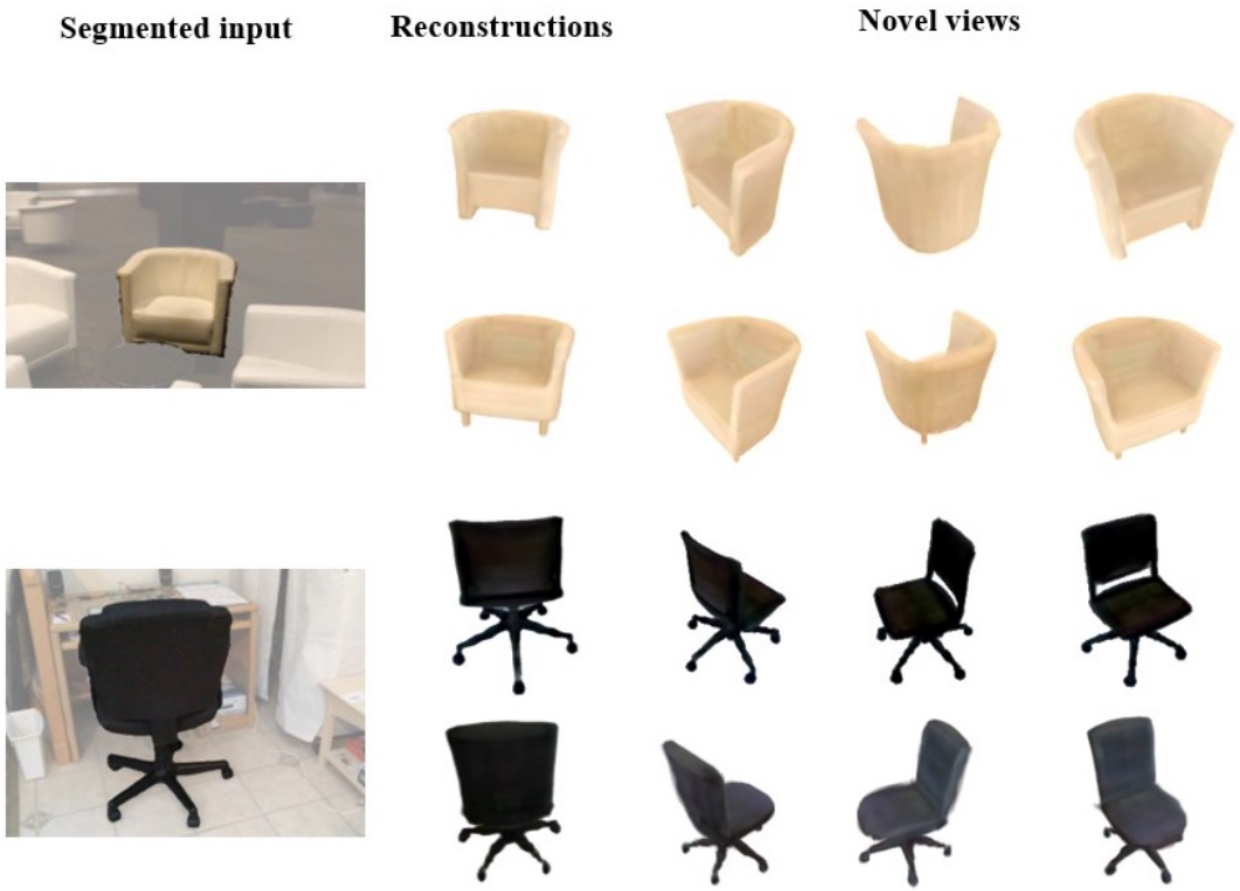


- DiffRF shows
  - more diverse proposals compared to EG3D
  - also maintaining the original non-masked regions

Conditional Generation

- Image-to-Volume Synthesis

$$\tilde{f}_0^t \leftarrow \tilde{f}_0^t - \lambda \nabla_{\tilde{f}_0^t}(\tilde{I}_t, I)$$



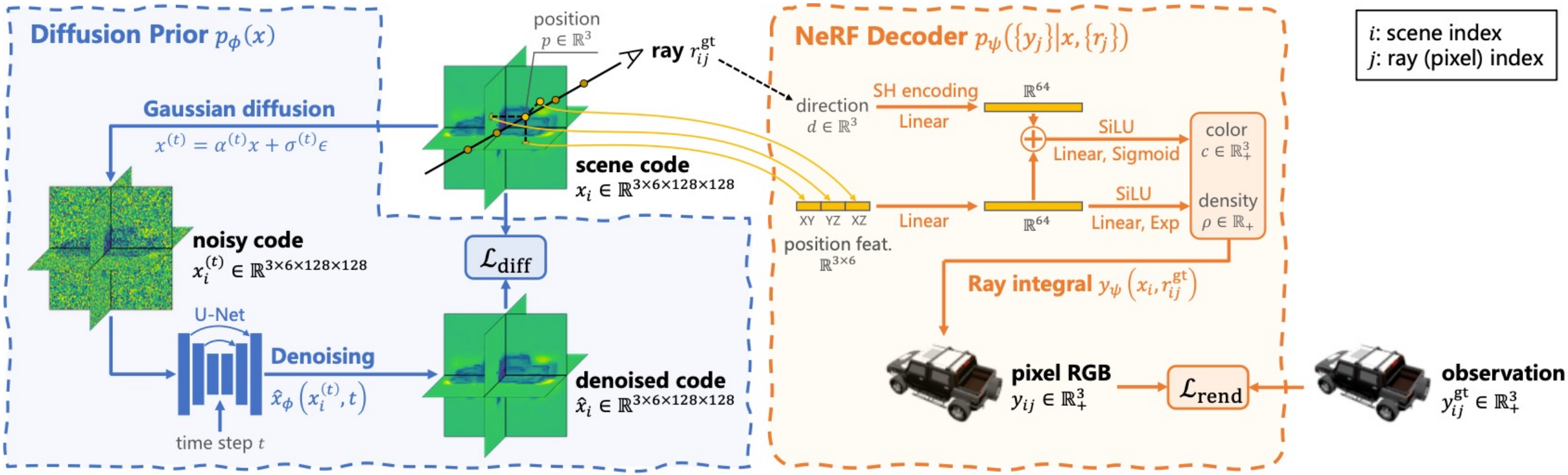
Conclusion of Related work

- CodeNeRF:
  - A neural radiance field that learns **separate latent embeddings** for shape and texture.
  - Given a single input image it can **estimate its associated camera pose and latent codes**, and do **one-shot reconstruction**.
- DiffRF:
  - **First generative diffusion-based method** to operate directly on volumetric radiance fields.
  - Learns **multi-view consistent priors** from collections of posed images, enabling generation in both conditional and unconditional 3D generation tasks.

Outline

- Introduction
- Related Work
- Framework
- Method
- Experiment
- Conclusion

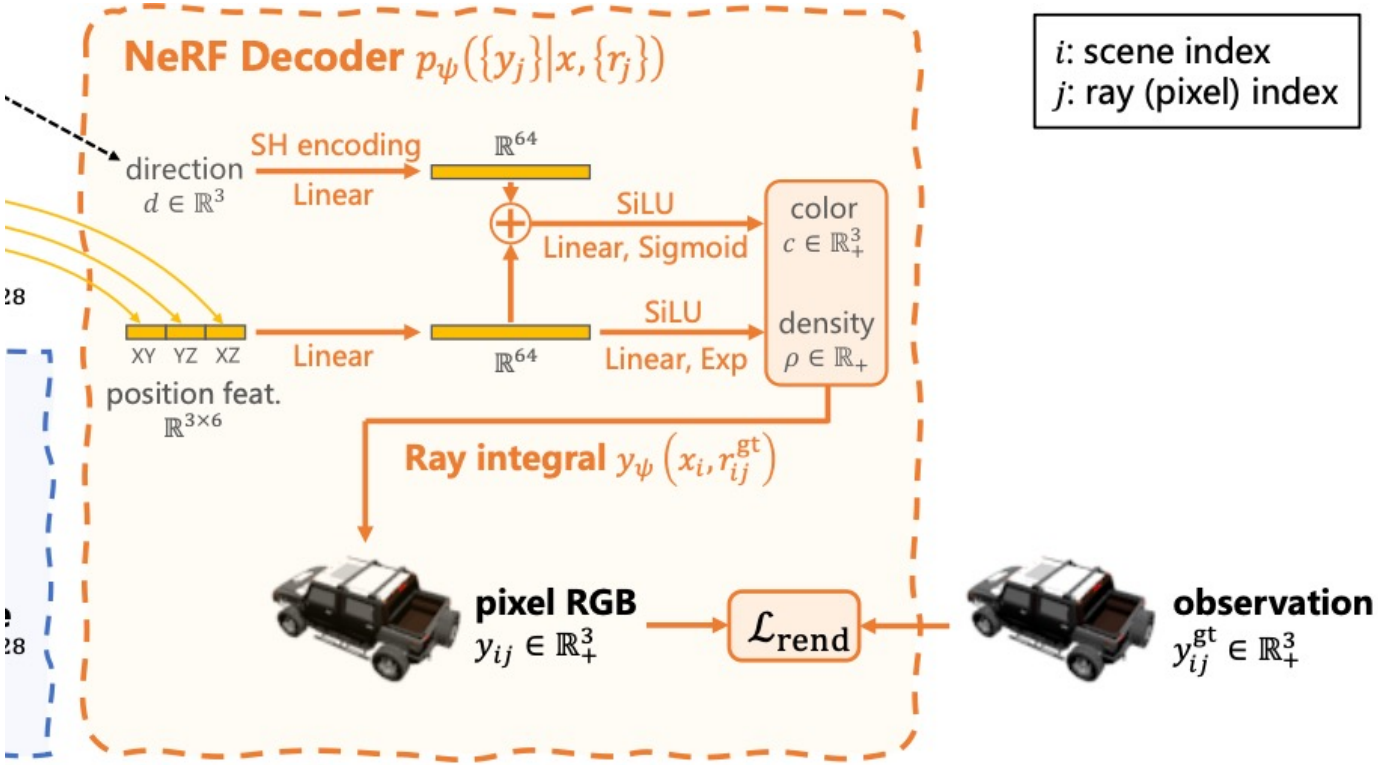
Framework


Diffusion Prior $p_\phi(x)$
Gaussian diffusion
$x^{(t)} = \alpha^{(t)}x + \sigma^{(t)}\epsilon$
noisy code
$x_i^{(t)} \in \mathbb{R}^{3\times6\times128\times128}$
U-Net
Denoising
$\hat{x}_\phi\left(x_i^{(t)}, t\right)$
time step t
denoised code
$\hat{x}_i \in \mathbb{R}^{3\times6\times128\times128}$
$\mathcal{L}_{\text{diff}}$
position
$p \in \mathbb{R}^3$
ray r_{ij}^{gt}
scene code
$x_i \in \mathbb{R}^{3\times6\times128\times128}$
NeRF Decoder $p_\psi(\{y_j\}|x,\{r_j\})$
direction
$d \in \mathbb{R}^3$
SH encoding
Linear
$\mathbb{R}^{64}$
SiLU
Linear, Sigmoid
color
$c \in \mathbb{R}_+^3$
SiLU
Linear, Exp
density
$\rho \in \mathbb{R}_+$
XY YZ XZ
position feat.
$\mathbb{R}^{3\times6}$
Linear
$\mathbb{R}^{64}$
Ray integral $y_\psi\left(x_i, r_{ij}^{\text{gt}}\right)$
pixel RGB
$y_{ij} \in \mathbb{R}_+^3$
$\mathcal{L}_{\text{rend}}$
observation
$y_{ij}^{\text{gt}} \in \mathbb{R}_+^3$
i: scene index
j: ray (pixel) index

Outline

- Introduction
- Related Work
- Framework
- Method
- Experiment
- Conclusion

NeRF as an Auto-Decoder



$$\mathcal{L}_{\mathrm{rend}}(\{x_i\},\psi) = \mathop{\mathbb{E}}_{i}\left[\sum_j \frac{1}{2}\left\|y_{ij}^{\mathrm{gt}} - y_\psi\left(x_i, r_{ij}^{\mathrm{gt}}\right)\right\|^2\right]. \quad (1)$$

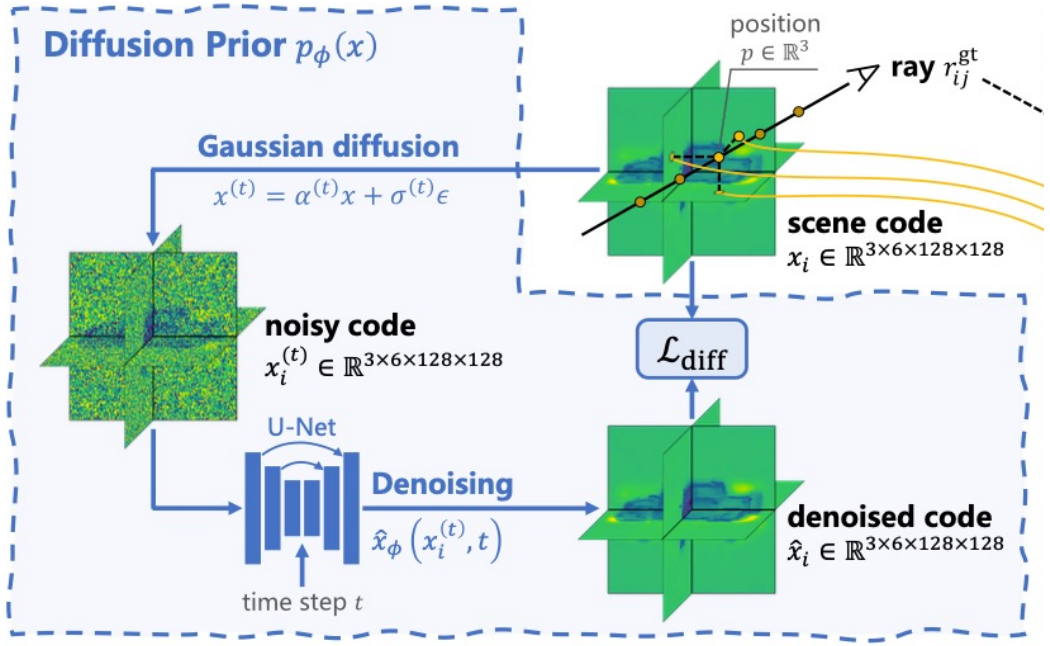
- the scene codes $\{x_i\}$ can be interpreted as the latent codes
- to ensure continuity in generation, a low-dimensional latent space and a complex decoder is required

Latent Diffusion Models

$$\mathcal{L}_{\text{diff}}(\phi) = \mathop{\mathbb{E}}_{i,t,\epsilon}\left[\frac{1}{2}w^{(t)}\left\|\hat{x}_\phi\left(x_i^{(t)}, t\right) - x_i\right\|^2\right], \qquad (2)$$

$$x_i^{(t)} := \alpha^{(t)}x_i + \sigma^{(t)}\epsilon$$

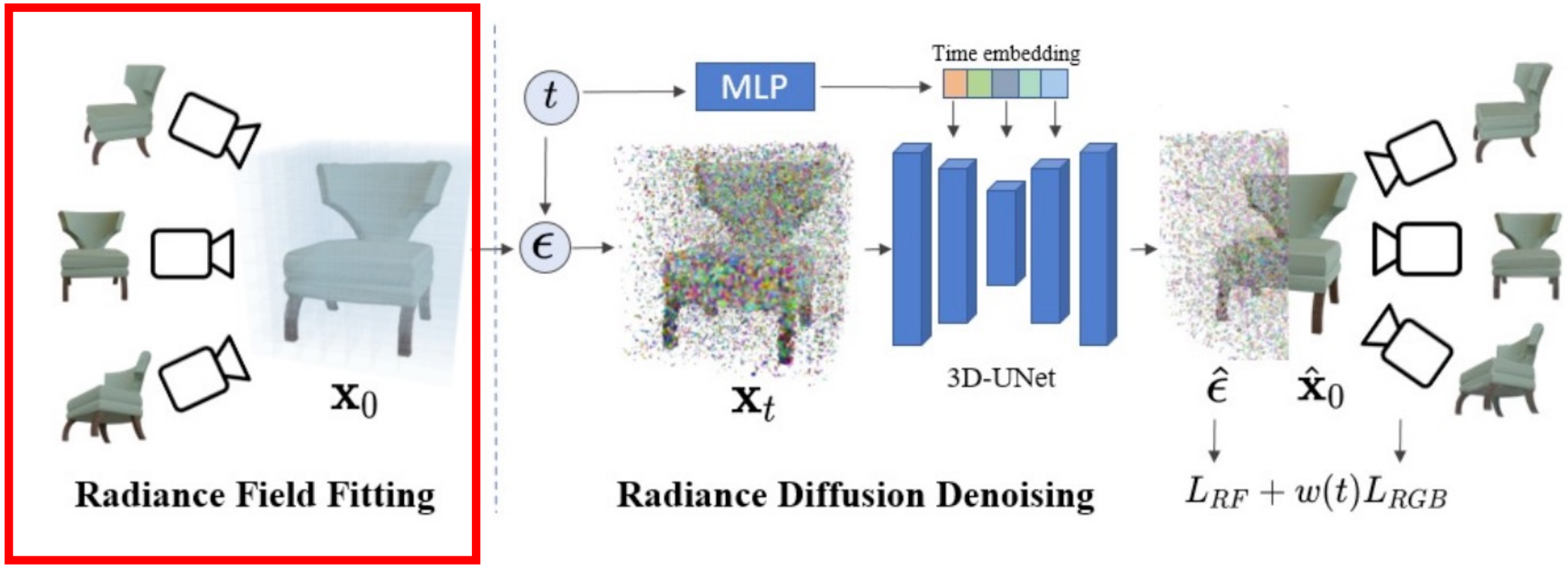
$$t \sim \mathcal{U}(0, T)$$



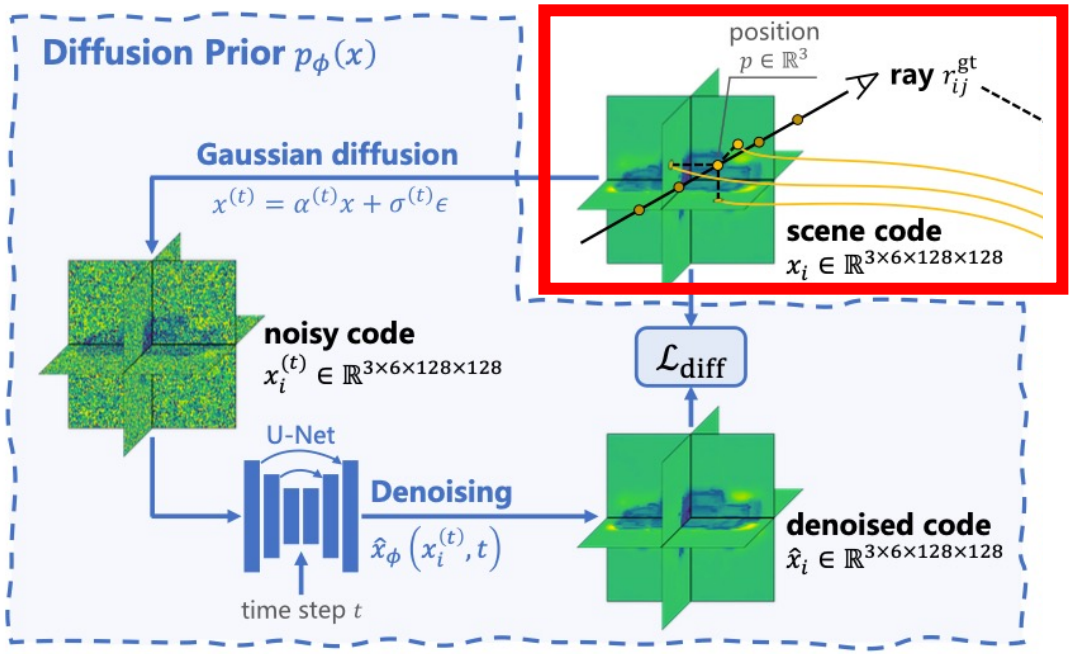
- LDM learn a prior distribution $p_\varphi\,(x)$ in the latent space with parameters φ
- previous work adopts a two-stage training scheme, where the auto-decoder is trained first to obtain the per-scene latent x_i, which is then treated as real data to train the LDM.

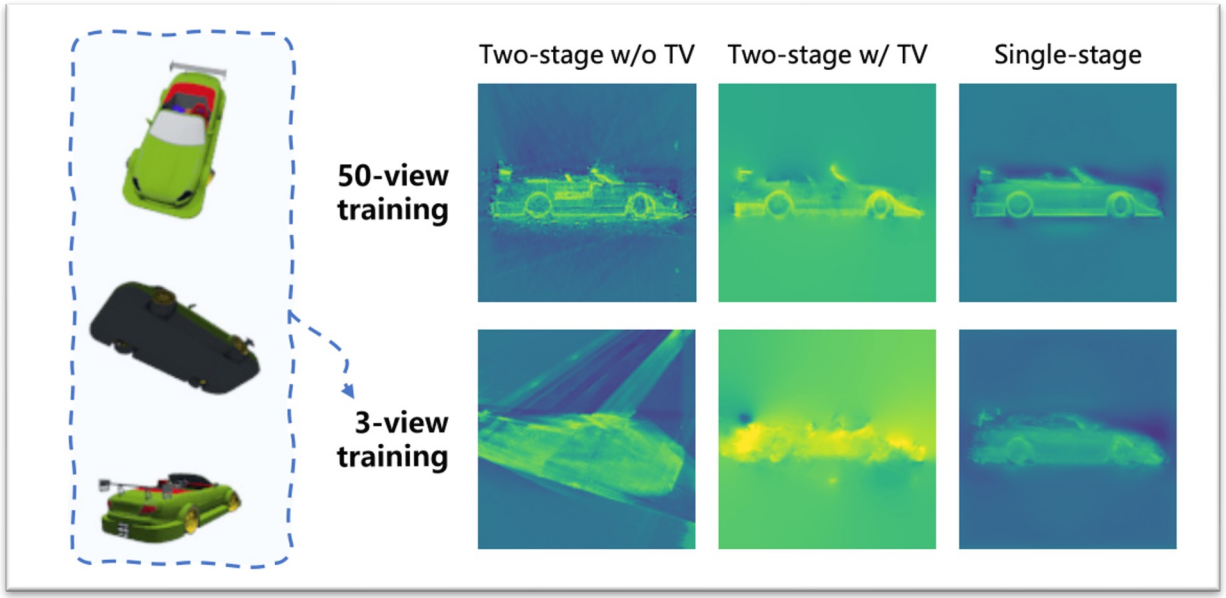
Two-stage Example

- DiffRF



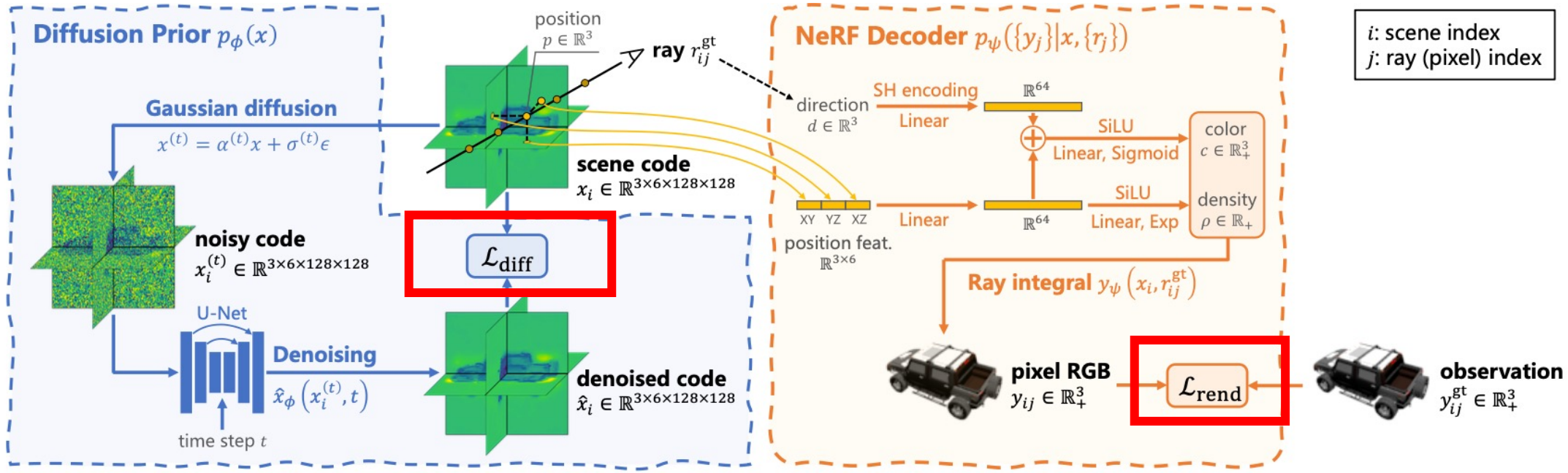
Latent Diffusion Models





- Visualization of the scene code x_{XZ} at XZ plane.
- In two stages, training LDMs with NeRF auto-decoders poses an unprecedented challenge.
- Two-stage methods ignore the prior term L_{diff} during the first stage of training the auto-decoders.

Single-Stage Diffusion NeRF Training



$$\mathcal{L} = \lambda_{\text{rend}}\mathcal{L}_{\text{rend}}(\{x_i\}, \psi) + \lambda_{\text{diff}}\mathcal{L}_{\text{diff}}(\{x_i\}, \phi). \qquad (4)$$

- Single-stage training constrains scene codes $\{x_i\}$ with both terms in the loss function, allowing the learned prior to complete the parts unseen to rendering.

Single-Stage Diffusion NeRF Training

$$\mathcal{L} = \lambda_{\text{rend}}\mathcal{L}_{\text{rend}}(\{x_i\}, \psi) + \lambda_{\text{diff}}\mathcal{L}_{\text{diff}}(\{x_i\}, \phi).$$

- the diffusion loss L_{diff} requires much longer time to evaluate
- introduce **prior gradient caching**

Algorithm 1: Single-stage diffusion NeRF training

Input: $\{y_{ij}^{\text{gt}}, r_{ij}^{\text{gt}}\}$

Initialize $\{x_i\}, \phi, \psi$

for $k_{\text{out}} := 1 \cdots K_{\text{out}}$ **do** // outer loop of K_{out} iterations

  Sample a batch of scenes $i \in B_{\text{sc}}$

  $g_\phi, g_x^{\text{diff}} \leftarrow \nabla_{\phi,\{x_i\}_{B_{\text{sc}}}} \lambda_{\text{diff}}\mathcal{L}_{\text{diff}}$ // diffusion grad

  $\phi \leftarrow \phi - Adam(g_\phi)$

  for $k_{\text{in}} := 1 \cdots K_{\text{in}}$ **do** // inner loop of K_{in} iterations

    Sample a batch of rays $j \in B_{\text{ray}}$

    $g_x^{\text{rend}} \leftarrow \nabla_{\{x_i\}_{B_{\text{sc}}}} \lambda_{\text{rend}}\mathcal{L}_{\text{rend}}$ // rendering grad

    $g_x \leftarrow g_x^{\text{rend}} + g_x^{\text{diff}}$ // add cached prior grad

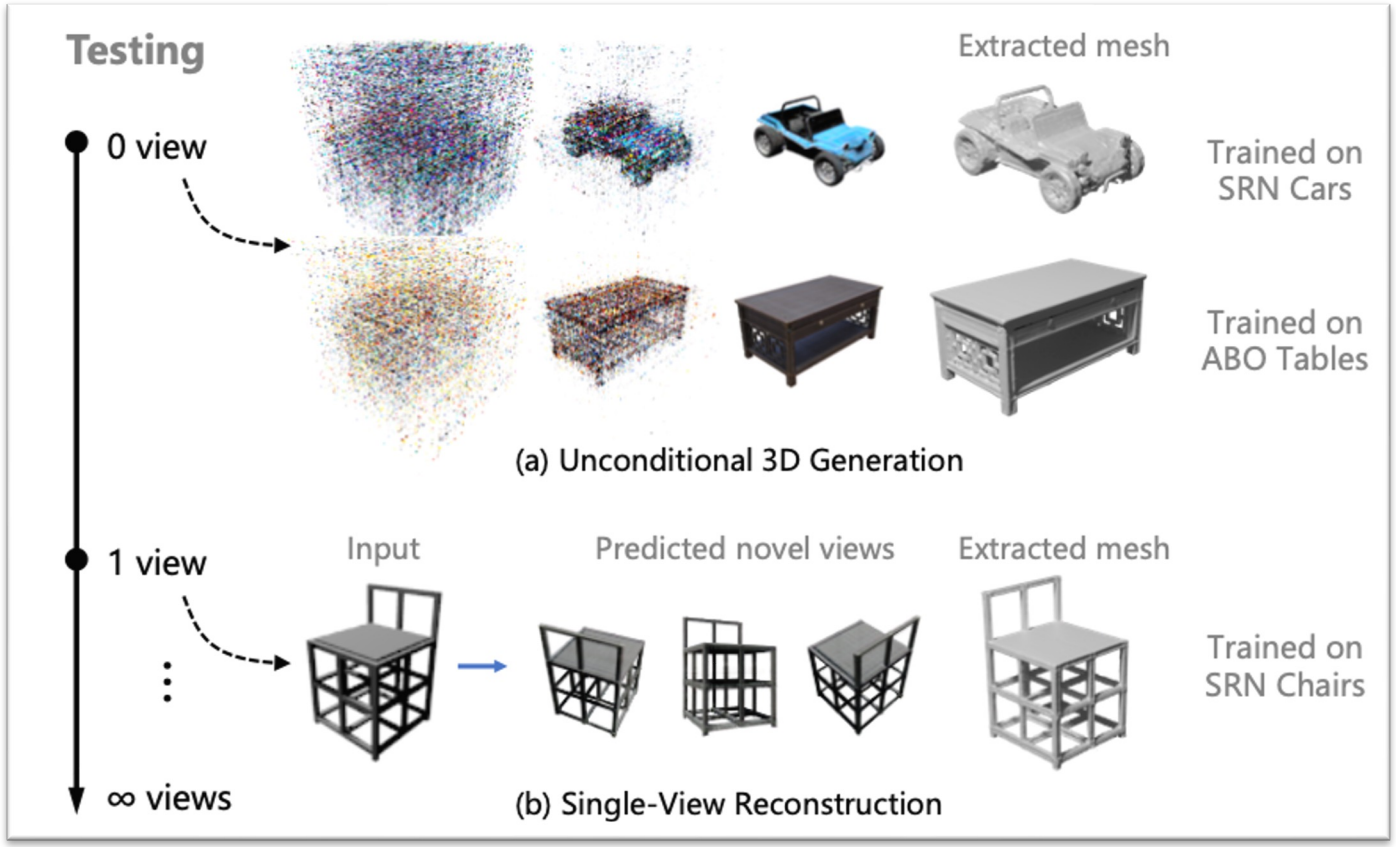
    $\{x_i\}_{B_{\text{sc}}} \leftarrow \{x_i\}_{B_{\text{sc}}} - Adam(g_x)$

    if $k_{\text{in}} = K_{\text{in}}$ **then** // last inner iteration

      $g_\psi \leftarrow \nabla_\psi \lambda_{\text{rend}}\mathcal{L}_{\text{rend}}$

      $\psi \leftarrow \psi - Adam(g_\psi)$

Image-Guided Sampling and Finetuning



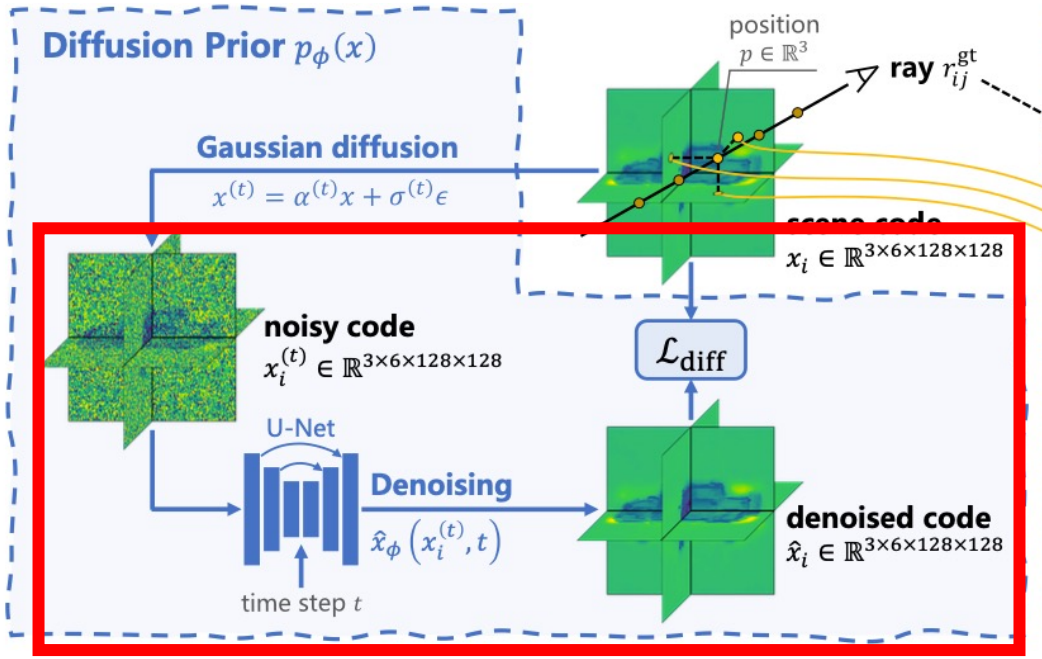
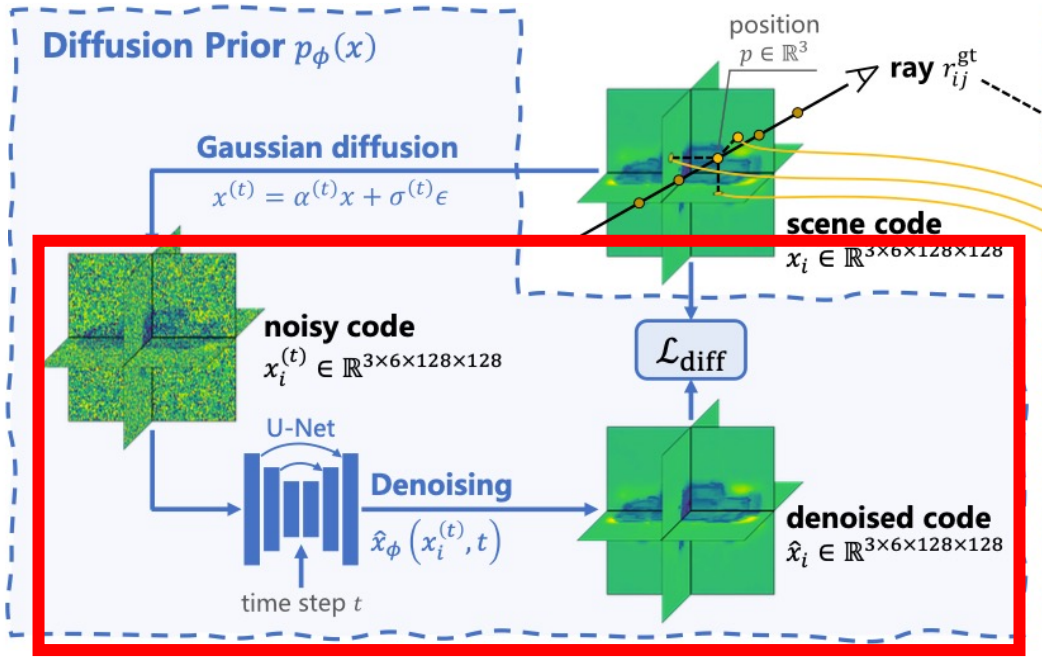


Image-Guided Sampling and Finetuning

$$g \leftarrow \nabla_{x^{(t)}} \lambda_{\text{rend}} \sum_j \frac{1}{2} \left(\frac{\alpha^{(t)}}{\sigma^{(t)}} \right)^{2\omega} \left\| y_j^{\text{gt}} - y_\psi \left(\hat{x}_\phi(x^{(t)}, t), r_j^{\text{gt}} \right) \right\|_2^2, \tag{5}$$

$$\hat{x} \leftarrow \hat{x} - \lambda_{\text{gd}} \frac{{\sigma^{(t)}}^2}{\alpha^{(t)}} g \tag{6}$$

$$\min_x \lambda_{\text{rend}} \mathcal{L}_{\text{rend}}(x) + \lambda'_{\text{diff}} \mathcal{L}_{\text{diff}}(x), \tag{7}$$



- to finetune the sampled scene code x, while freezing the diffusion and decoder parameters

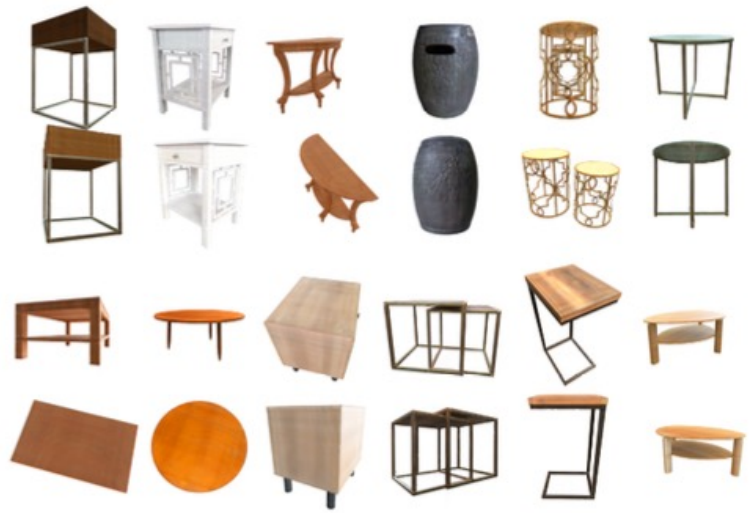
Outline

- Introduction
- Related Work
- Framework
- Method
- Experiment
- Conclusion

Unconditional Generation

(a) EG3D Tables

(b) DiffRF Tables



(c) SSDNeRF Tables (ours)

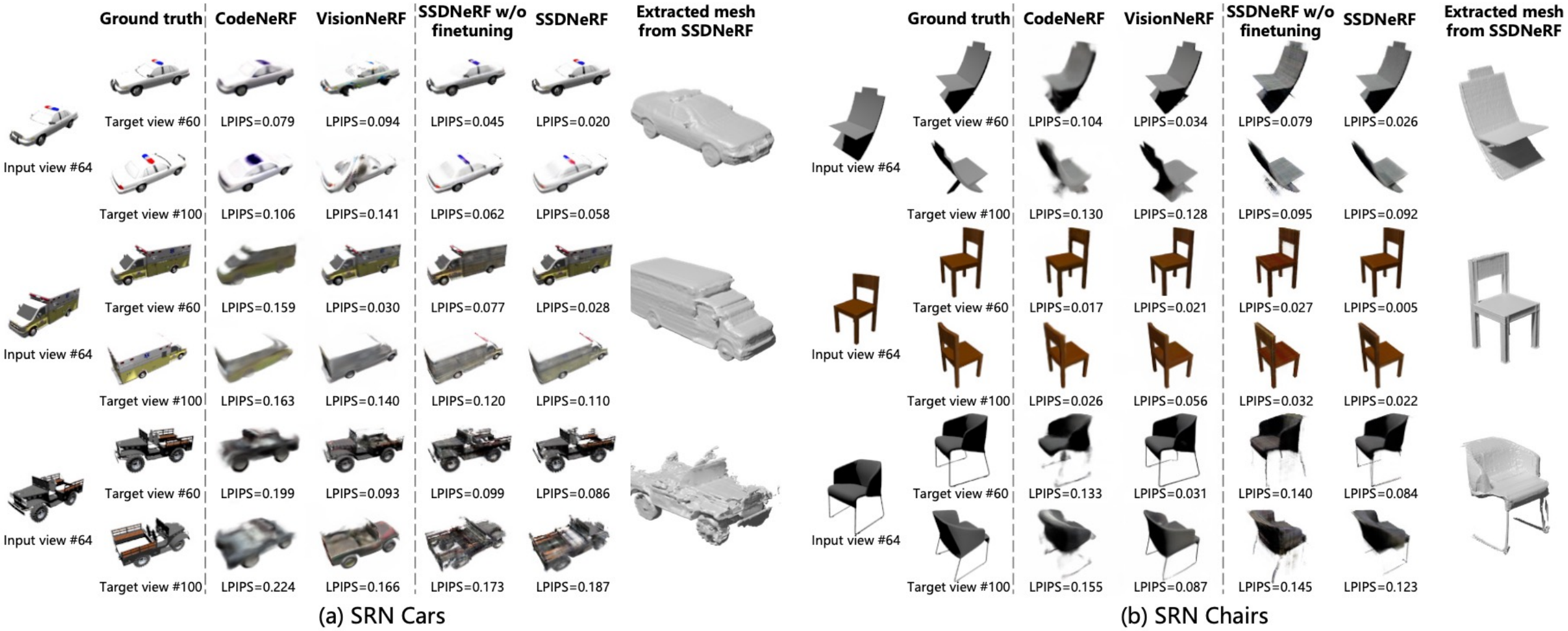
(d) EG3D Cars

(e) SSDNeRF Cars (Ours)

Unconditional Generation

Method	Type	Cars		Tables	
		FID↓	KID/10^{-3}↓	FID↓	KID/10^{-3}↓
Functa [12]	LDM	80.3	-	-	-
π-GAN [4]	GAN	36.7†	-	41.67§	13.82§
EG3D [5]	GAN	**10.46**∗	4.90∗	31.18§	11.67§
DiffRF [29]	LDM	-	-	27.06	10.03
Ours (2-stage)	LDM	16.33±0.93	6.38±0.41	-	-
Ours (1-stage)	LDM	11.08±1.11	**3.47**±0.23	**14.27**±0.66	**4.08**±0.33

Sparse-View NeRF Reconstruction



Sparse-View NeRF Reconstruction

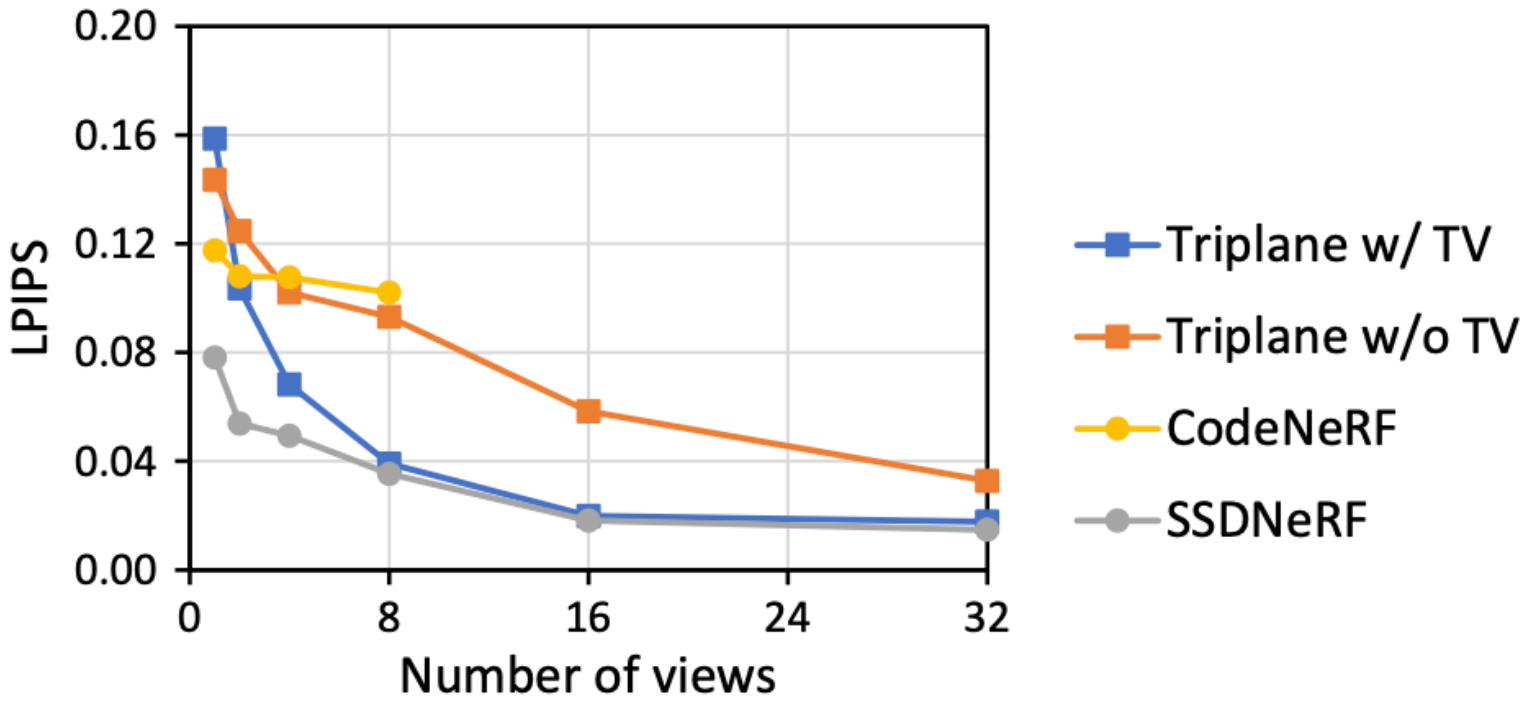
Method	Cars 1-view				Cars 2-view				Chairs 1-view				Chairs 2-view			
	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
3DiM [57]	21.01	0.57	-	**8.99**	-	-	-	-	17.05	0.53	-	**6.57**	-	-	-	-
PixelNeRF [59]	23.17	0.90	0.146‡	59.24†	25.66	**0.94**	-	-	23.72	0.91	0.128‡	38.49†	26.20	0.94	-	-
SRN [48]	22.25§	0.89§	0.129‡	41.21†	24.84§	0.92§	-	-	22.89§	0.89§	0.104‡	26.51†	24.48§	0.92§	-	-
CodeNeRF [24]	**23.80**	**0.91**	0.118∗	56.34∗	25.71	0.93	0.108∗	56.13∗	23.66	0.90	0.106∗	31.65∗	25.63	0.91	0.097∗	29.90∗
VisionNeRF [28]	22.88	**0.91**	0.084	21.31†	-	-	-	-	**24.48**	**0.93**	0.077	10.05†	-	-	-	-
Ours (1-stage)	23.52	**0.91**	**0.078**	16.39	**26.49**	**0.94**	**0.054**	**10.66**	24.35	**0.93**	**0.067**	10.13	**26.94**	**0.95**	**0.055**	**10.85**

Ablation Studies on Test-Time Finetuning

ID	Training	Finetuning	PSNR↑	SSIM↑	LPIPS↓	FID↓
A0	1-stage	Rend + Diff	23.52	0.913	0.078	16.39
A1	2-stage	Rend + Diff	22.83	0.906	0.090	20.97
A2	1-stage	Rend	23.13	0.907	0.088	27.93
A3	1-stage	None	23.07	0.905	0.092	30.95

- 2-stage training and finetuning helps faithfully reconstruct the exact observations

Sparse-to-Dense Reconstruction



- LPIPS scores (lower is better) of novel view synthesis from sparse-to-dense inputs, evaluated on SRN Cars test set.
- we evaluate its novel view synthesis performance with the number of input views varying from 1 to 32.
- SSDNeRF excels in all settings, especially in 1 to 4 views.

Training SSDNeRF on Sparse-View Dataset

- Unconditional Generation
  - A fixed set of only three views are randomly picked from each scene.
  - The model achieves a decent FID of 19.04±1.10 andaKID/10−3 of 8.28±0.60
- Single-View Reconstruction
  - the model achieves an LPIPS score of 0.106, even outperforming most of the previous methods in Table 2 that use the full training set.

NeRF Interpolation

(a) Models trained with early stopping (80K iters) for sparse-view reconstruction

(b) Models trained with long schedule (1M iters) for unconditional generation

- we can sample two initial values $x(T) \sim N(0, I)$, interpolate them using linear interpolation
- **early stopping preserves a smoother prior**, leading to better generalization for sparse-view reconstruction

Outline

- Introduction
- Related Work
- Framework
- Method
- Experiment
- Conclusion

Conclusion

- Proposed a unified approach that employs an **expressive diffusion model** to learn a **generalizable prior** of neural radiance fields.

- New **single-stage** training paradigm that **jointly optimizes** a **NeRF auto-decoder,** a **latent diffusion model** and **scene codes** enabling simultaneous 3D reconstruction and prior learning.

- At test time, we can directly sample the diffusion prior for **unconditional generation**, or **combine it with arbitrary observations of unseen objects** for NeRF reconstruction.