DeepRob

[Student] Lecture 18

by Mani Deep Cherukuri, Claire Chen
NeRF and their Variants
University of Michigan and University of Minnesota

InstantNGP on PROPS dataset

Outline

- Recap
- Limitations of the original NeRF paper
- NeRF Explosion
  - PixelNeRF
  - Instant NeRF
  - Plenoxels
- Main paper
  - Depth-Supervised NeRF
  - Dynamic NeRF - Neural Scene Flow Fields
- Other NeRF applications
  - Robotics
  - Transparency
  - Scene Labelling
  - Articulation

DR

RECAP !!

Recap - What's NeRF?

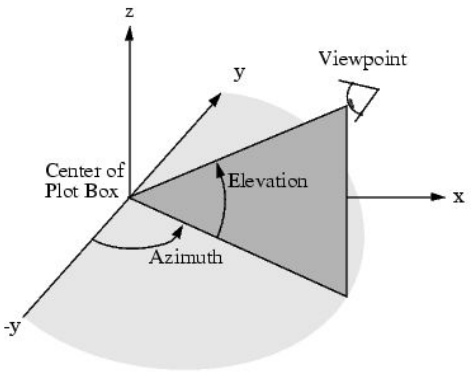
Coordinate sampling (In the world coordinates)

Input Images

COLMAP

Camera Poses
Intrinsic Parameters
Scene Bounds

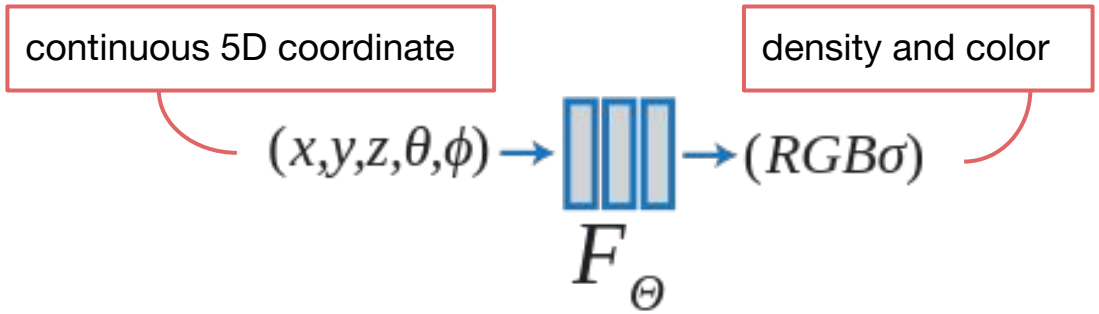
Input for NeRF
(x, y, z, θ, Φ)



https://arxiv.org/pdf/2003.08934.pdf
http://www.ece.northwestern.edu/local-apps/matlabhelp/techdoc/visualize/chview3.html

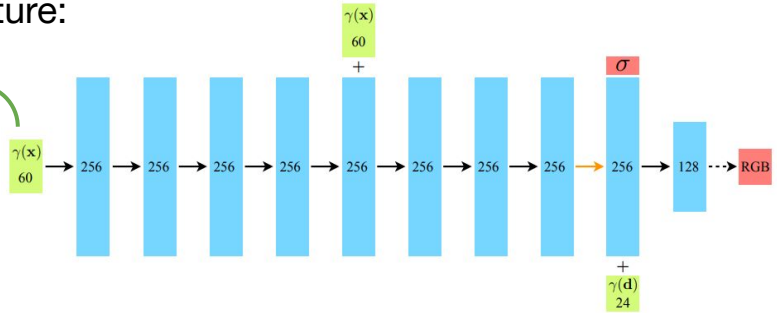
Recap - What's NeRF?

Neural Network (In the field quantities)



Fully-connected network architecture:

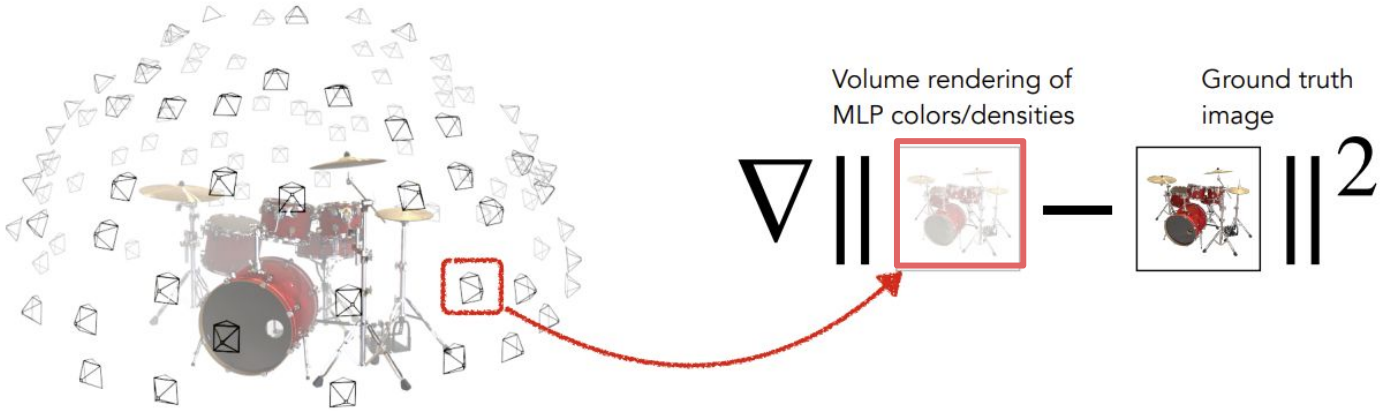
γ: positional encoding



https://arxiv.org/pdf/2003.08934.pdf

Recap - What's NeRF?

Loss

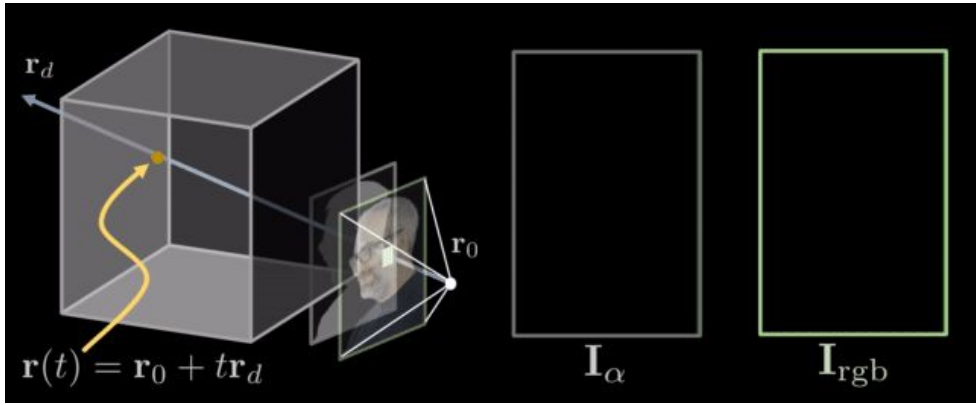


https://arxiv.org/pdf/2003.08934.pdf

Recap - What's NeRF?

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

Ray Marching



https://arxiv.org/pdf/2003.08934.pdf
https://stephenlombardi.github.io/projects/neuralvolumes/

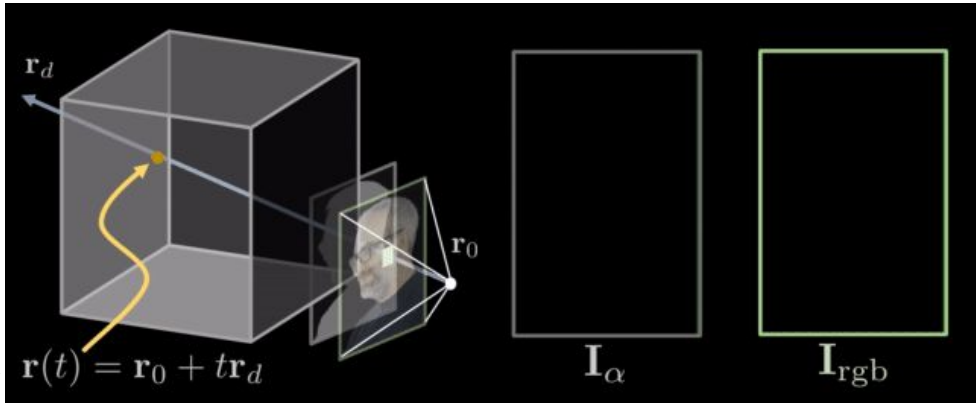
Recap - What's NeRF?

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

Ray color

Camera ray

Ray Marching



https://arxiv.org/pdf/2003.08934.pdf
https://stephenlombardi.github.io/projects/neuralvolumes/

Recap - What's NeRF?

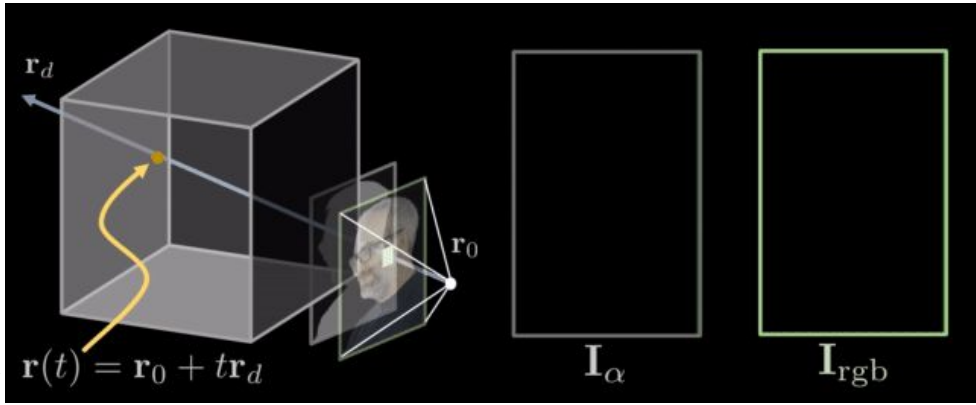
Volume Rendering: $C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt\,,\ \text{where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$

Ray color

Camera ray

bound of the ray

Ray Marching



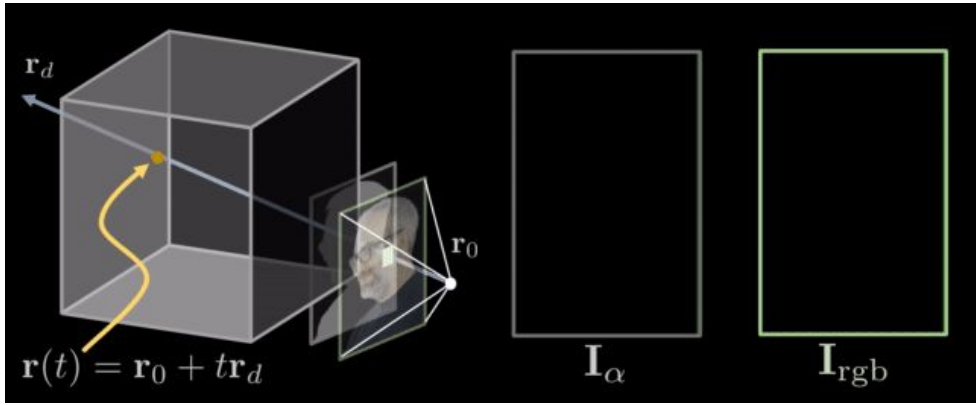
https://arxiv.org/pdf/2003.08934.pdf
https://stephenlombardi.github.io/projects/neuralvolumes/

Recap - What's NeRF?

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

- Ray color
- Camera ray
- bound of the ray
- Accumulated transmittance

Ray Marching



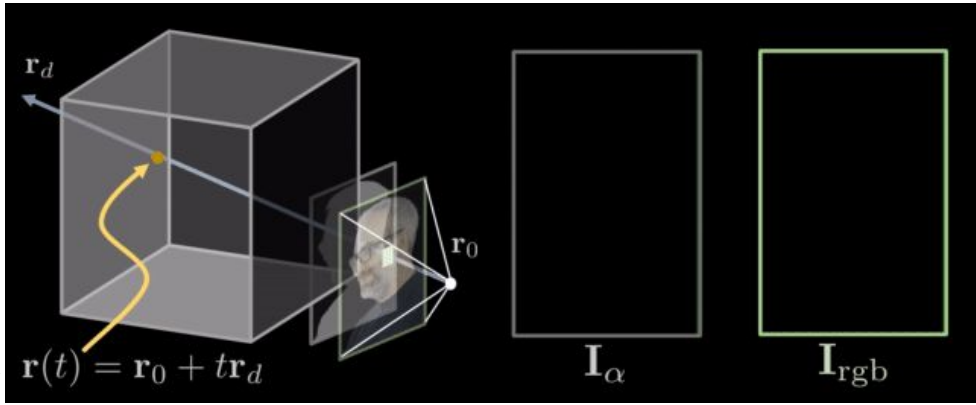
https://arxiv.org/pdf/2003.08934.pdf
https://stephenlombardi.github.io/projects/neuralvolumes/

Recap - What's NeRF?

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

- Ray color
- Camera ray
- bound of the ray
- Accumulated transmittance
- Volume Density
- View-dependent color

Ray Marching

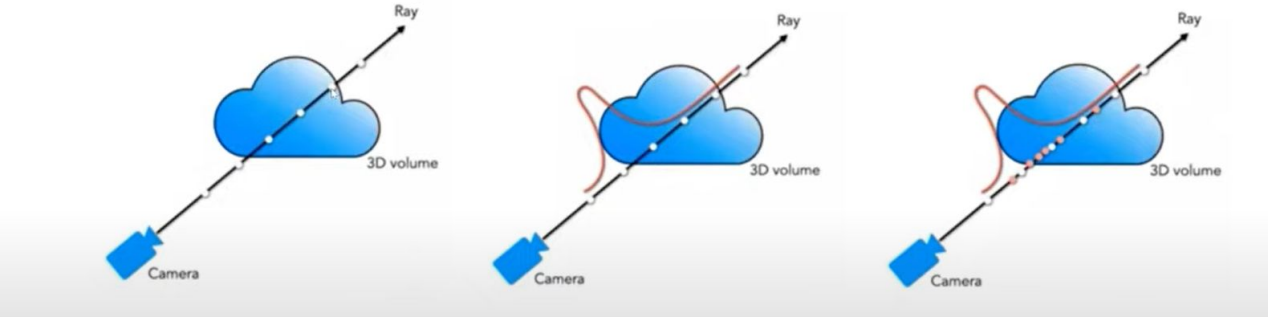


https://arxiv.org/pdf/2003.08934.pdf
https://stephenlombardi.github.io/projects/neuralvolumes/

Recap - What's NeRF?

Hierarchical volume sampling:

$$\hat{C}_c(\mathbf{r}) = \sum_{i=1}^{N_c} w_i c_i \,, \qquad w_i = T_i(1 - \exp(-\sigma_i \delta_i))$$



Actual loss:

$$\mathcal{L} = \sum_{\mathbf{r} \in \mathcal{R}} \left[\left\| \hat{C}_c(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right]$$

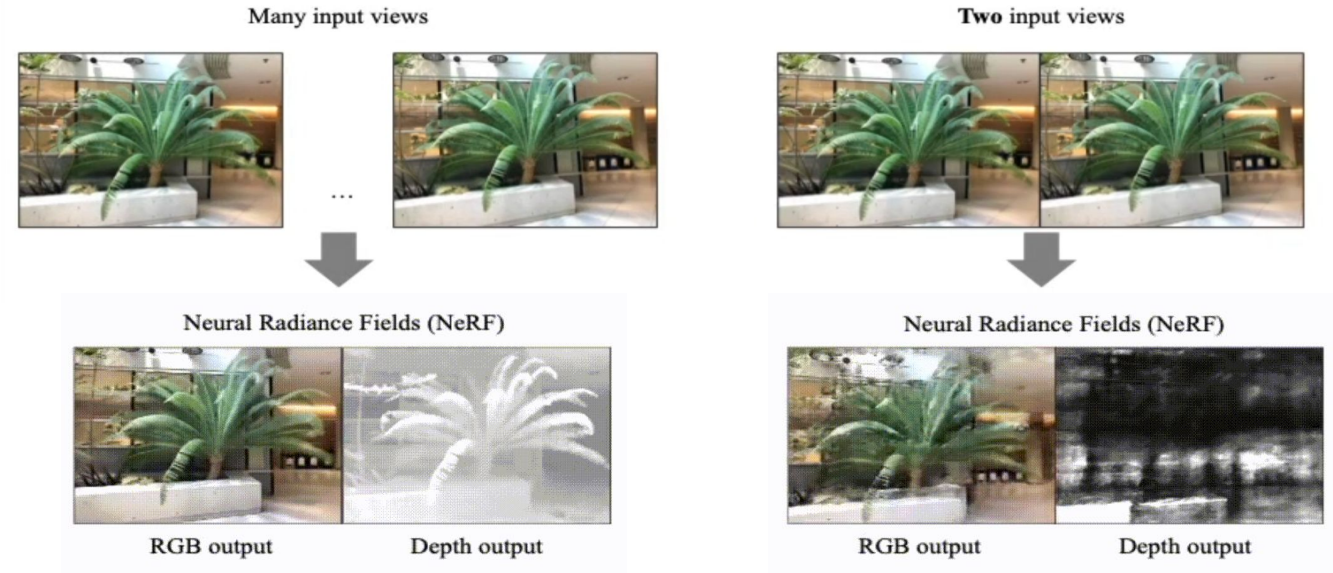
coarse rendering + fine rendering

https://arxiv.org/pdf/2003.08934.pdf
https://www.youtube.com/watch?v=WSfEfZ0ilw4

Limitations

Limitations - Views

- **It requires a significant number of images of the same object.**



https://github.com/dunbar12138/DSNeRF

Limitations - Training Time

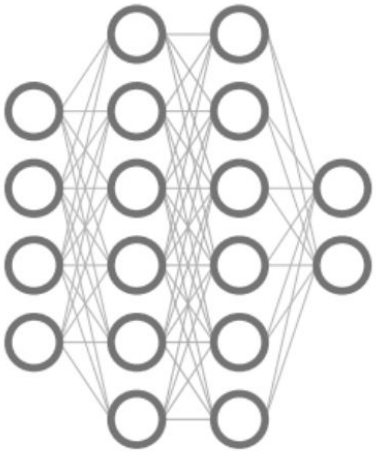
- It requires a significant number of images of the same object.
- **The training time is very long.**

		Training speed	Rendering speed
	Original NeRF	1-2 days	30 sec
	KiloNeRF, cached voxels	1-2 days	1/60 sec
	Learned voxels	10-15 mins	1/15-1/2 sec
	Learned hash maps (Instant NGP)	5 sec - 5 mins	1/60 sec

http://graphics.stanford.edu/courses/cs348n-22-winter/

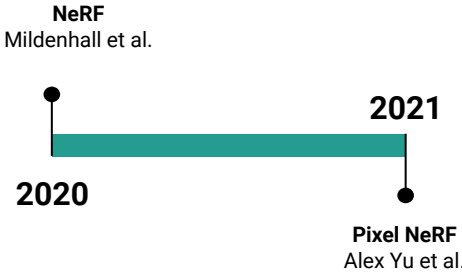
Limitations - Interpretability

- It requires a significant number of images of the same object.
- The training time is very long.
- **Neural Networks are involved making it hard to diagnose issues during training.**

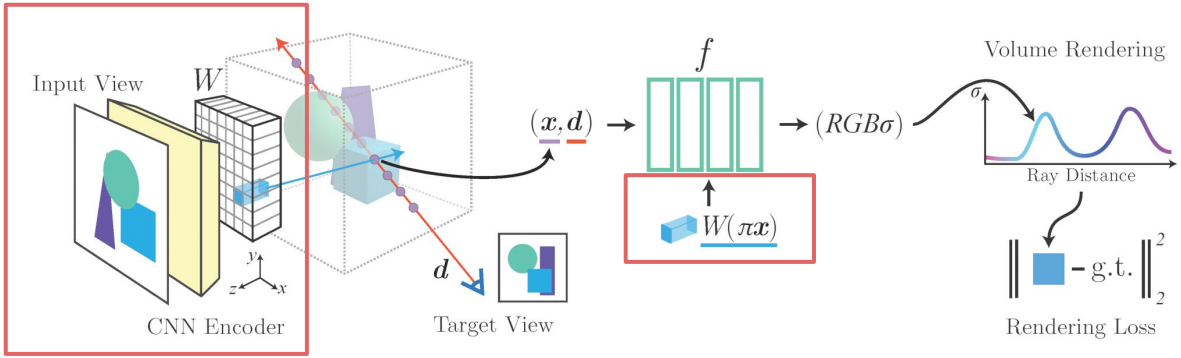


Extending beyond Standard NeRFs !!

Pixel NeRF

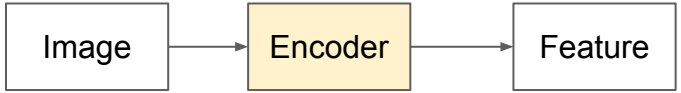


- Original limitation: Optimizing the representation to every scene independently.
- Their solution: Conditioning a NeRF on image inputs in a fully convolutional manner.

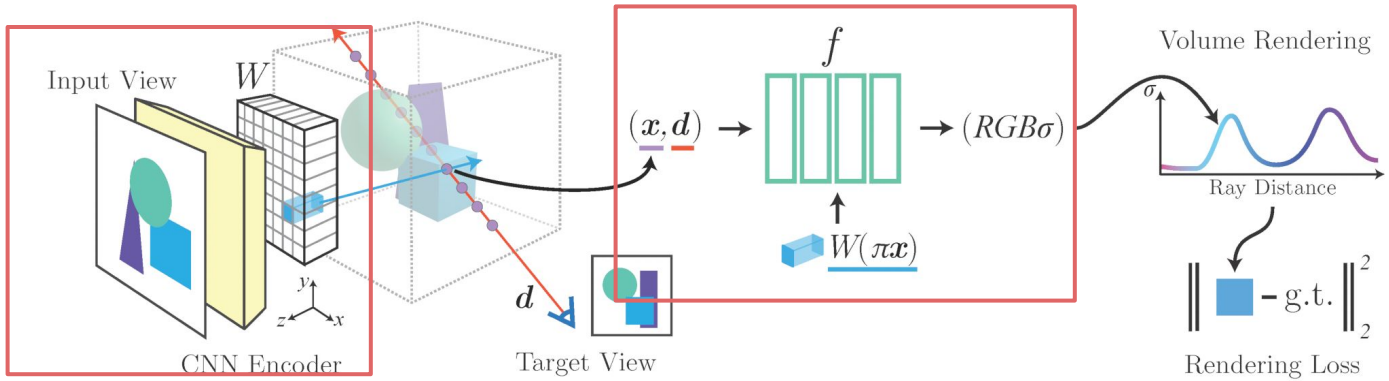


https://alexyu.net/pixelnerf/

Pixel NeRF



$$f(\gamma(\mathbf{x}), \mathbf{d}; \mathbf{W}(\pi(\mathbf{x}))) = (\sigma, \mathbf{c})$$

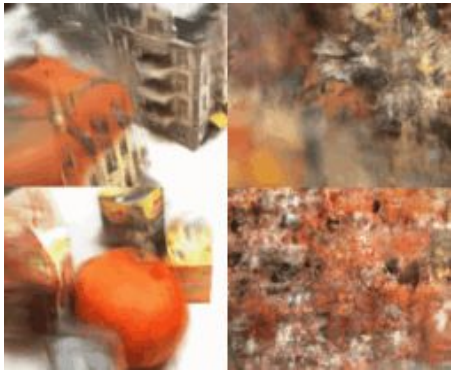


https://alexyu.net/pixelnerf/

Pixel NeRF

3 Input Views

pixel-NeRF NeRF

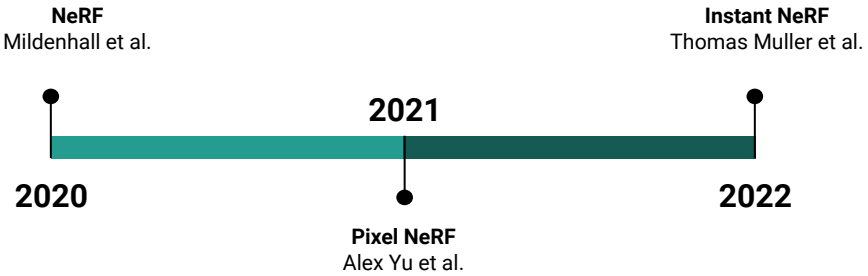


1 Input View

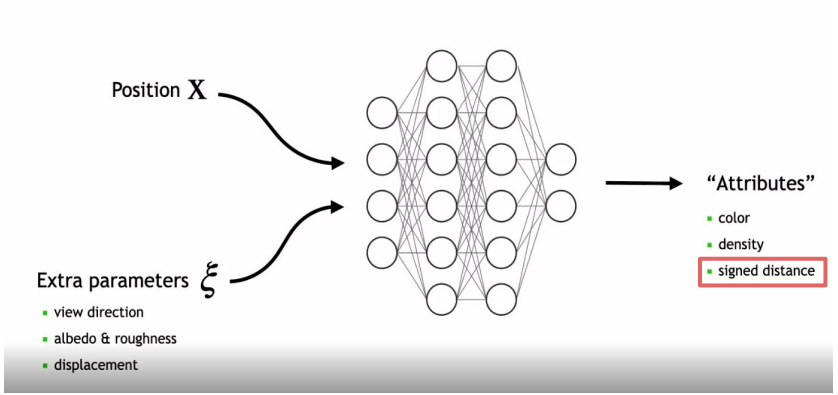
- Features:
  - Learns scene prior
  - Few-view novel-view synthesis

https://alexyu.net/pixelnerf/

Instant ~~NeRF~~ Neural Graphics Primitives



- Original limitation: NeRF is costly to train and evaluate.
- Their solution:
  - Small neural network - fully fused implementation
  - Multiresolution hash encoding

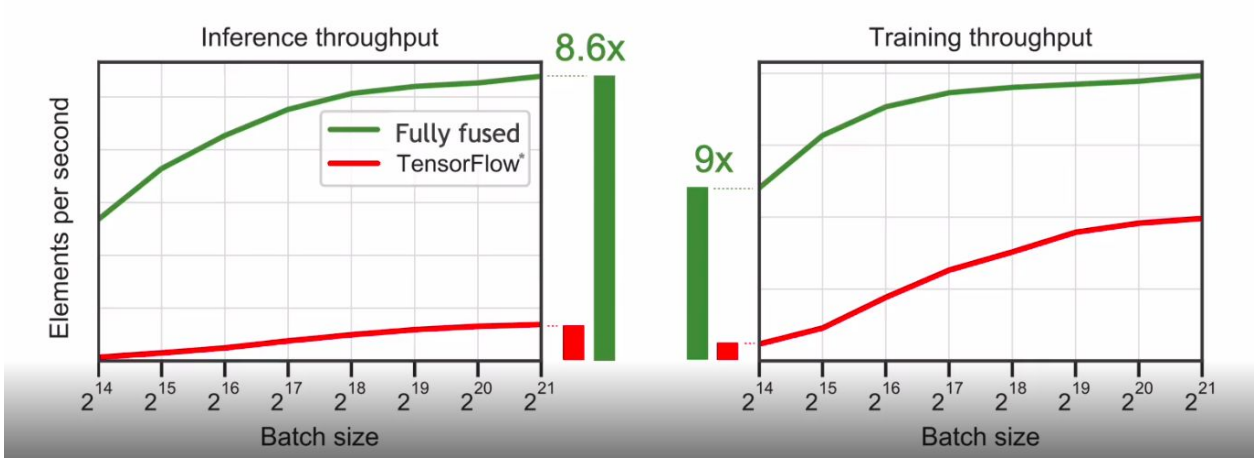


https://nvlabs.github.io/instant-ngp/

Instant ~~NeRF~~ Neural Graphics Primitives

Small neural network - Fully Fused Implementation

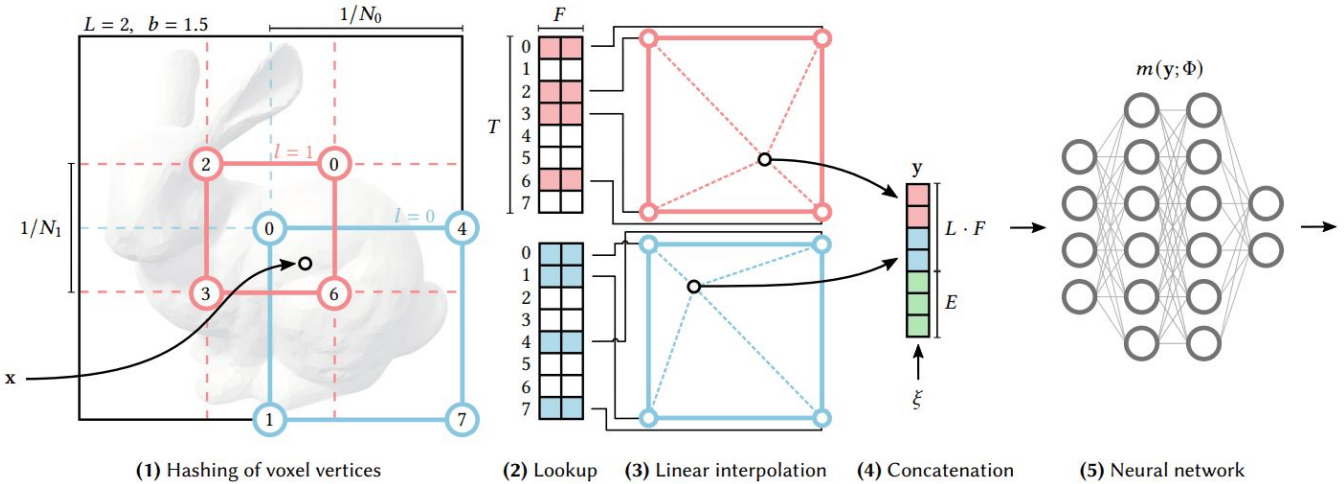
- Entire neural network implemented as single CUDA kernel
- Memory traffic >> Computation



https://nvlabs.github.io/instant-ngp/

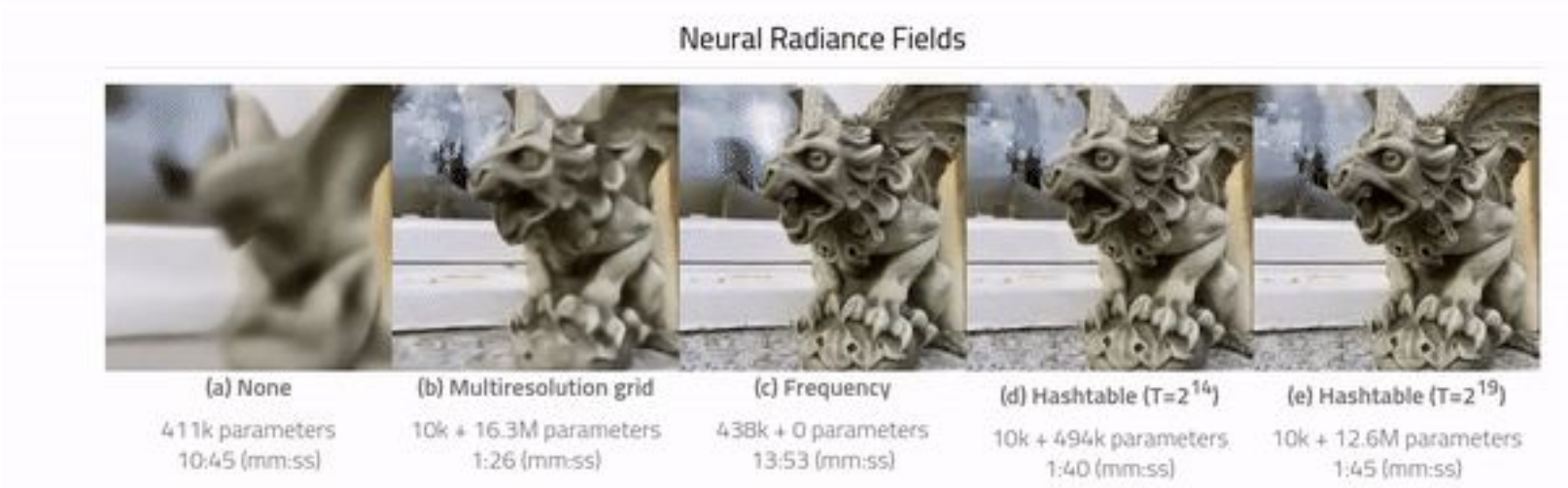
Instant ~~NeRF~~ Neural Graphics Primitives

Multiresolution Hash Encoding



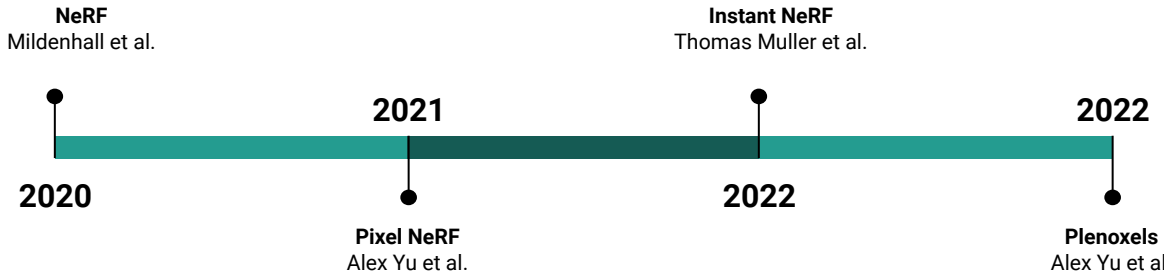
https://nvlabs.github.io/instant-ngp/

Instant ~~NeRF~~ Neural Graphics Primitives

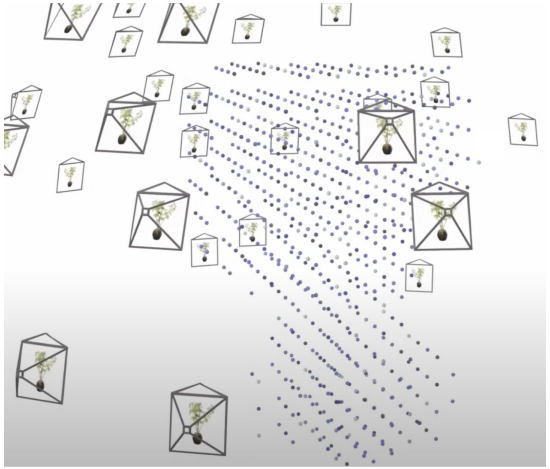


https://nvlabs.github.io/instant-ngp/

Plenoxels

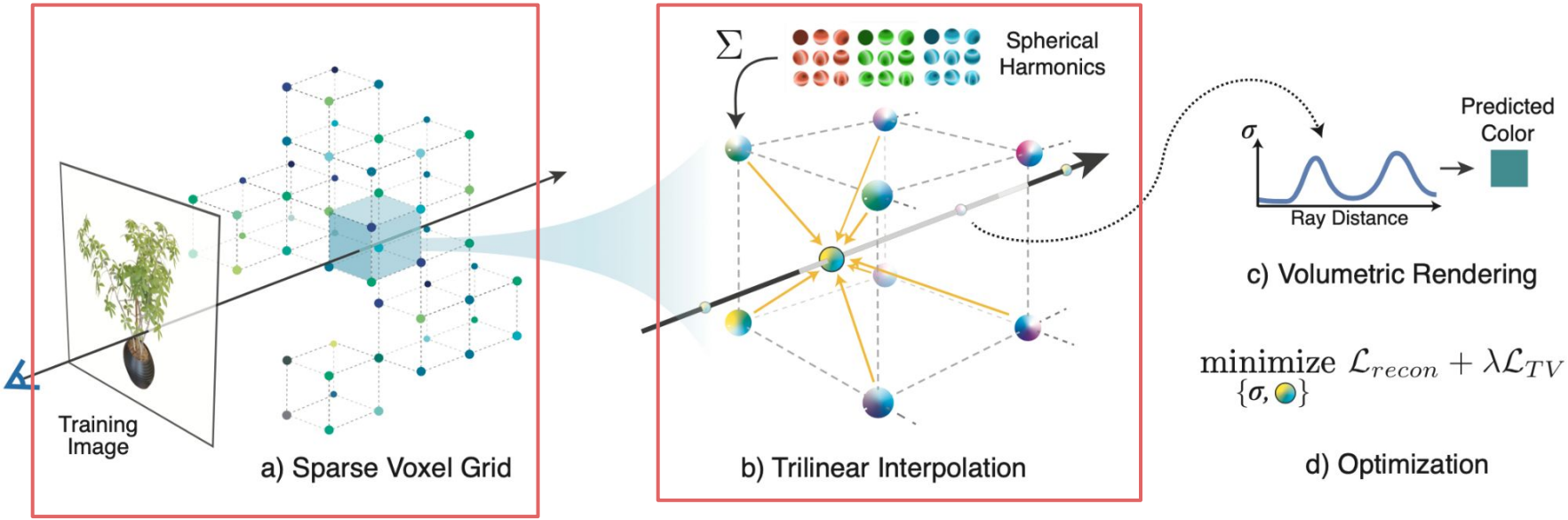


- Original limitation: Lack of transparency to identify issues during training.
- Their solution: Avoid usage of Neural Networks and use voxels instead.
  - speedup of two orders of magnitude compared to NeRF
  - Can be updated in real time with new information



https://arxiv.org/abs/2112.05131

Plenoxels



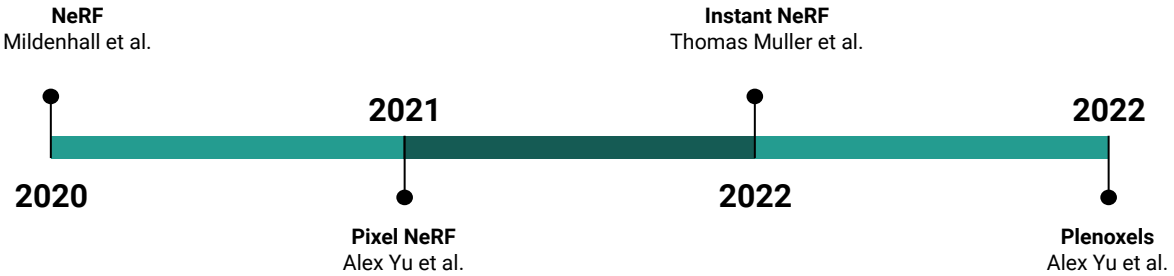
https://arxiv.org/abs/2112.05131

Plenoxels

Key results show that the key component in NeRF is the differential volumetric rendering, but not the neural network

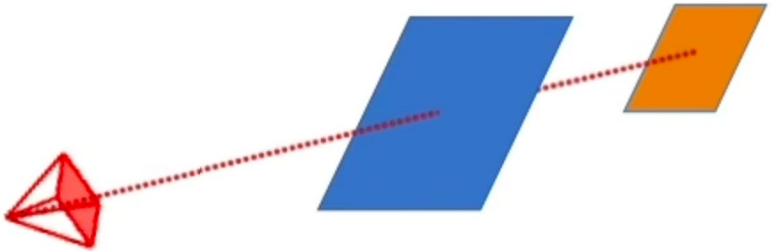
https://arxiv.org/abs/2112.05131

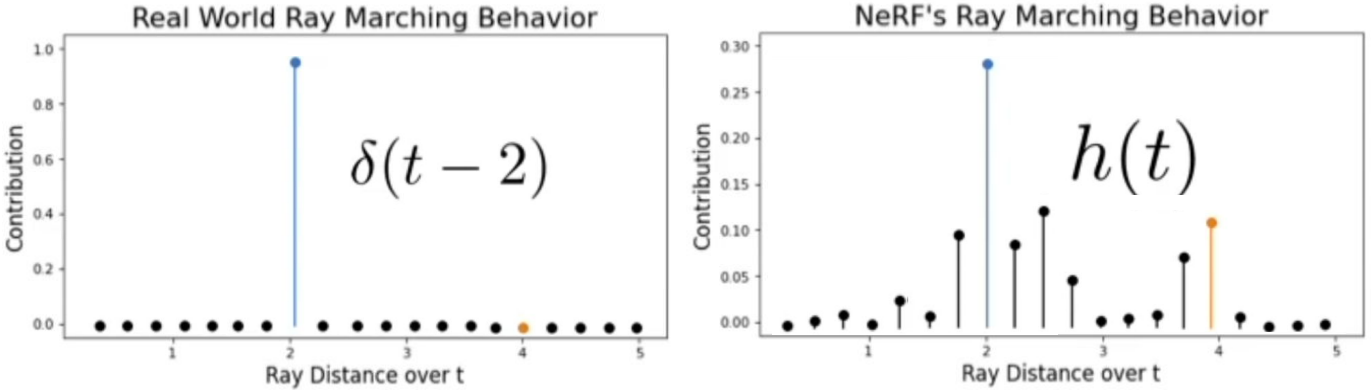
Limitations



	NeRF	Pixel NeRF	Instant NeRF	Plenoxels
Sparse Input Views	❌	✅	❌	❌
Faster Training Time	❌	❌	✅	✅
Interpretability	❌	❌	❌	✅

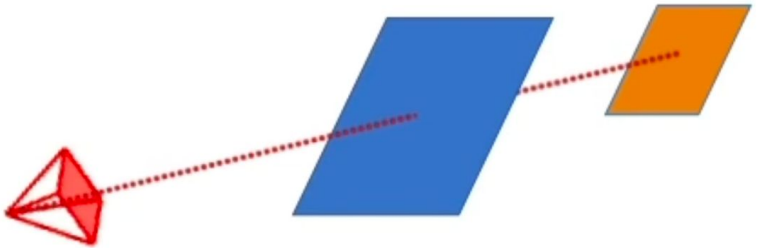
Rendering Mechanics - Regular NeRF

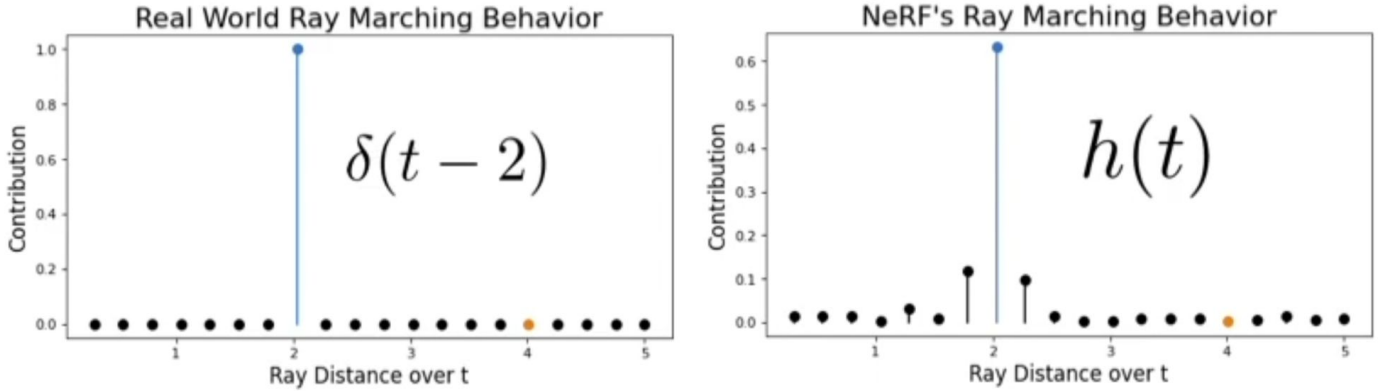




https://www.cs.cmu.edu/~dsnerf/

Rendering Mechanics - DS NeRF





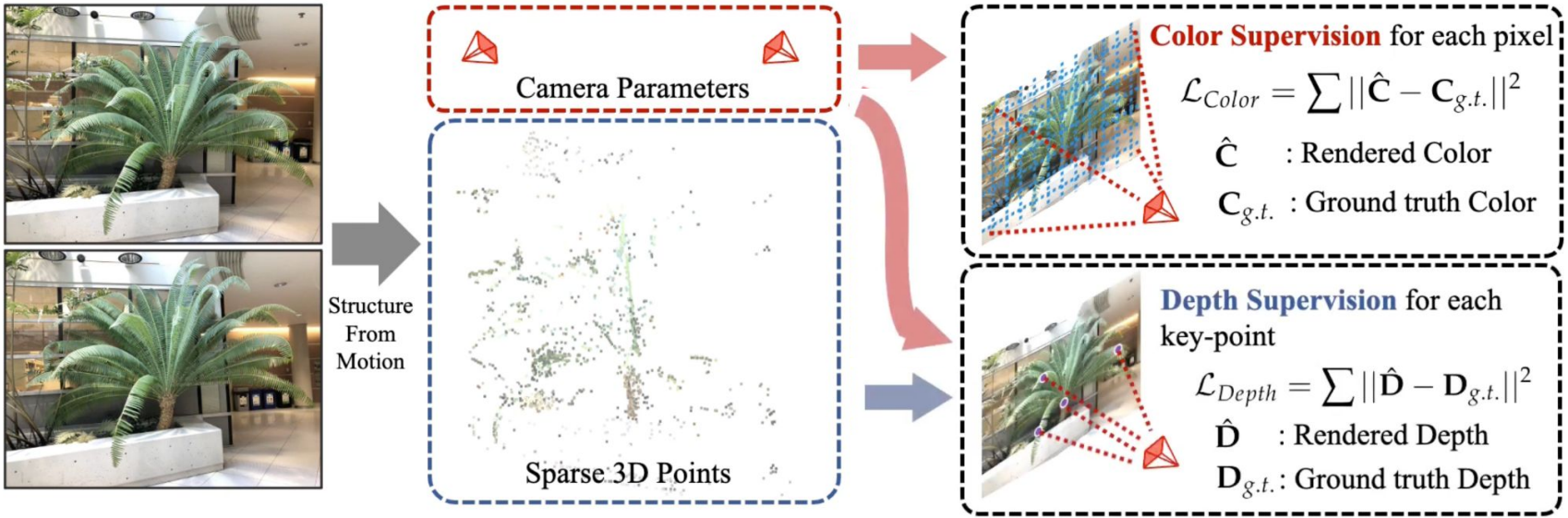
https://www.cs.cmu.edu/~dsnerf/

Depth- Supervised NeRF: Fewer View and Faster Training For Free

Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan.

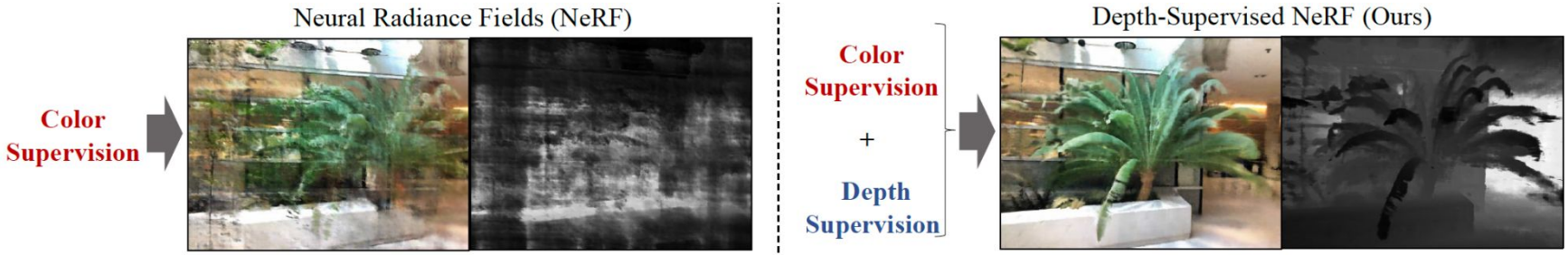
CVPR 2022

Architecture



https://www.cs.cmu.edu/~dsnerf/

Depth-supervised NeRF



https://www.cs.cmu.edu/~dsnerf/

Results

Training Results - 2 Views

NeRF

DS - NeRF

https://www.cs.cmu.edu/~dsnerf/

Training Results- 5 Views

NeRF

DS - NeRF

https://www.cs.cmu.edu/~dsnerf/

Depth Results - 2 Views

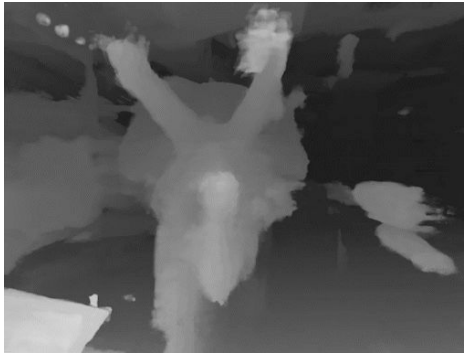
DS - NeRF

NeRF

https://www.cs.cmu.edu/~dsnerf/

Depth Results - 2 Views

DS - NeRF

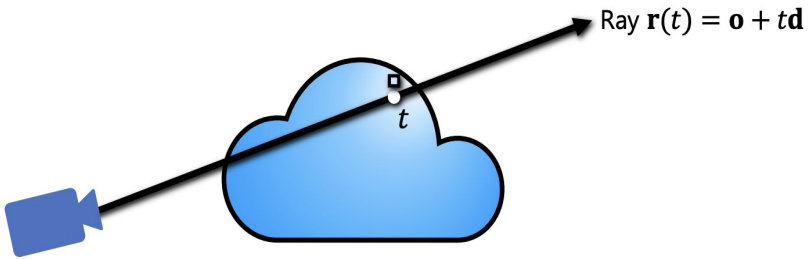


NeRF

https://www.cs.cmu.edu/~dsnerf/

How to achieve this?

Volumetric Rendering



Ray travelling through a scene and at a distance t, we look at its color c (t) and density σ (t)

https://www.cs.unc.edu/~ronisen/

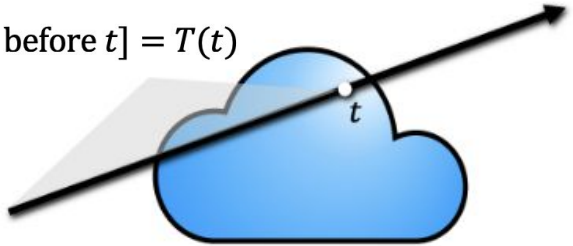
Ray Distribution

Volumetric Rendering: $\hat{\mathbf{C}} = \int_0^\infty T(t)\sigma(t)\mathbf{c}(t)dt,$ where $T(t) = \exp(-\int_0^t \sigma(s)ds)$

$h(t)$:

Continuous probability Distribution

$P[\text{no hits before } t] = T(t)$



Ray Distribution: $\hat{\mathbf{C}} = \int_0^\infty h(t)\mathbf{c}(t)dt = \mathbb{E}_{h(t)}[\mathbf{c}(t)].$

https://www.cs.unc.edu/~ronisen/

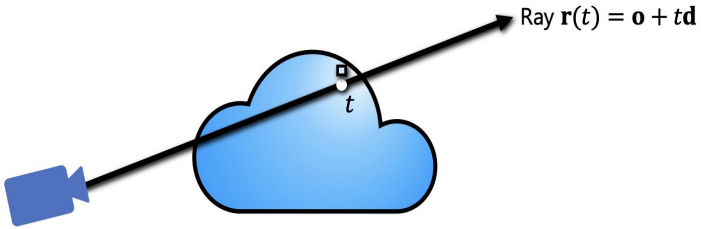
Idealized Distribution

Volumetric Rendering: $\hat{\mathbf{C}} = \int_0^\infty T(t)\sigma(t)\mathbf{c}(t)dt,$ where $T(t) = \exp(-\int_0^t \sigma(s)ds)$

Ray Distribution: $\hat{\mathbf{C}} = \int_0^\infty h(t)\mathbf{c}(t)dt = \mathbb{E}_{h(t)}[\mathbf{c}(t)].$

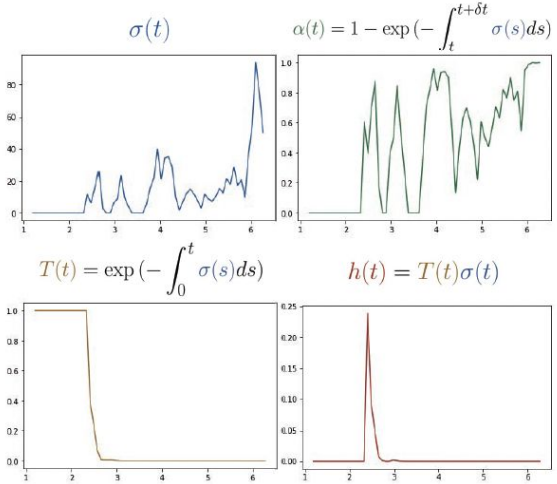
Idealized Distribution: $h(t)$ should be $\delta(t - \mathbf{D})$

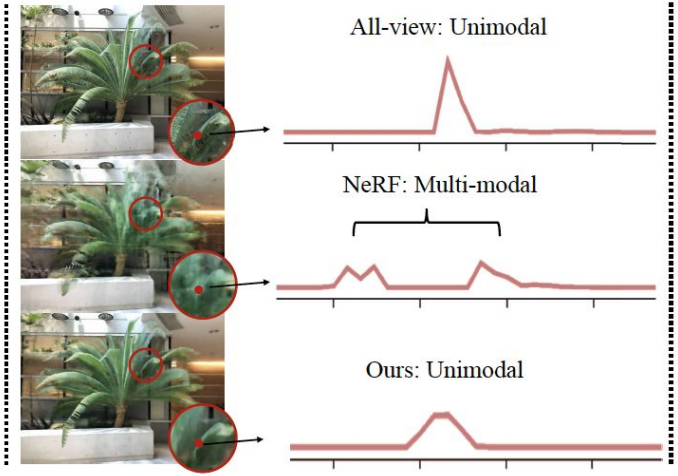
(As discussed in rendering Mechanics)

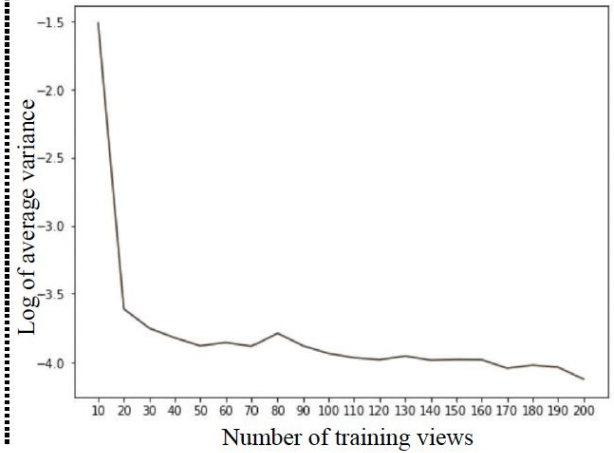


https://www.cs.unc.edu/~ronisen/

NeRF Termination distribution wrt. views

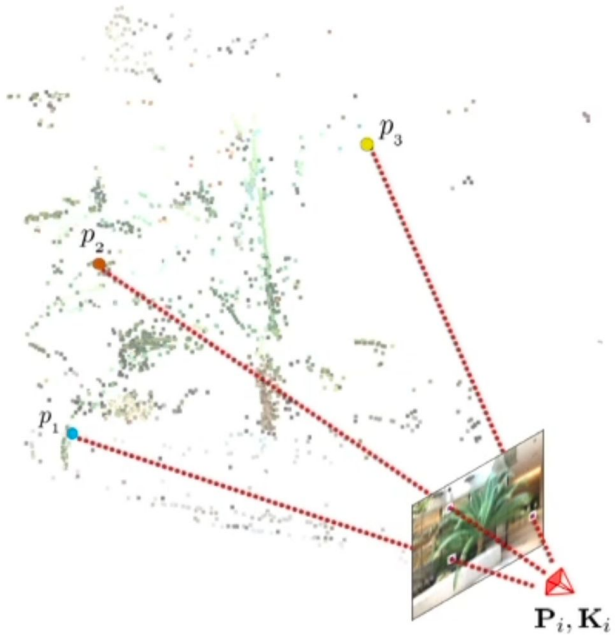






https://www.cs.cmu.edu/~dsnerf/

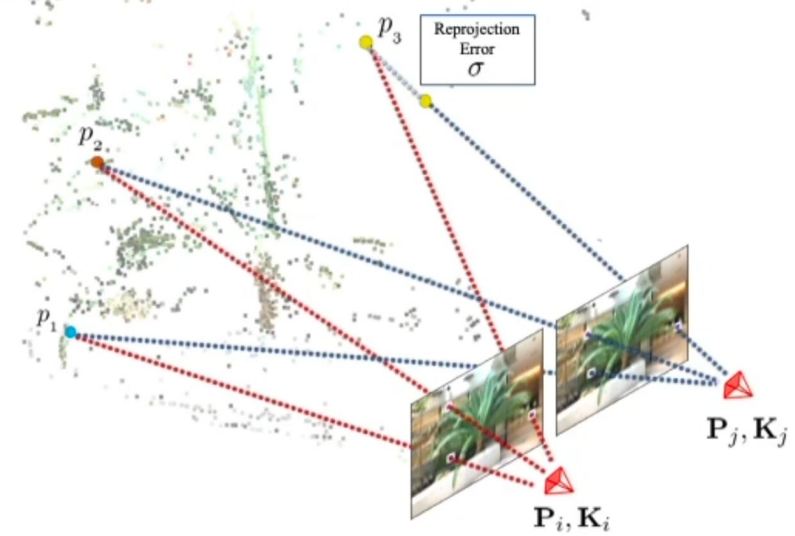
Depth Supervision



https://www.cs.cmu.edu/~dsnerf/

Depth Supervision

$$\mathcal{L}_{\text{depth}} = \text{KL}[\mathbb{N}(\mathbf{D}_{ij}, \hat{\sigma}_i)||h_{ij}(t)]$$



https://www.cs.cmu.edu/~dsnerf/

Loss

Depth Supervision Loss:

$$\mathcal{L}_{Depth} = \sum \mathrm{KL}[\mathbb{N}(\mathbf{D}, \hat{\sigma}) || h(t)]$$

- $\mathbb{N}(\mathbf{D}, \hat{\sigma})$: Normally distributed around the COLMAP-estimated depth
- $h(t)$: Rendered ray distribution

Color Supervision Loss:

$$\mathcal{L}_{Color} = \sum ||\hat{\mathbf{C}} - \mathbf{C}_{g.t.}||^2$$

- $\hat{\mathbf{C}}$: Expected color
- $\mathbf{C}_{g.t.}$: Ground truth color

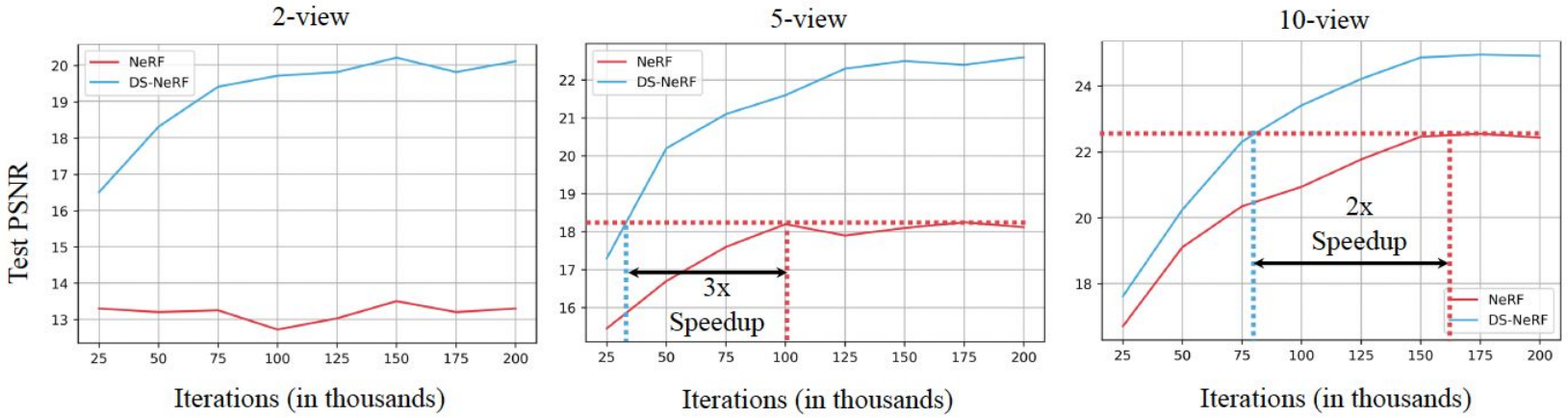
Total Loss:

$$L = L_{Color} + \lambda_D L_{Depth}$$

https://www.cs.cmu.edu/~dsnerf/

Evaluation

Performance Speed



https://www.cs.cmu.edu/~dsnerf/

Evaluation on NeRF Real Dataset

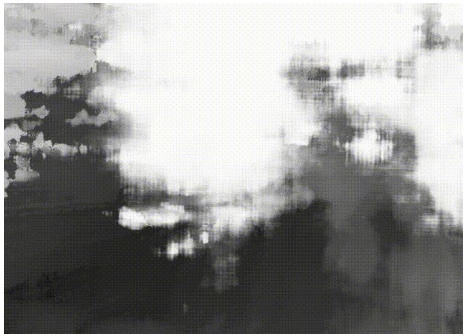
NeRF Real [14]	PSNR↑ 2-view	PSNR↑ 5-view	PSNR↑ 10-view	SSIM↑ 2-view	SSIM↑ 5-view	SSIM↑ 10-view	LPIPS↓ 2-view	LPIPS↓ 5-view	LPIPS↓ 10-view
LLFF	14.3	17.6	22.3	0.48	0.49	0.53	0.55	0.51	0.53
NeRF	13.5	18.2	22.5	0.39	0.57	0.67	0.56	0.50	0.52
metaNeRF-DTU	13.1	13.8	14.3	0.43	0.45	0.46	0.89	0.88	0.87
pixelNeRF-DTU	9.6	9.5	9.7	0.39	0.40	0.40	0.82	0.87	0.81
finetuned	18.2	22.0	24.1	0.56	0.59	0.63	0.53	0.53	0.41
finetuned w/ DS	18.9	22.1	24.4	0.54	0.61	0.66	0.55	0.47	0.42
IBRNet	14.4	21.8	24.3	0.50	0.51	0.54	0.53	0.54	0.51
finetuned w/ DS	19.3	22.3	24.5	0.63	0.66	0.68	**0.39**	0.36	0.38
MVSNeRF	-	17.2	17.2	-	0.61	0.60	-	0.37	0.36
fintuned	-	21.8	22.9	-	**0.70**	0.74	-	**0.27**	**0.23**
fintuned w/ DS	-	22.0	22.9	-	**0.70**	**0.75**	-	0.27	0.25
DS-NeRF									
MSE	19.5	22.2	24.7	0.65	0.69	0.71	0.43	0.40	0.37
KL divergence	**20.2**	**22.6**	**24.9**	**0.67**	0.69	0.72	**0.39**	0.35	0.34

https://www.cs.cmu.edu/~dsnerf/

Scanned Depth Input

Rendered Views

Rendered Depth



NeRF

DS-NeRF with COLMAP

DS-NeRF with RGB-D

https://www.cs.cmu.edu/~dsnerf/

Real World is not Static!!

The Matrix Effect

The matrix effect = free viewpoint + slow motion

https://gfycat.com/miserablerawalpineroadguidetigerbeetle-bullettime-matrix

Space- Time Interpolation

Input

Fixed time

Fixed View

Space - Time Interpolation

https://github.com/zhengqili/Neural-Scene-Flow-Fields

Scene Manipulation via Neural Flow Fields

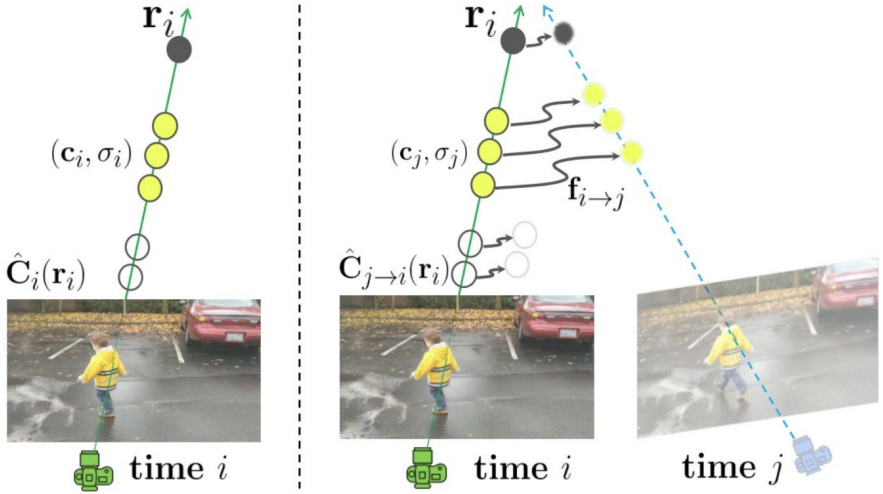
$$(c_i, \sigma_i, F_i, W_i) = F_\Theta^{dy}(x, d, i)$$

Output RGB colour, density at time t.

forward and backward 3D scene flow

Disocclusion weights

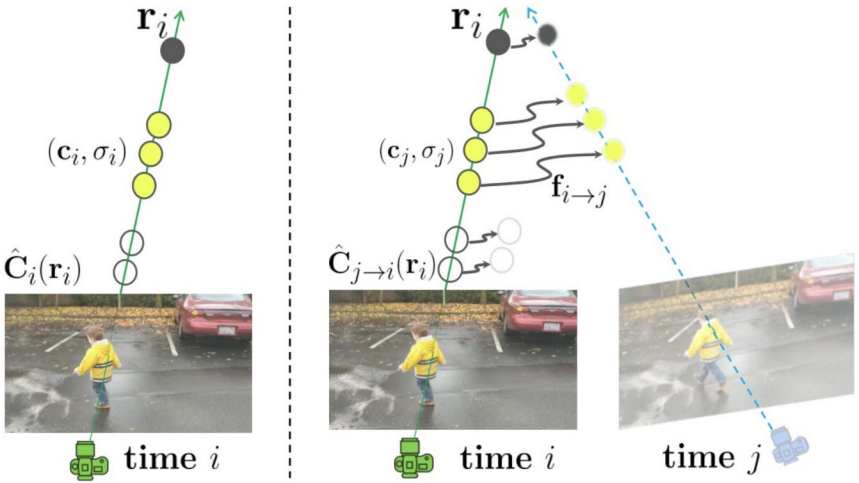
Input position, viewing direction and time index.



$$\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\, \sigma_j(\mathbf{r}_{i\to j}(t))\, \mathbf{c}_j(\mathbf{r}_{i\to j}(t), \mathbf{d}_i) dt$$

$$\text{where } \mathbf{r}_{i\to j}(t) = \mathbf{r}_i(t) + \mathbf{f}_{i\to j}(\mathbf{r}_i(t)).$$

Scene Manipulation via Neural Flow Fields



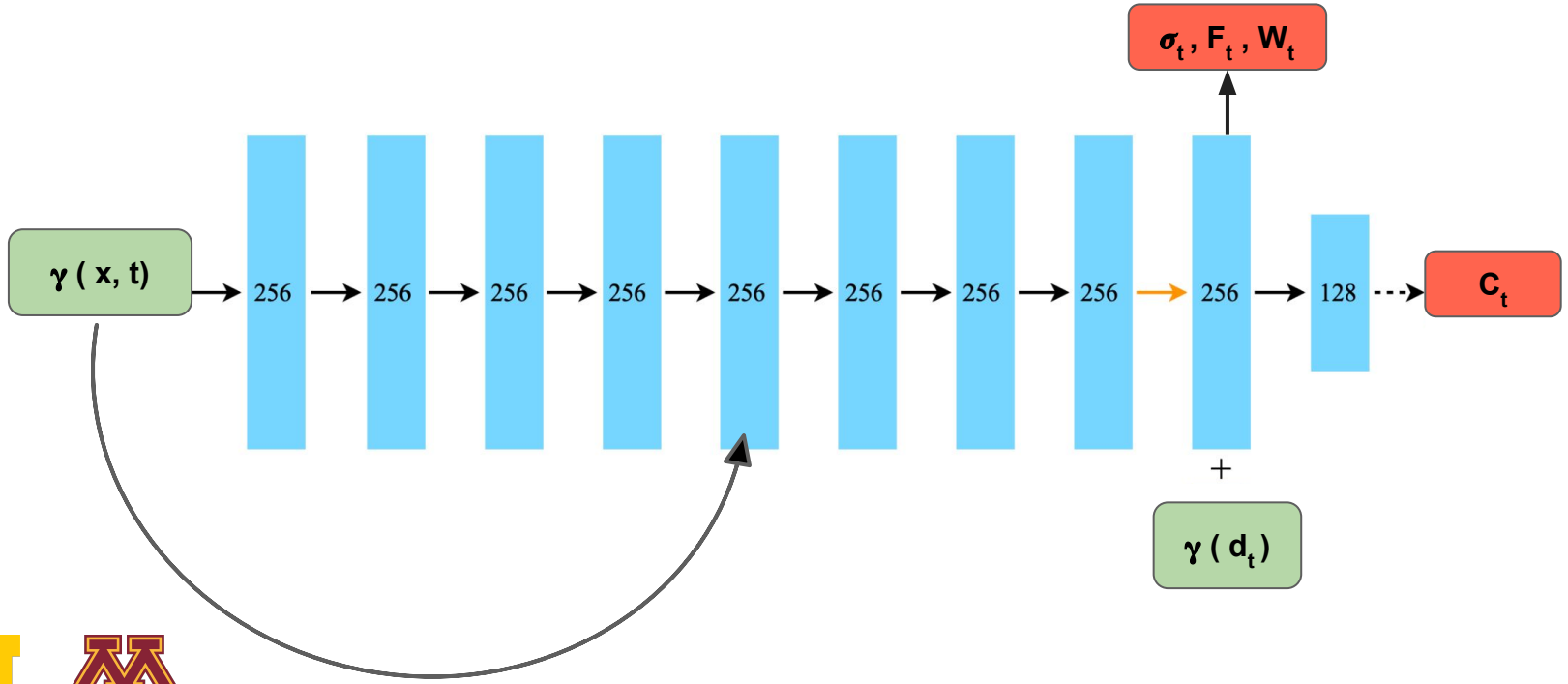
Temporal photometric Consistency Loss:

$$\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i} \sum_{j \in \mathcal{N}(i)} ||\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2$$

3d Scene Flow Cycle Consistency Loss:

$$\mathcal{L}_{\text{cyc}} = \sum_{\mathbf{x}_i} \sum_{j \in i \pm 1} w_{i\to j} ||\mathbf{f}_{i\to j}(\mathbf{x}_i) + \mathbf{f}_{j\to i}(\mathbf{x}_{i\to j})||_1$$

Architecture



Dynamic NeRF

https://arxiv.org/abs/2103.02597

Dynamic NeRF

D-NeRF: Neural Radiance Fields for Dynamic Scenes
https://www.albertpumarola.com/research/D-NeRF/index.html

Dynamic View Synthesis from Dynamic Monocular Video
https://free-view-video.github.io/

Nerfies: Deformable Neural Radiance Fields
https://nerfies.github.io/

NeRF & Robotics

Robotics

Learning Multi-Object Dynamics with Compositional Neural Radiance Fields

https://dannydriess.github.io/compnerfdyn/

Robotics

NeRF2Real: Sim2real Transfer of Vision-guided Bipedal Motion Skills using Neural Radiance Fields

https://sites.google.com/view/nerf2real/home

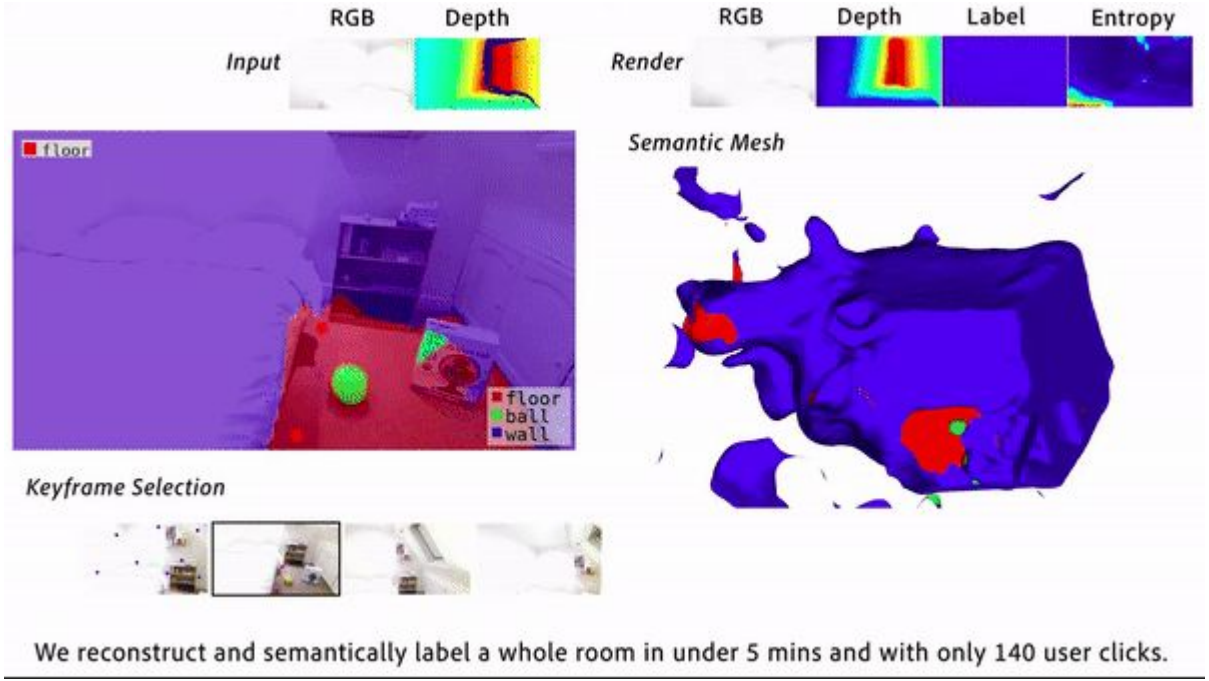
Robotics

- 3D Neural Scene Representations for Visuomotor Control
  https://3d-representation-learning.github.io/nerf-dy/
- ACID: Action-Conditional Implicit Visual Dynamics for Deformable Object Manipulation
  https://arxiv.org/pdf/2203.06856.pdf
- Vision-Only Robot Navigation in a Neural Radiance World
  https://arxiv.org/abs/2110.00168

NeRF Labeling

NeRF Labelling

iLabel: Interactive Neural Scene Labelling

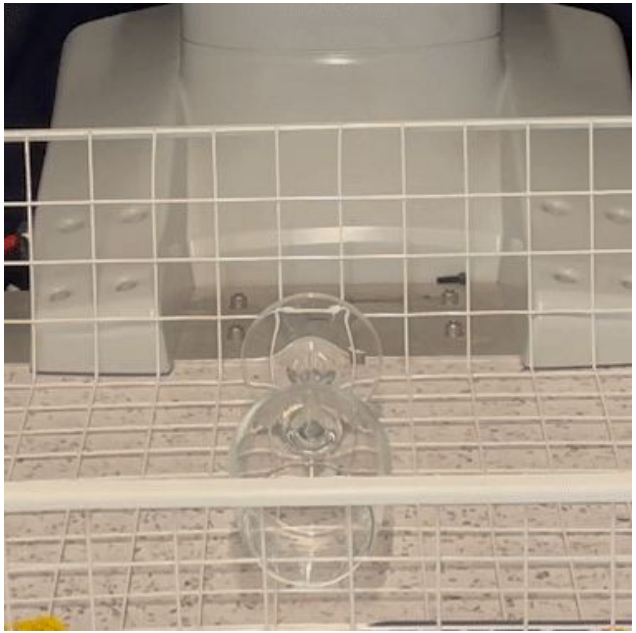


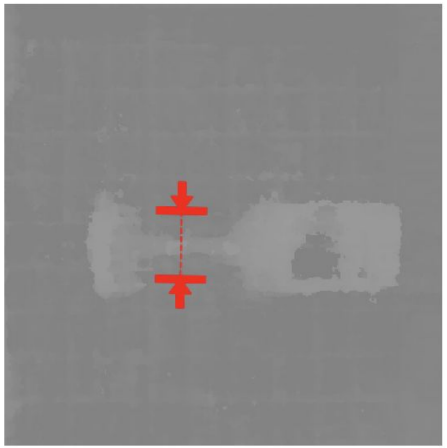
https://shuaifengzhi.com/Semantic-NeRF/

Transparency NeRF

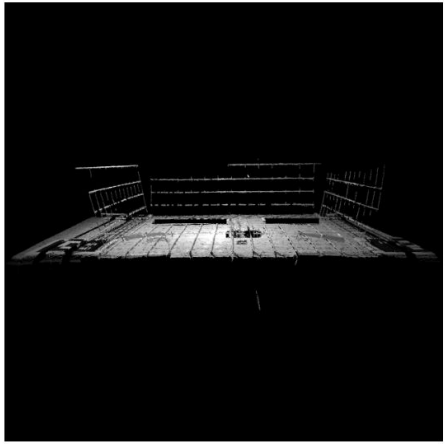
Transparency

Dex-NeRF: Grasping Transparent Objects using NeRF, Ichnowski et al., CoRL 2021





Dex-NeRF + Grasp



PhoXi point cloud

https://arxiv.org/abs/2110.14217

Articulated NeRF

Articulation

- NARF22: Neural Articulated Radiance Fields for Configuration-Aware Rendering
  https://arxiv.org/abs/2210.01166
- Neural Articulated Radiance Field
  https://arxiv.org/abs/2104.03110
- CLA-NeRF: Category-Level Articulated Neural Radiance Field
  https://arxiv.org/abs/2202.00181

Summary

- Limitations of NeRF
  - Training time, number of input views required, and interpretability
- Rendering Mechanics
  - Depth plays a crucial role in accurately rendering 3D scenes
- Depth Supervision NeRF
  - improved architecture that uses depth supervision to improve the quality of 3D models to generate high quality results with sparse views and achieve faster training times.
- Dynamic NeRF
  - Used for the reconstruction of 3D scenes that change over time.
  - Neural Scene Flow Fields provide a more accurate representation of scene manipulation
- NeRF Variants
  - Pixel NeRF, Instant NeRF, and Plenoxels are other variants of NeRF which address some of the limitations of the original NeRF
- Applications in Robotics
  - Used in labeling scenes, detecting transparent objects, pose estimation, and more.

DeepRob

[Student] Lecture 18

by Mani Deep Cherukuri, Claire Chen
NeRF and their Variants
University of Michigan and University of Minnesota

InstantNGP on PROPS dataset