Towards to fast neural rendering

Introduction to recent methods of efficient and fast neural rendering

Sogang University
Vision & Display Systems Lab, Dept. of Electronic Engineering

Presented By
Hosung Son

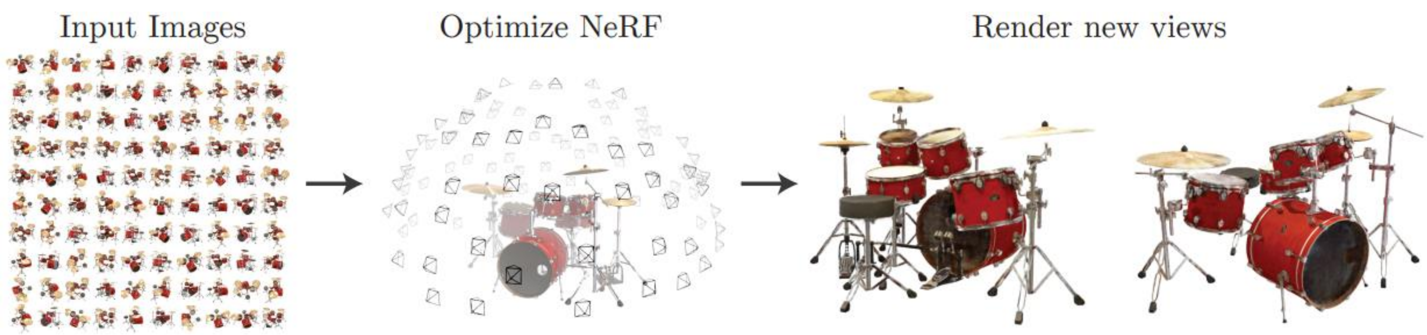
Contents

- Introduction
  - What is NeRF?
  - Challenges of NeRF
- Papers
  - Knowledge Distillation
    - R2L [ECCV2022]
  - Ray sampling
    - EfficientNeRF [CVPR 2022]
- Conclusion

What is NeRF?

- Definition
  - Novel view synthesis with neural network by construct radiance field.
  - Implicit function mapping between 5D input data and 4D Output data.
    - Explicit(양함수) vs. Implicit(음함수) ?
      - $S_{explicit} = \left\{ \begin{pmatrix} x \\ y \\ f_{explicit}(x,y) \end{pmatrix} \middle| \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2 \right\}$
      - Ex) $f(x,y) = x^2 - y + 1$
      - $S_{implicit} = \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^2 \middle| f_{implicit}(x,y,z) = k \right\}$
      - Ex) $x^y - 2y + e^z = 3$

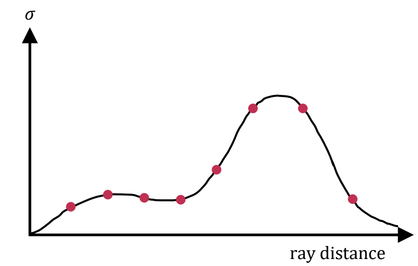
Examples of view synthesis by NeRF



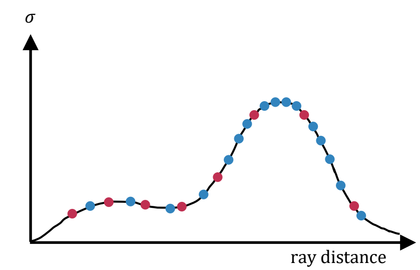
Novel view synthesis process with NeRF

What is NeRF?

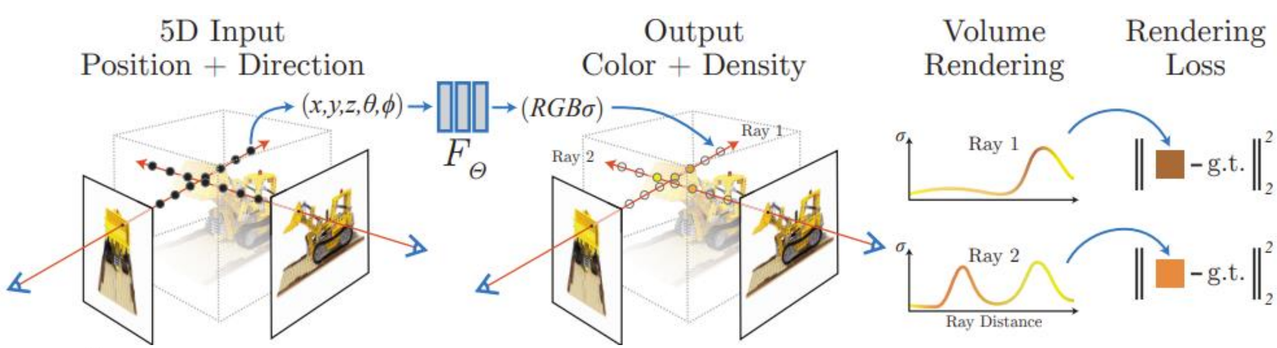
- Pipeline of NeRF
  - Data loading
    - Images, Camera poses(extrinsic), Intrinsic, Render poses(extrinsic), Sampling depth(near, far)
  - Coarse sampling
    - 2D Pixels in image plane → 3D Points in world coordinates
    - Coarse sample N_c(64) points along a query ray
  - Fine sampling
    - Additional sample N_f(128) points based on queried density value(σ)
      - Get density value passing through network query function
      - High density → Dense sampling



Coarse sampling



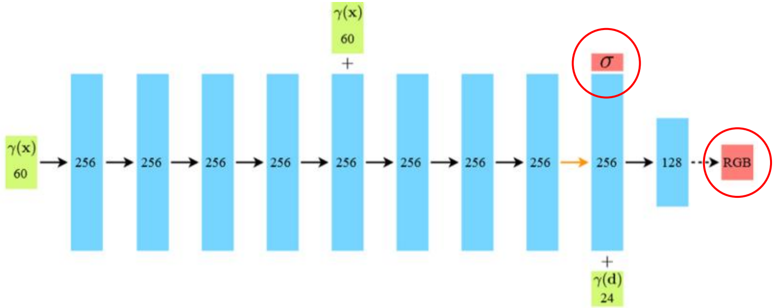
Fine sampling



What is NeRF?

- Pipeline of NeRF
  - MLP
    - 10 MLP Layers
    - *Positional Encoding*(γ)
      - It helps model represent high frequency region
      - L=10 for x, L=4 for d

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



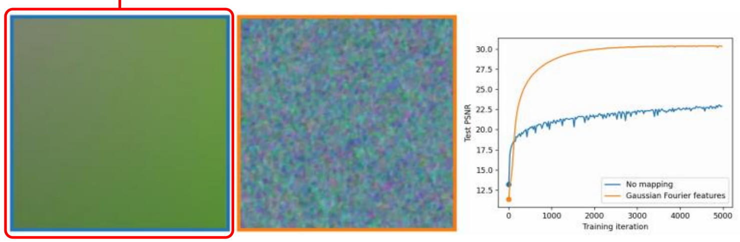
MLP networks of NeRF

  - Rendering

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

Volume rendering equation

FC based DNN are biased to learn low frequencies faster



Comparison of model learning results based on absence of PE

  - Loss function

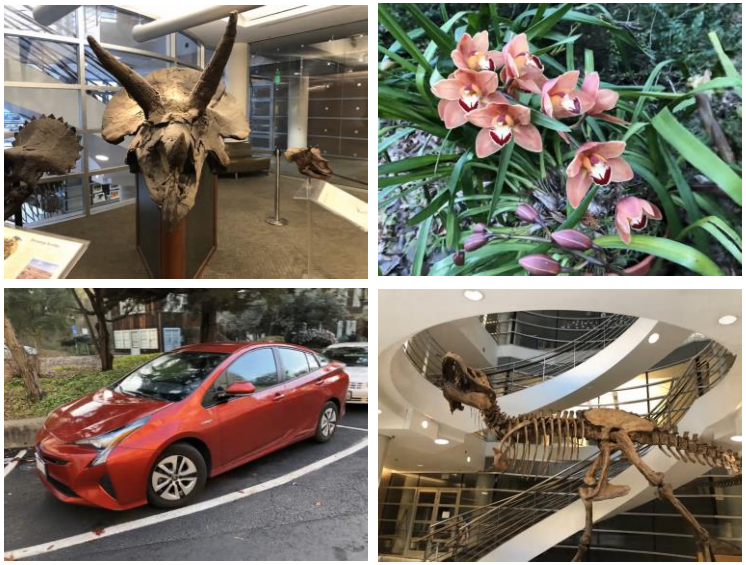
$$\mathcal{L} = \sum_{\mathbf{r} \in \mathcal{R}} \left[\left\| \hat{C}_c(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right]$$

L2 Loss function (Coarse + Fine)

Challenges of NeRF

- Limitation of generalization
  - Struggle to synthesize unseen scenes
    - NeRF model has low inductive bias
      - FC-layers just mapping 5D input to 4D output.
  - Time consumption*
    - Training(optimizing) time
      - Model size: 5MB
      - 30~50 hours for NeRF-*synthetic* dataset .
      - 20~36 hours for *forward-facing scene* dataset.
    - Testing(rendering) time
      - 0.023 fps(43s) for NeRF-*synthetic* dataset.
      - 0.018 fps(55s) for *forward-facing scene* dataset.

NeRF-*synthetic* dataset



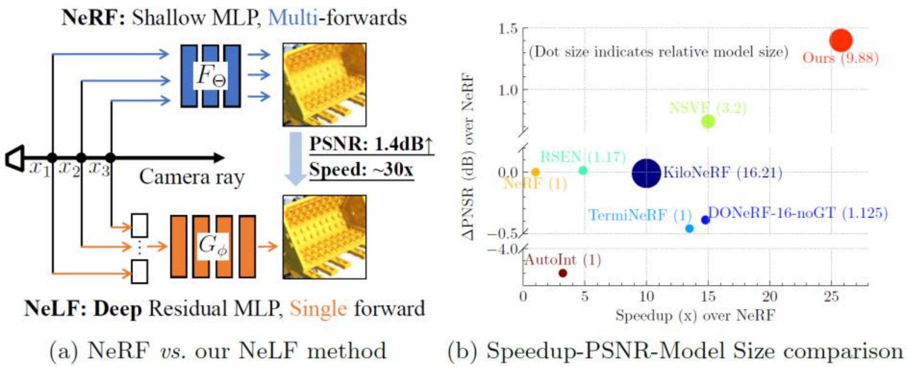
***forward-facing* dataset**

Challenges of NeRF

- Why reducing time consumption is important?
  - If we want to serve the “NeRF” to people as a program or mobile application, people wouldn’t ready for 4~50 seconds to get synthesized view images of their own contents.
  - Training(optimization) time is related to how many times be taken to reconstruct radiance field of the contents.
  - Testing(Inference) time is related to how fast the model can render novel view images from reconstructed radiance field.
- How can we reduce the time consumption of NeRF?
  - Number of samples along a ray → EfficientNeRF
  - Number of rays for rendering
  - Size of the model
  - Computational complexity → R2L

Papers – Knowledge Distillation

- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]

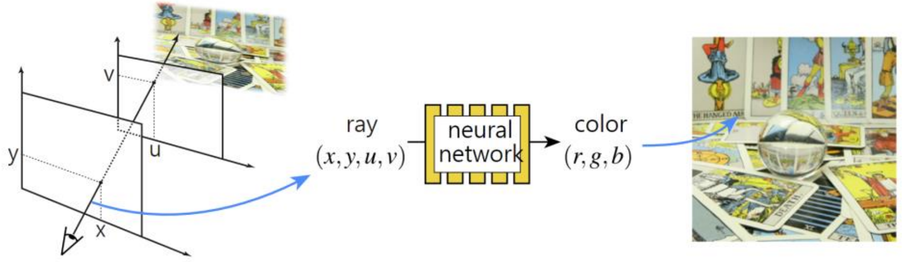


R2L structure and Visualization of performance improvement

- Contribution
  - *First* attempt to improve the rendering efficiency via network architecture optimization.
  - Proposed an <u>effective training</u> strategy by distilling knowledge from a pre-trained NeRF.
  - Achieved 26 ~ 35× FLOPs, 28 ~ 31× inference time reduction over the original NeRF.
- Key methods
  - From "NeRF" to "NeLF" knowledge distillation
  - Long skip connection and repeated deep residual MLP blocks architecture
  - Training hard examples

Papers – Knowledge Distillation

- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]
  - Light field
    - Advantages
      - One evaluation per pixel → methodologically straightforward for novel view synthesis.
      - NeLF outputs the RGB color of a queried ray while NeRF outputs the radiance of a sampled point along a ray.

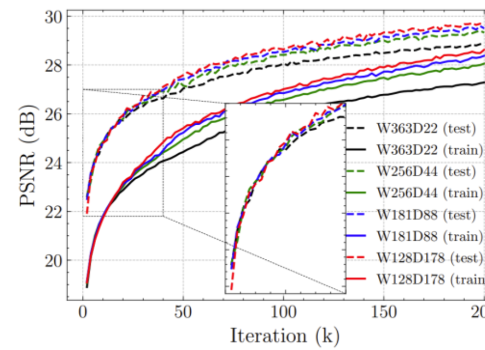


Neural light field rendering

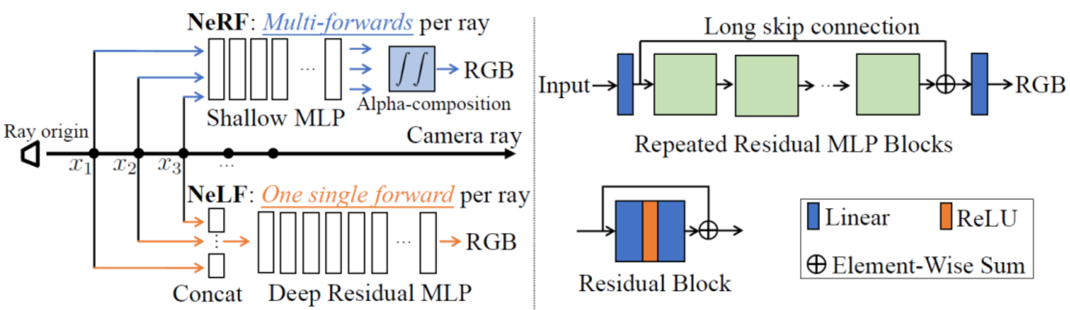
- Drawbacks
  - NeLFs tend to considerable storage costs and harder to learn than radiance field.
    - ✓Compute concatenate samples and render them in a single forward pass.
    - ✓Radiance at neighbor spaces does not change dramatically given the radiance field while two neighbor rays can point to starkly different colors because of occlusion.
  - They are hard to achieve a full 360° representation without concatenating multiple light fields.

Papers – Knowledge Distillation

- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]
  - Deep residual MLP structure
    - To learn light field, network optimizes the concatenated high-dimensional samples along a ray.
    - NeLF should have deep layer neural network architecture.
    - Bunch of networks
      - 6M FLOPs per ray: W181D88, W256D44, W363D22 (For ablation study: W181D88)
      - 12M FLOPs per ray: W256D88 best

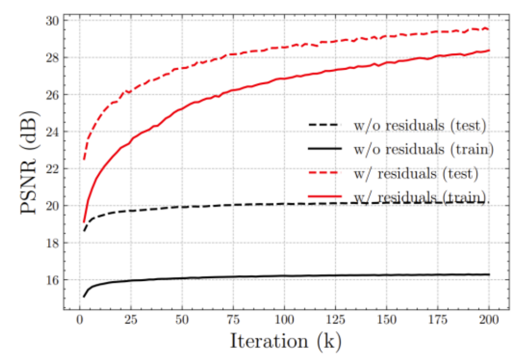


Depth-width tradeoff



(a) NeRF (top) *vs.* NeLF (ours, bottom)
(b) Detailed architecture of our NeLF network

(a) Comparison between NeLF and NeRF structure
(b) Detailed architecture of the *deep* light field network



With *vs.* without residuals

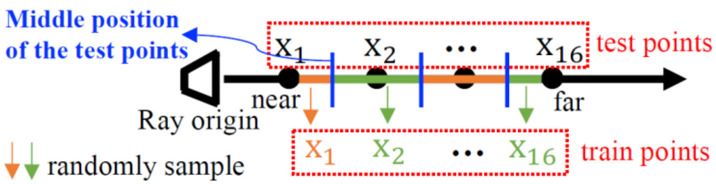
서강대학교 SOGANG UNIVERSITY

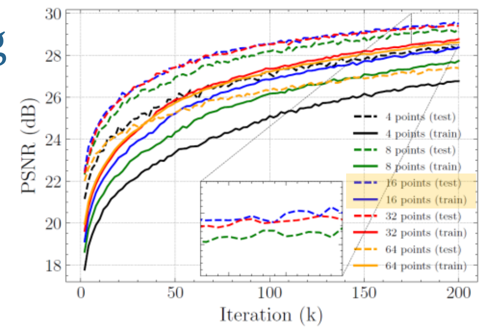
VDS LAB

Papers – Knowledge Distillation

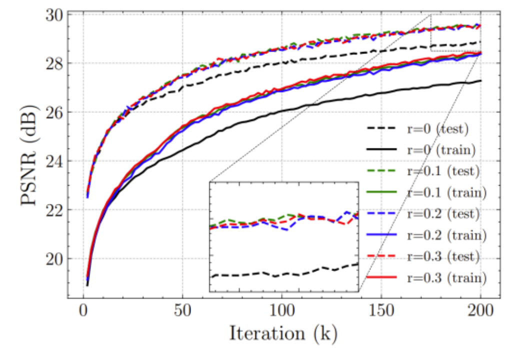
- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]

  - Ray representation and Point Sampling





Comparison of number of sampled points



Comparison of hard example ratio

    - Test points
      - Uniformly sampled K points along a ray.
    - Train points
      - Randomly sampled K points while each point are randomly sampled in a segment.

  - Training hard examples
    - In training, Model maintain a *hard example pool* which have *larger* loss.
      - Add the top *r* (pre-defined percentage constant) points to *hard example pool.*
    - The same amount *r* of hard examples are randomly picked out of the pool to augment the training batch in each iteration.

Papers – Knowledge Distillation

- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]
  - Knowledge distillation for synthesize pseudo-data
    - Employ a pre-trained NeRF model to synthesize extra pseudo-data for training.
      - NeLF needs sufficient training data to optimize deep MLPs
        - ✓"*Lego*" scene of the NeRF-*synthetic* dataset has only 100 training images.
      - Where to sample to synthesize the pseudo data?
        - ✓Randomly sample the normalized ray origin, ray direction of the original training data.
        - ✓The bounding box can be inferred from the training data (min, max)

$$x_o \sim U(x_o^{\min}, x_o^{\max}),\ y_o \sim U(y_o^{\min}, y_o^{\max}),\ z_o \sim U(z_o^{\min}, z_o^{\max})$$
$$x_d \sim U(x_d^{\min}, x_d^{\max}),\ y_d \sim U(y_d^{\min}, y_d^{\max}),\ z_d \sim U(z_d^{\min}, z_d^{\max})$$

Papers – Knowledge Distillation

- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]
  - Knowledge distillation for synthesize pseudo-data
    - Employ a pre-trained NeRF model to synthesize extra pseudo-data for training.

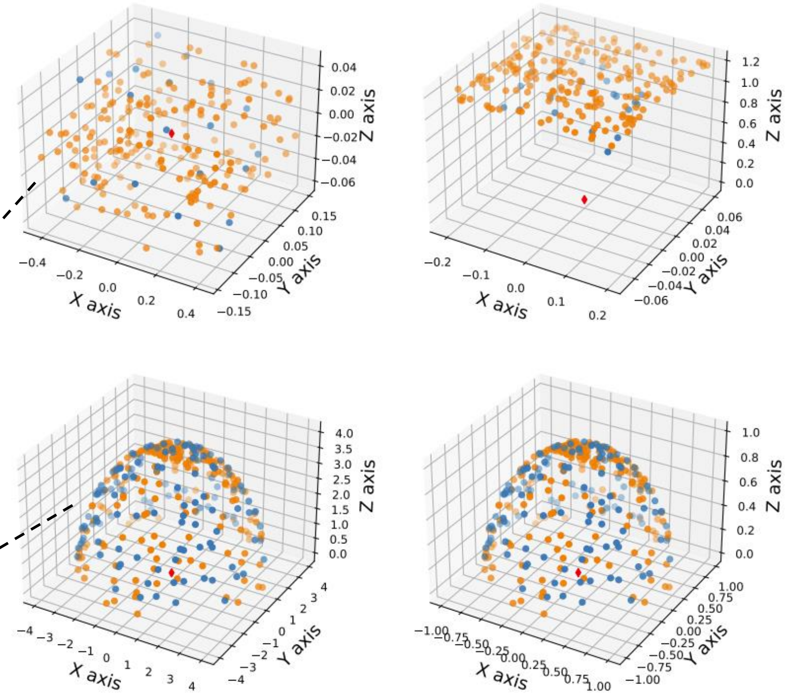
Network	Data	Train PSNR (dB)	Test PSNR (dB)
NeRF [34]	Original (0.1k imgs)	25.61	19.81
NeRF+dropout [47]	Original (0.1k imgs)	25.56	19.83
NeRF+BN [19]	Original (0.1k imgs)	25.43	19.76
NeRF [34]	Pseudo (10k imgs)	23.82	26.67
R2L (W181D88)	Pseudo (10k imgs)	28.38	29.50
R2L (W181D88)	Pseudo + Original (10.1k imgs)	29.85	30.09

Ablation study of different network and data scheme

Training PSNR is lower than the test because of using the hard examples.

Original training data has 20 views

Original training data has 100 views



(a) Origins (b) Directions

Visualization of the origins and directions in 3D space (Row1: real-world scene *Fern*, Row2: synthetic scene *Lego*)

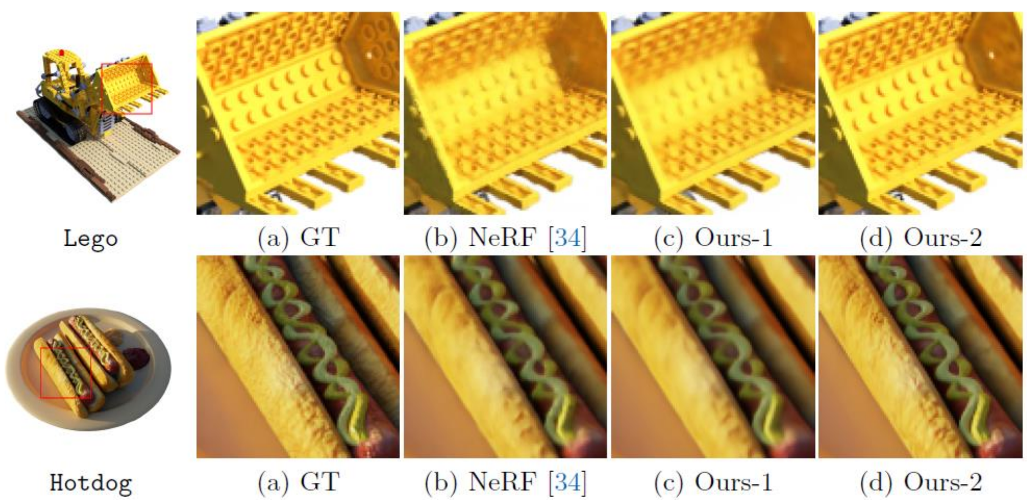
서강대학교 SOGANG UNIVERSITY

VDS LAB

Papers – Knowledge Distillation

- R2L: Distilling NeRF to NeLF for Efficient Novel View Synthesis [ECCV2022]
  - Knowledge distillation for synthesize pseudo-data
    - Employ a pre-trained NeRF model to synthesize extra pseudo-data for training.

Method	Storage (MB)	FLOPs (M)	Synthetic			Real-world		
			PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Teacher NeRF [34]	2.4	303.82	30.47	0.9925	0.0391	27.68	0.9725	0.0733
Ours-1 (Pseudo)	23.7	11.79	30.48 (+0.01)	0.9939	0.0467	27.58 (-0.10)	0.9722	0.0997
Ours-2 (Pseudo+real)	23.7	11.79	31.87 (+1.40)	0.9950	0.0340	27.79 (+0.11)	0.9729	0.0968
Teacher NeRF in [43]	2.4	303.82	31.01	0.95	0.08	-	-	-
KiloNeRF [43]	38.9	~500†	31.00 (-0.01)	0.95	0.03	-	-	-
Teacher NeRF in [4]	4.6	~300	-	-	-	27.928	0.9160	0.065
RSEN [4]	5.4	67.2	-	-	-	27.941 (+0.013)	0.9161	0.060

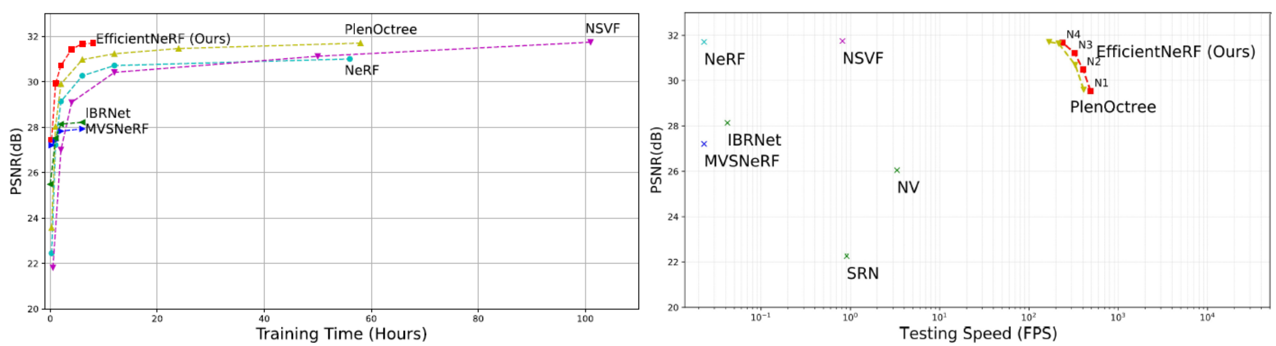


Quantitative and qualitative results

Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]
  - Contributions
    - *First* work to significantly accelerate both training(88%↓) and testing(200fps↑) of NeRF-based methods while maintaining reasonable accuracy.
    - “Valid Sampling” was proposed by constructing dynamic Voxels to accelerate the sampling process at the coarse stage and “Pivotal Sampling” at the fine stage.
    - A simple and efficient data structure, NerfTree, was proposed for NeRF-based methods.

      It quickly caches and queries 3D scenes, thus improving the rendering speed by over 4000 times.
  - Key methods
    - Valid Sampling
    - Pivotal Sampling
    - NerfTree



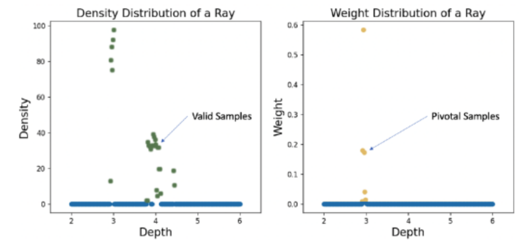
Training and testing efficiency on NeRF-*synthetic* dataset

Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]
  - Why the original NeRF computes even un-necessary points having sparse density?
    - Original NeRF uniformly samples the points along a ray in coarse and fine stage.
    - For common scenes with uniformly sampling, there are only around 10~20% of valid samples and 5~10% are pivotal samples.

Scene	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	**Mean**
Valid Samples (V, %)	9.58 %	7.00 %	3.85 %	9.35 %	15.43 %	19.47 %	8.44 %	11.32 %	**10.56 %**
Pivotal Samples (P, %)	3.79 %	2.25%	1.68 %	3.59 %	5.81 %	7.42 %	3.14 %	4.62 %	**4.04 %**
Scene	Fern	Flower	Fortress	Horns	Leaves	Orchids	Room	Trex	**Mean**
Valid Samples (V, %)	24.28 %	13.68 %	23.45 %	21.34 %	15.09%	19.74 %	30.62 %	18.27 %	**20.81%**
Pivotal Samples (P, %)	15.63 %	7.49 %	4.48 %	10.45 %	8.49%	9.43 %	15.23 %	7.89 %	**9.89%**

Proportions of valid and pivotal samples on the NeRF-*synthetic* dataset and *Forward-Facing* dataset



Density and weight distributions of a typical ray

- Sample proportion
  - N_v: number of samples have density $\sigma_v > 0$ along a ray.
  - N_p: number of samples have weight $w_p > \epsilon$ along a ray. $\rightarrow 1 \times 10^{-4}$
  - N_c: Number of samples in coarse stage
  - Valid sample proportion $V = \frac{N_v}{N_c}$
  - Valid sample proportion $P = \frac{N_p}{N_c}$

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} w_i c_i,$$

$$w_i = T_i \alpha_i, \quad \rightarrow \text{contribution to render color}$$

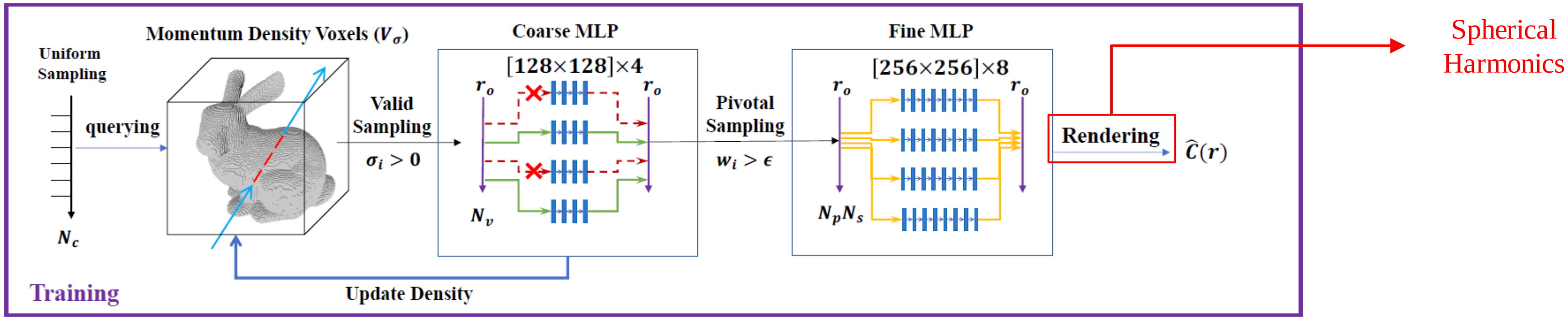
$$T_i = exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j),$$

$$\alpha_i = 1 - exp(-\sigma_i \delta_i),$$

Volume rendering equation

Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]



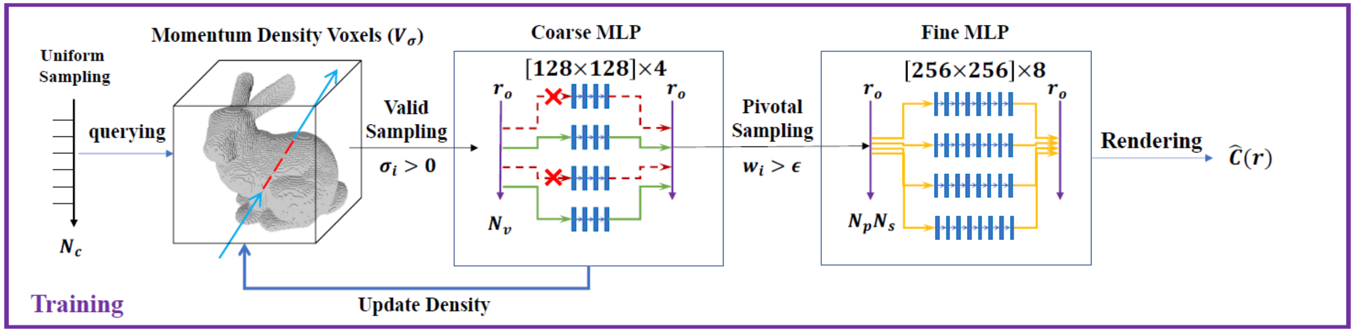
  - Valid Sampling at the Coarse Stage
    - Density voxels V_σ
      - Size: $D \times D \times D$
      - Initialization: All voxel density value set to $\varepsilon = 10.0$
      - 3D Voxel index $\mathrm{i} \in \mathbb{R}^3$
        - ✓ $\mathbf{i} = \frac{\mathbf{x} - \mathbf{x}_{min}}{\mathbf{x}_{max} - \mathbf{x}_{min}} D$
      - Update for every training iteration by Momentum
        - ✓ $V_\sigma[\mathbf{i}] \leftarrow (1-\beta)V_\sigma[\mathbf{i}] + \beta \cdot \sigma_c(\mathbf{x}), \quad \beta \in [0,1]$

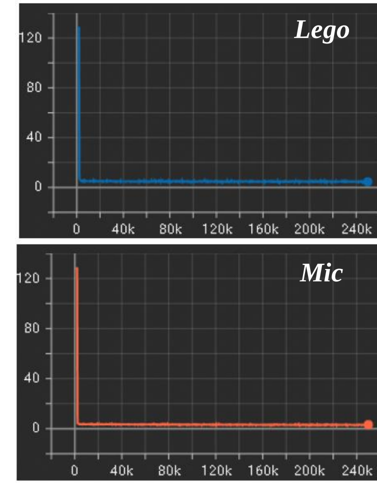
ε	Training Speed	PSNR (↑)
0.1	**0.018s / iter**	31.54
1.0	0.021s / iter	31.68
10.0	0.057 s / iter	**31.75**

Influence of difference value of initial density ε

Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]



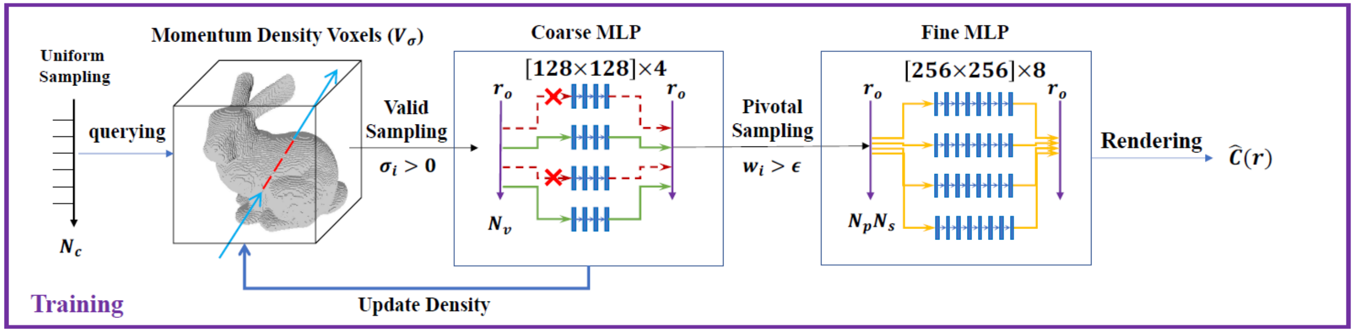


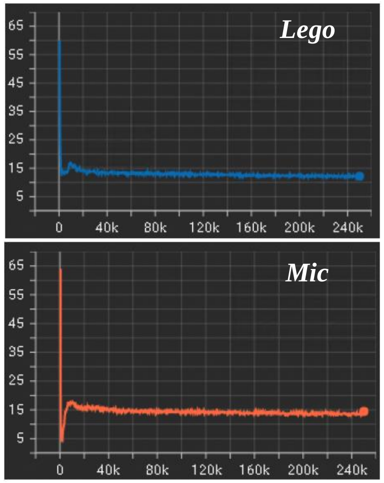
Number of valid samples during training

  - Valid Sampling at the Coarse Stage
    - Sampling N_v valid point $V_\sigma[\mathbf{i}] > 0$
    - First query the latest density from V_σ, and only input $\mathbf{x}_i$ with global densities $V_\sigma[\mathbf{i}] > 0$
    - Time taken
      - $N_v T_m + (N_c - N_v) T_q$
        - ✓T_m: Time for inferring a single point by a coarse MLP.
        - ✓T_q: Time for querying a single point from voxels.
    - Acceleration ratio A_c
      - $A_c = \frac{N_c T_m}{N_v T_m + (N_c - N_v) T_q} \approx \frac{N_c}{N_v} = \frac{1}{V}$

Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]





Number of pivotal samples during training

- Pivotal Sampling at the Fine Stage
  - Sampling N_p follows the coarse weight distribution ($N_p = 2N_c$)
  - pivotal sample $\mathbf{x}_p : w_p > \epsilon$
  - sampling N_s points around $\mathbf{x}_p$
    - $\mathbf{x}_{p,j} = \mathbf{x}_p + j\delta_f r_d$
  - Acceleration ratio A_f
    - $A_f = \frac{N_f}{N_p N_s} \approx \frac{2N_c}{N_p N_s} = \frac{2}{PN_s}$

ϵ	Training Speed	PSNR (↑)
1×10^{-2}	**0.018s / iter**	31.27
1×10^{-4}	0.021 s / iter	31.68
1×10^{-6}	0.029 s / iter	**31.71**

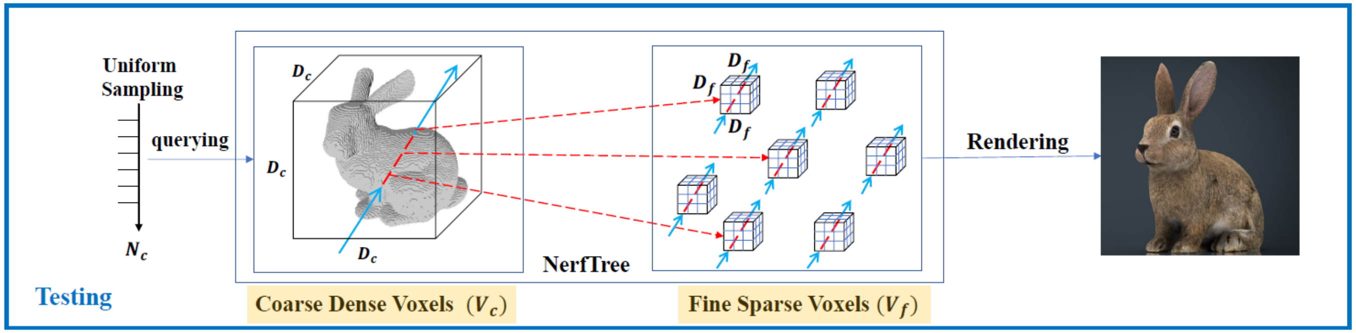
Influence of difference value of pivotal threshold ϵ

	Version	# Sampling Coarse	# Sampling Fine	PSNR (↑)	Rendering Speed (FPS, ↑)
EfficientNeRF	$N1$	64	2	29.54	493.62
	$N2$	64	3	30.49	403.28
	$N3$	96	4	31.22	324.62
	$N4$	128	5	31.68	238.46

Different versions of our EfficientNeRF with Trade-off

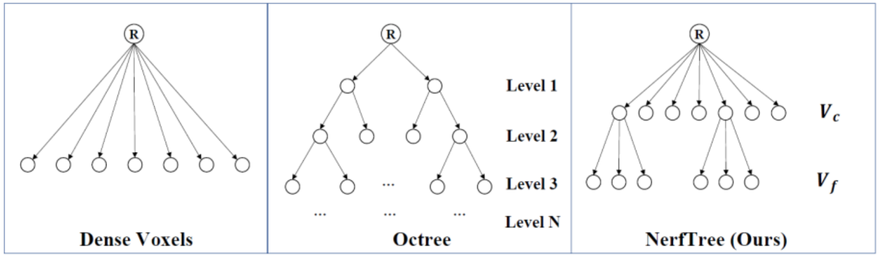
Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]



→ constructed at the training phase

- NerfTree
  - Tree-based data structure (2-depth tree)
  - NerfTree can store the whole scene offline.
    → Eliminating the coarse and fine MLP.
  - Dense voxels only have one depth layer.
    → Achieving the minimal access time and maximum storage.
  - Octree has the opposite characteristic of the Dense voxels.
  - NerfTree combines the advantages of both Voxels and Octrees.
    ∴ Fast speed while not much consuming storage.
  - In experiments, $D_c = 384, D_f = 4$



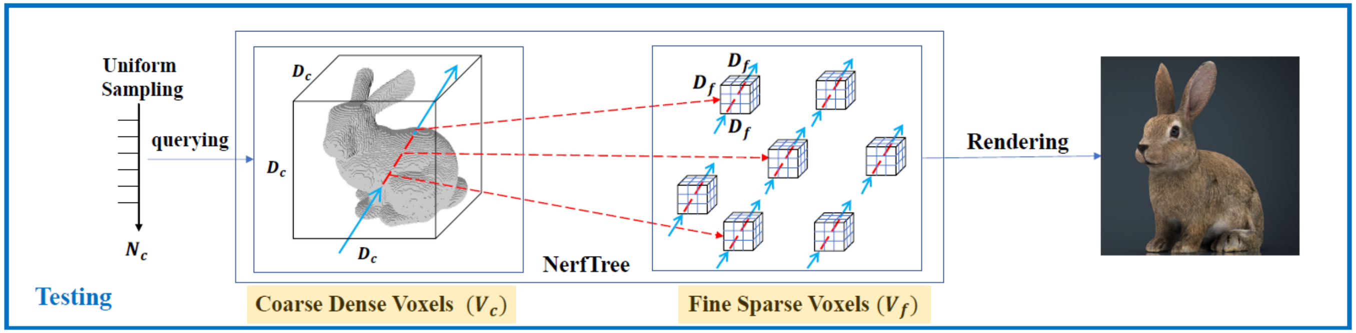
2D graph representation of different 3D data structures

Scene Representation	Memory	Caching Time	Querying Time
Dense Voxels	16 GB	16.55 ms	13.64 ms
Sparse Tensor (Minkowski Engine [3])	2.1 GB	24.72 s	121.21 ms
Octree (PlenOctree [35])	2.6 GB	14.51 s	18.84 ms
NerfTree (Ours)	2.8 GB	22.43 ms	15.39 ms

Comparison of different data structures

Papers – Ray sampling

- EfficientNeRF: Efficient Neural Radiance Fields [CVPR2022]



→ constructed at the training phase

- NerfTree

Inference	Speed (FPS)	PSNR (↑)
Coarse and Fine MLPs	0.18	**31.71**
NerfTree	**238.46**	31.68

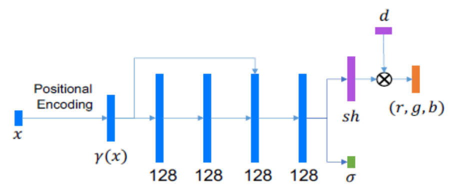
Comparison between different testing model

Coarse MLP		Fine MLP		Time	PSNR(↑)
Lightweight	Standard	Lightweight	Standard	(s / iter, ↓)	
	✓		✓	0.184	**31.01**
✓		✓		0.121	29.28
	✓	✓		0.132	29.39
✓			✓	**0.138**	**30.96**

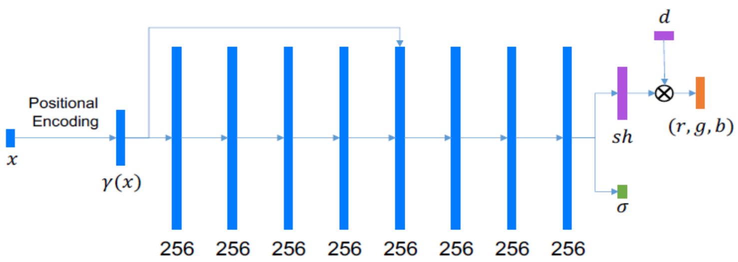
Performance of different combinations

Method	PSNR (↑)	Time (↓)	Improvement (↑)
NeRF [17]	31.01	0.184 s / iter	-
+ SH [35]	31.57	0.183 s / iter	-
+ Lightweight Coarse MLP	31.52	0.137 s / iter	25.54%
+ Coarse Valid Sampling	31.49	0.085 s / iter	53.80%
+ Fine Pivotal Sampling	**31.68**	**0.021 s / iter**	**88.58%**

Contributions of the proposed modules to the training time



(a) Lightweight Coarse MLP.



(b) Standard Fine MLP.

Testing model for comparison using Spherical Harmonics

Conclusion

- Synthesizing pseudo data for sufficient training data by distilling from the Teacher.
- Reduce computational complexity by single forward pass in Light Field.
- Novel ray sampling strategy focusing on the meaningful sample points to reduce time consumption.
- For testing, 2-depth Tree structure improves model efficiency.