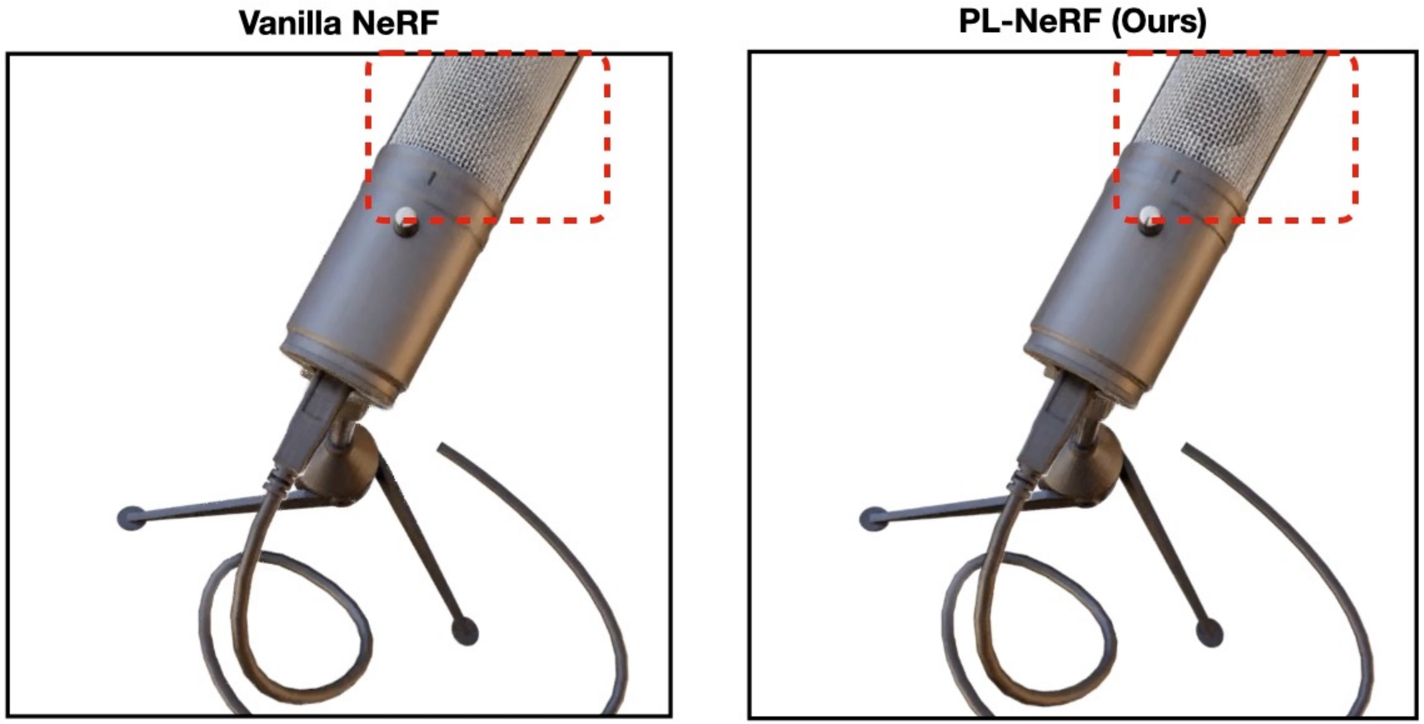
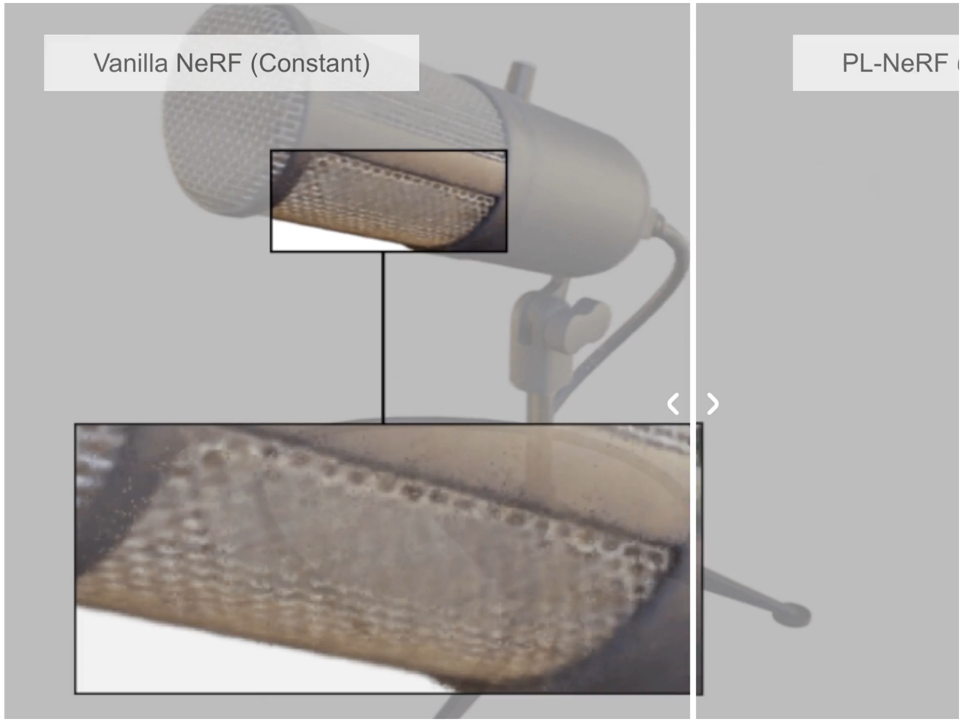
NeRF Revisited: Fixing Quadrature Instability in Volume Rendering

Mikaela Angelina Uy [1], George Nakayama [1], Guandao Yang [1,2],
Rahul Thomas [1], Leonidas Guibas [1] , Ke Li [3, 4]

Stanford University [1]
Cornell University [2]
Simon Fraser University [3]
Google [4]



Observe that the structure inside the mic is lost in piecewise constant opacity approximation.

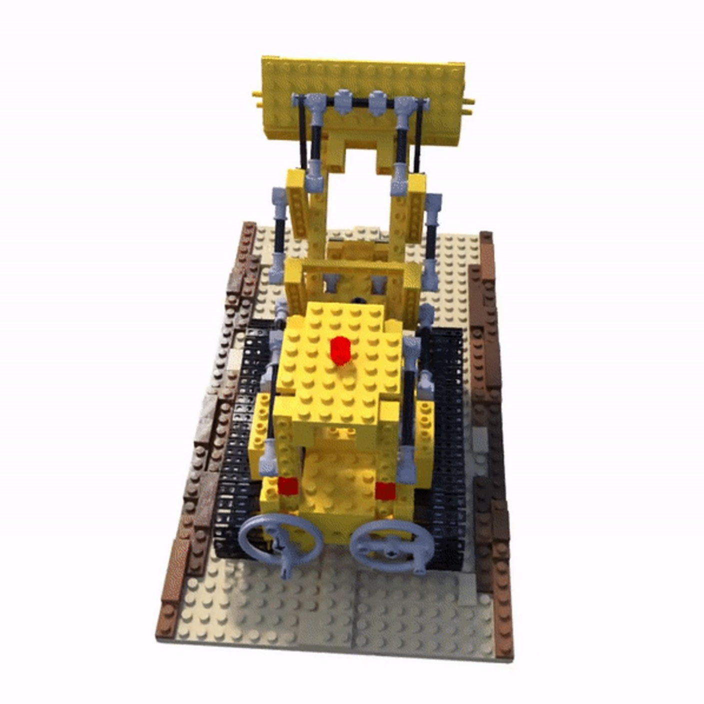


Leonidas Guibas Laboratory
Geometric Computing

NEURAL INFORMATION PROCESSING SYSTEMS

NeRF: Neural Radiance Fields

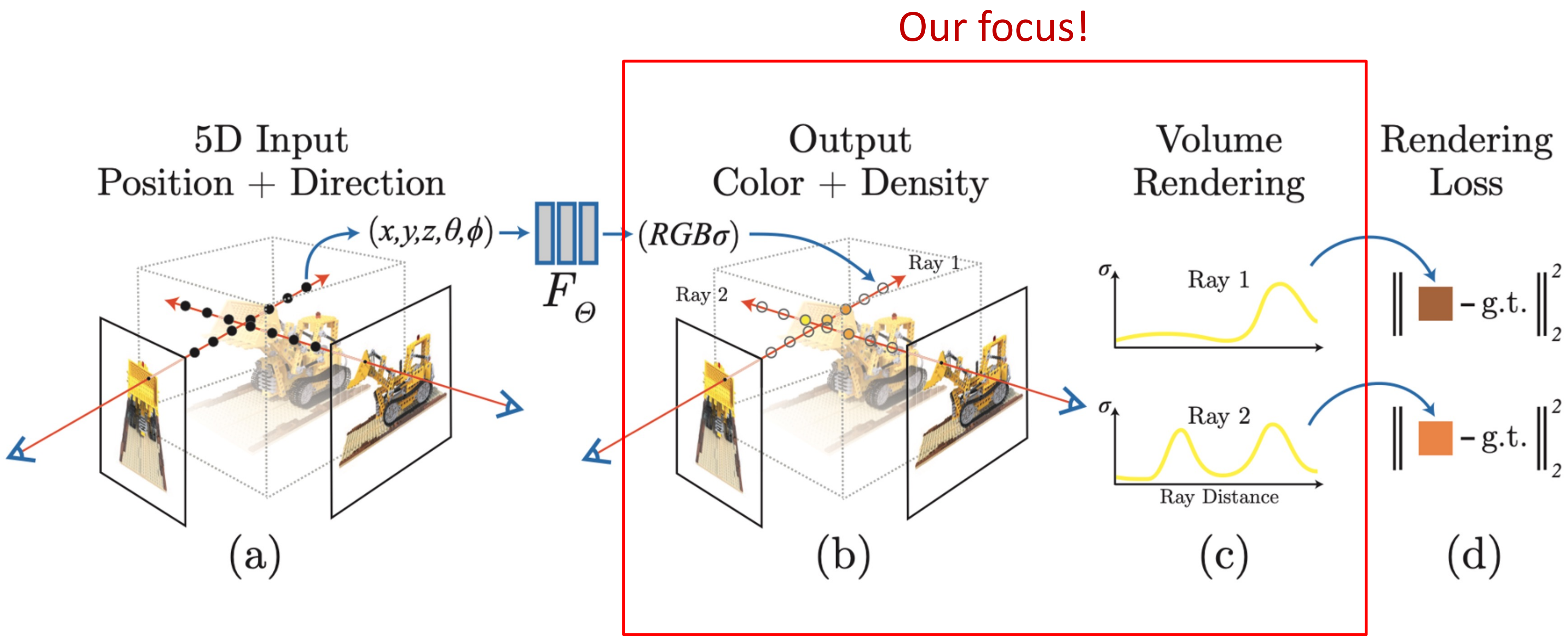
- Novel view synthesis given only input images with known poses



- Input: 5D continuous coordinate
- Output: volume density (1D), color (rgb)

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. B. Mildenhall et. al., ECCV ‘20

NeRF: Neural Radiance Fields



- We dive into the actual volume rendering equation.

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. B. Mildenhall et. al., ECCV '20

What We ~~Are~~ Were Used to

- The formula that we are all used takes the following form:

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

- This comes from the continuous integral:

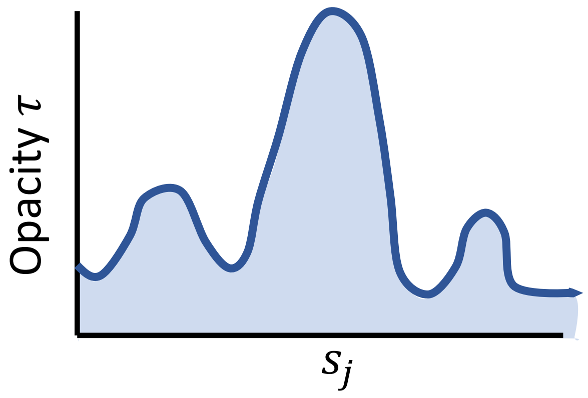
$$\hat{y} = \mathbb{E}_{s \sim p(s)}[c(s)] = \int_0^\infty \underline{p(s)} c(s) \, \mathrm{d}s$$

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

Where did it come from?

- Continuous integral:

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



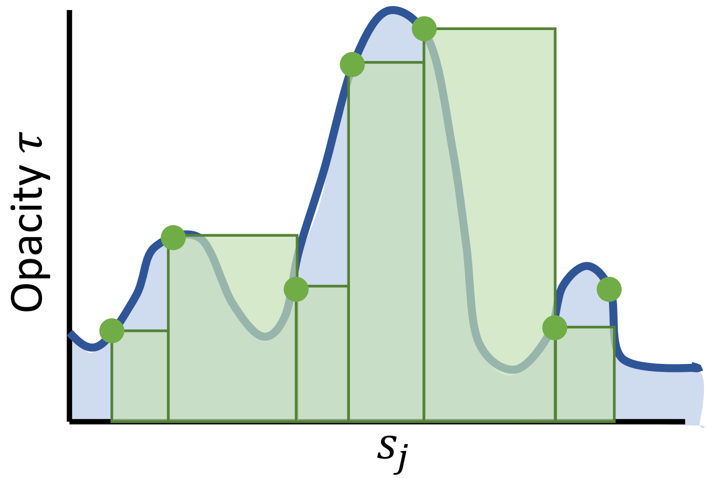
- In practice, it is approximated with **quadrature**, resulting in the expression we are used to.

$s_1, s_2, \ldots, s_N$ be N (ordered) samples

$$\boxed{\forall s \in [s_j, s_{j+1}], \tau(s) = \tau(s_j),}$$

$$\Downarrow$$

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



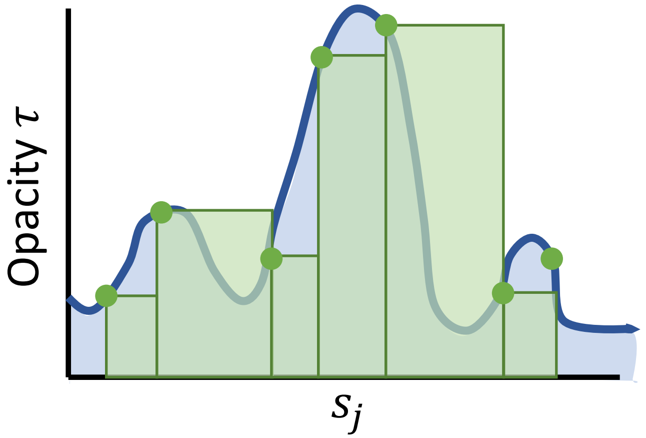
This is the **exact integral** under the **piecewise constant** opacity approximation!

NeRFs with Piecewise Constant Opacity

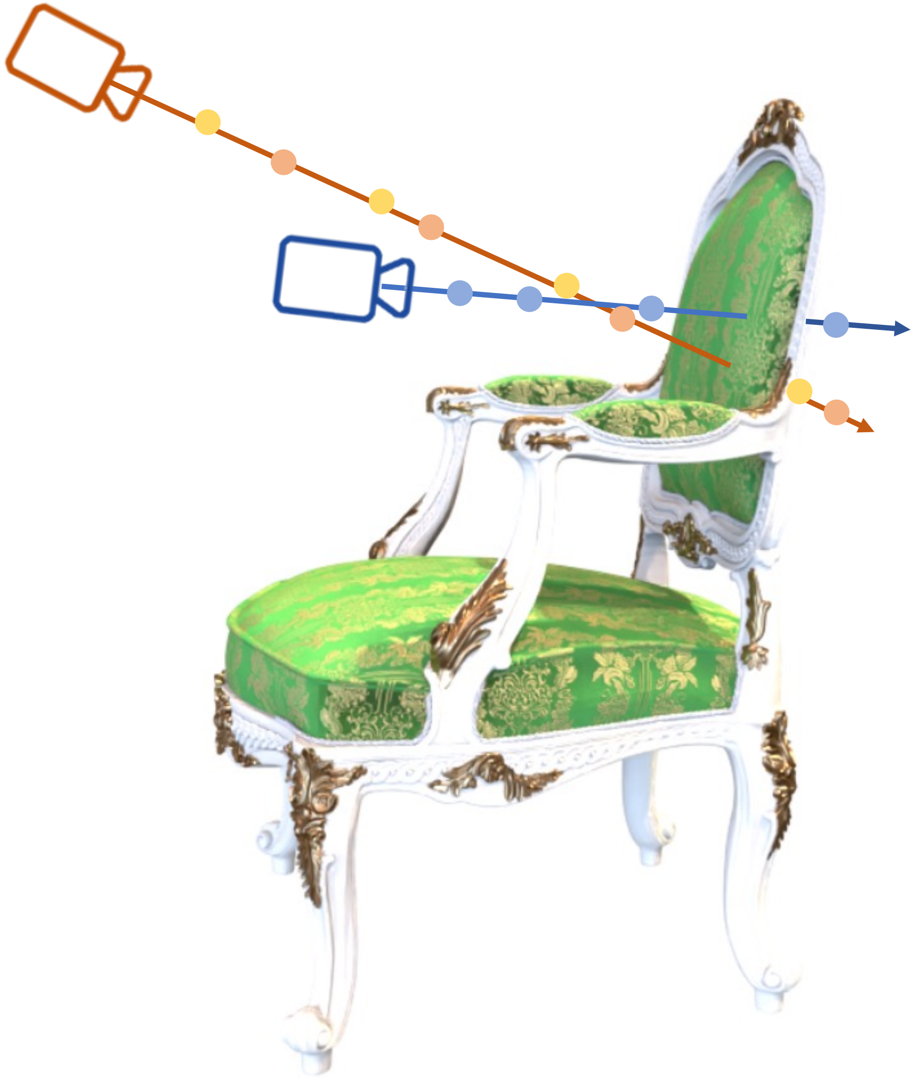
- Different from the classical works, the difference with the emergence of NeRFs is optimization.
  - Opacity and color are learned and trained using the volume rendering equation.
  - Instead of just being evaluated as in classical works.

- Note: for each set of samples along the ray, the opacity at the **left bin** is assigned for each interval.

$$\boxed{\forall s \in [s_j, s_{j+1}], \tau(s) = \tau(s_j),}$$



Quadrature Instability



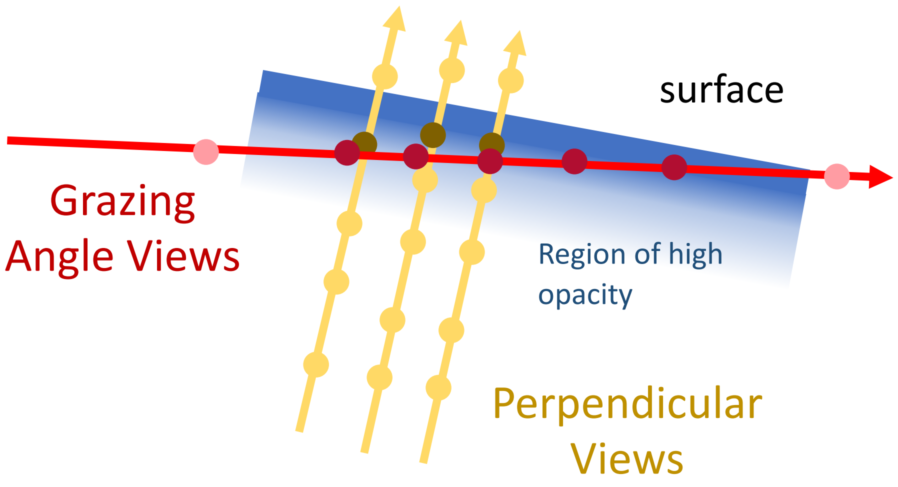
$$\forall s \in [s_j, s_{j+1}], \tau(s) = \tau(s_j),$$

Problems with Quadrature Instability

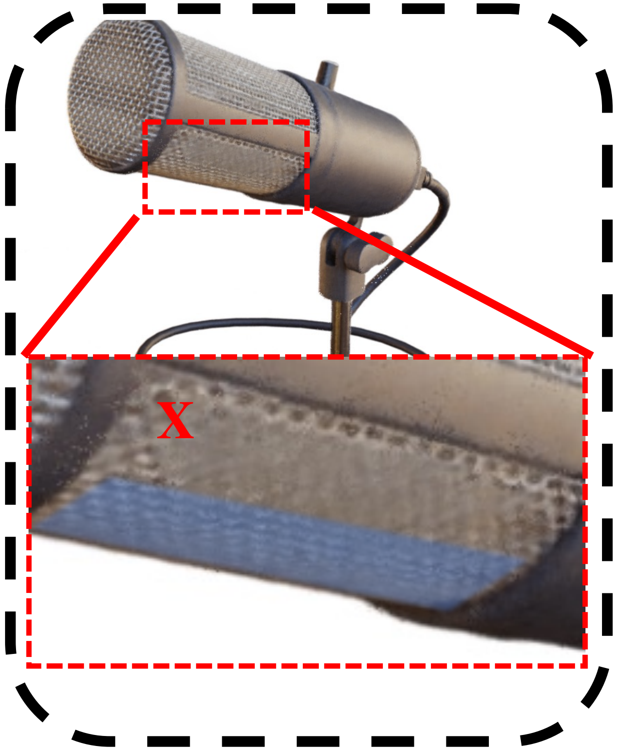
- Ray Conflicts

$$\forall s \in [s_j, s_{j+1}], \tau(s) = \tau(s_j),$$

Piecewise Constant Opacity

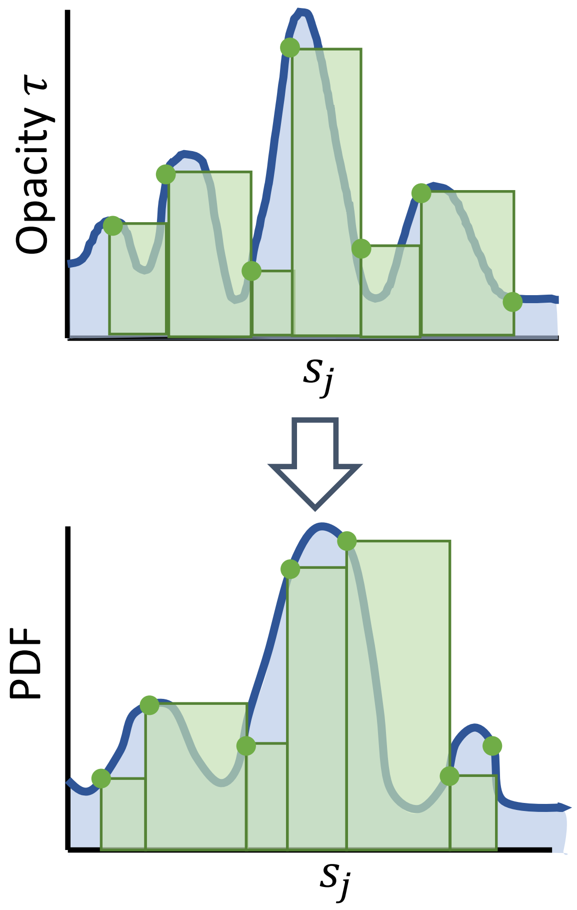


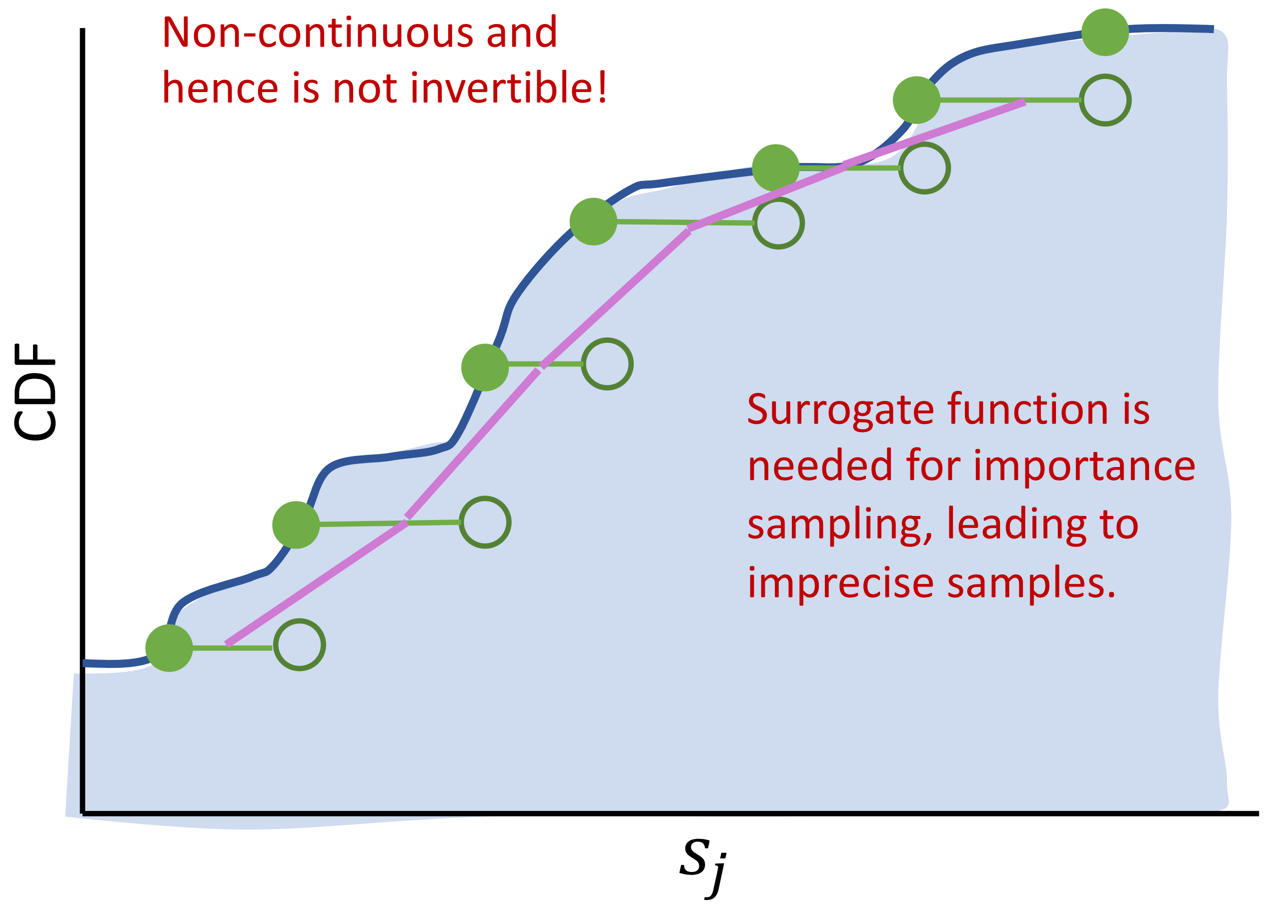
Rendered Test View



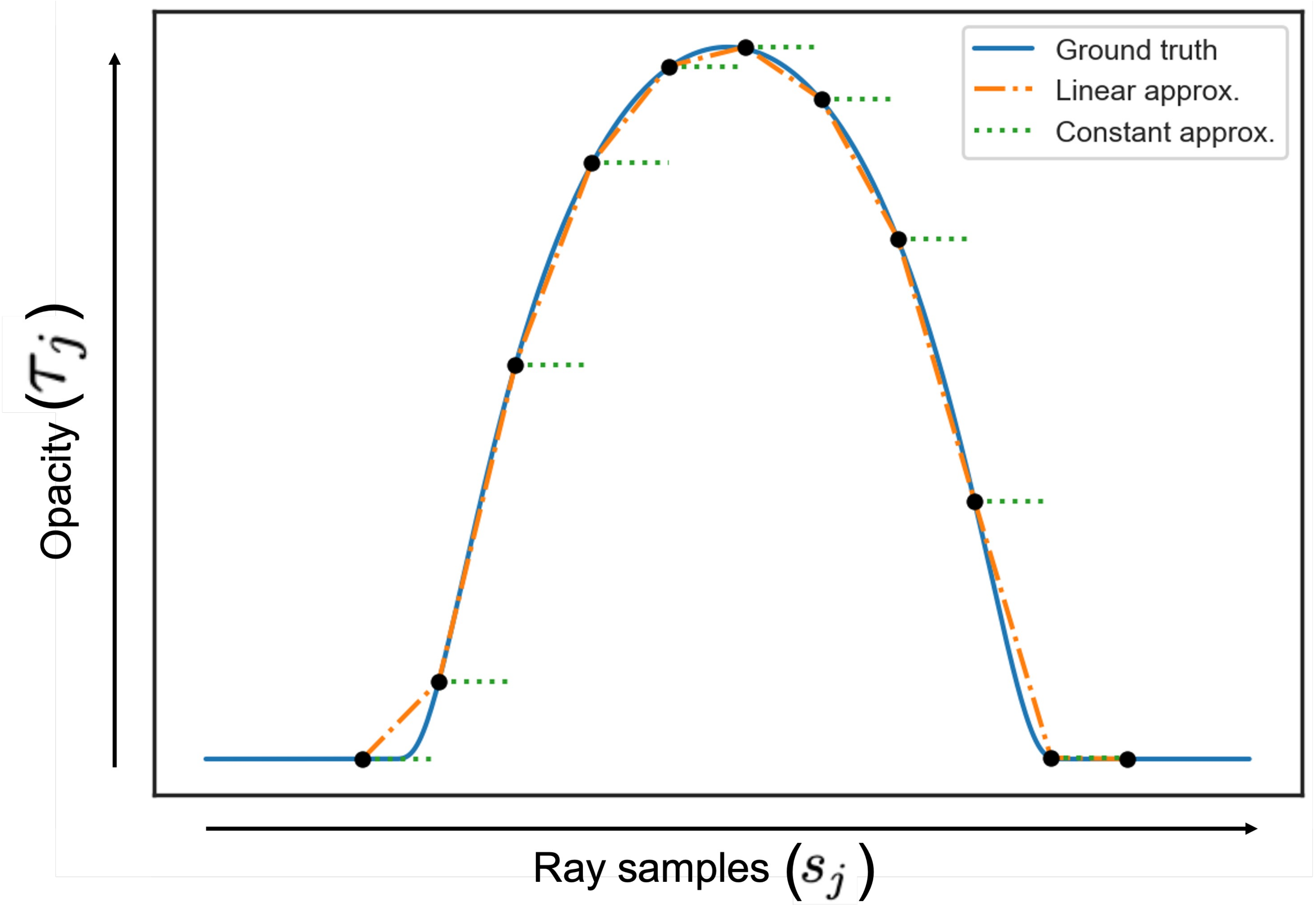
Problems with Quadrature Instability

- Non-invertibility of the CDF





Let’s do better!



General Form for P_j

- We first showed that the probability of an interval can be exactly evaluated iff transmittance (T) is in closed-form:

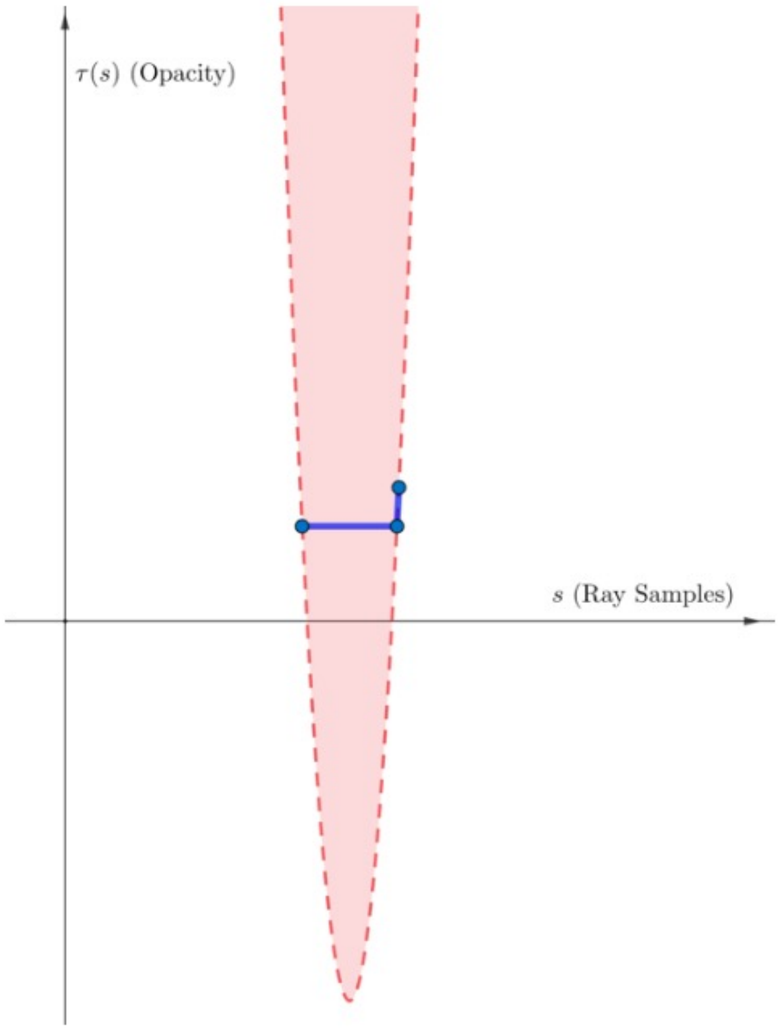
$$P_j = \int_{s_j}^{s_{j+1}} \tau(s) T(s) \, \mathrm{d}s = - \int_{s_j}^{s_{j+1}} T'(s) \, \mathrm{d}s = T(s_j) - T(s_{j+1}).$$

- We know that this holds for any piecewise polynomial approximation for opacity.

Quadratic and Higher Order Polynomials

- Degree = 0 results in the original piecewise constant assumption with quadrature instability.

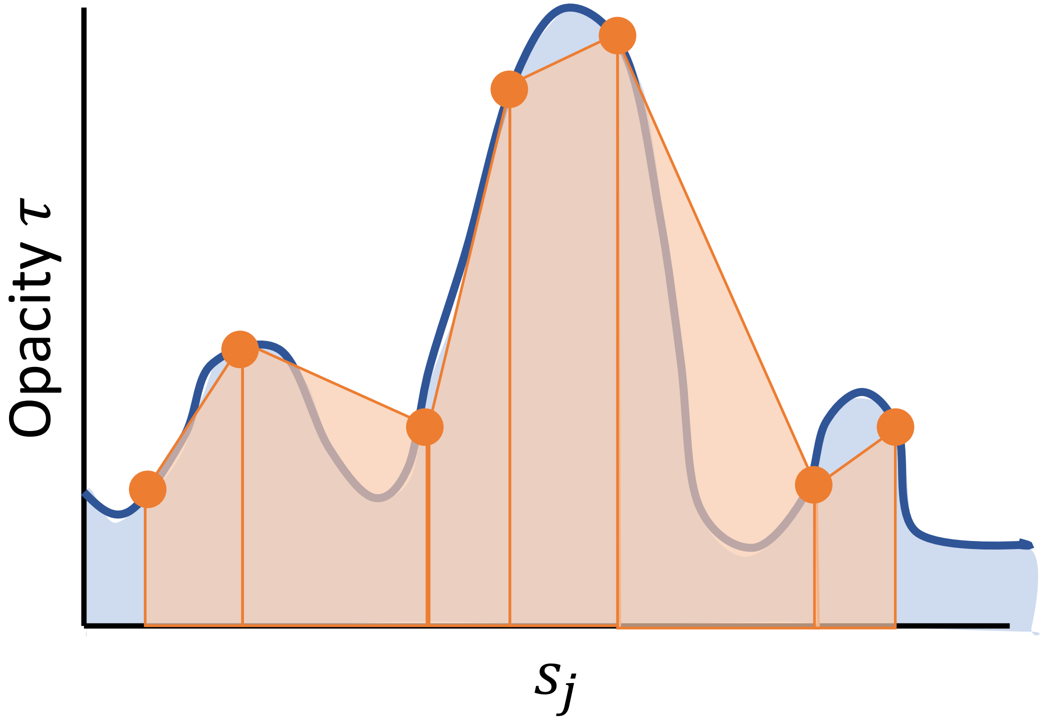
- We further show that for degree $\geq$ 2, it would lead to poor numerical conditioning.
  - Interpolating a higher degree polynomial can give negative opacity values when samples are close.



Our Piecewise Linear Derivation for Opacity

- We use a quadrature under a **piecewise linear** assumption for opacity.
  - For $s \in [s_j, s_{j+1}]$, where $\tau_j = \tau(s_j), \tau(s_{j+1}) = \tau(s_{j+1})$

$$\boxed{\tau(s) = \left(\frac{s_{j+1} - s}{s_{j+1} - s_j}\right) \tau_j + \left(\frac{s - s_j}{s_{j+1} - s_j}\right) \tau_{j+1}.}$$



Our Piecewise Linear Derivation for Opacity

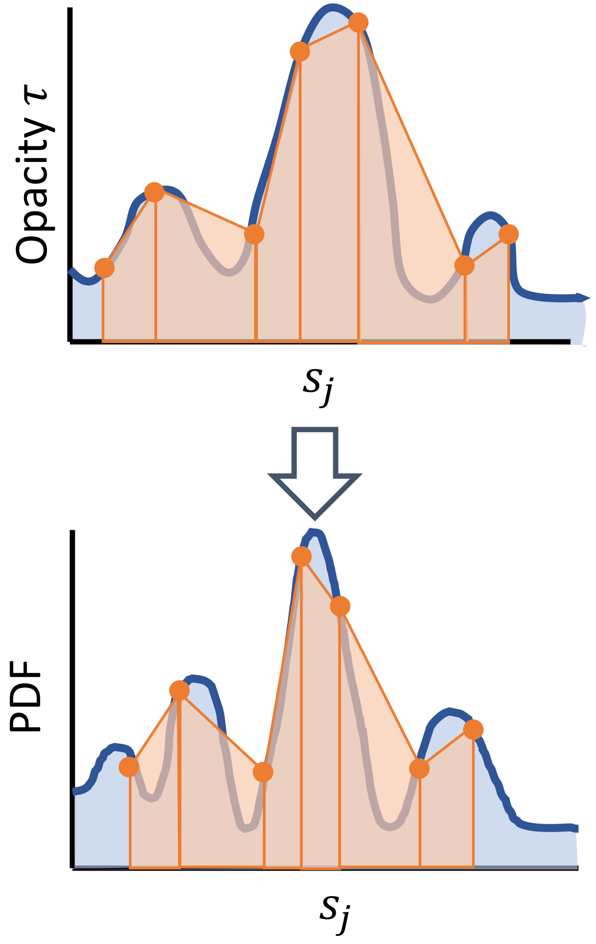
- Under this assumption, we get the following simple and closed-form expressions:

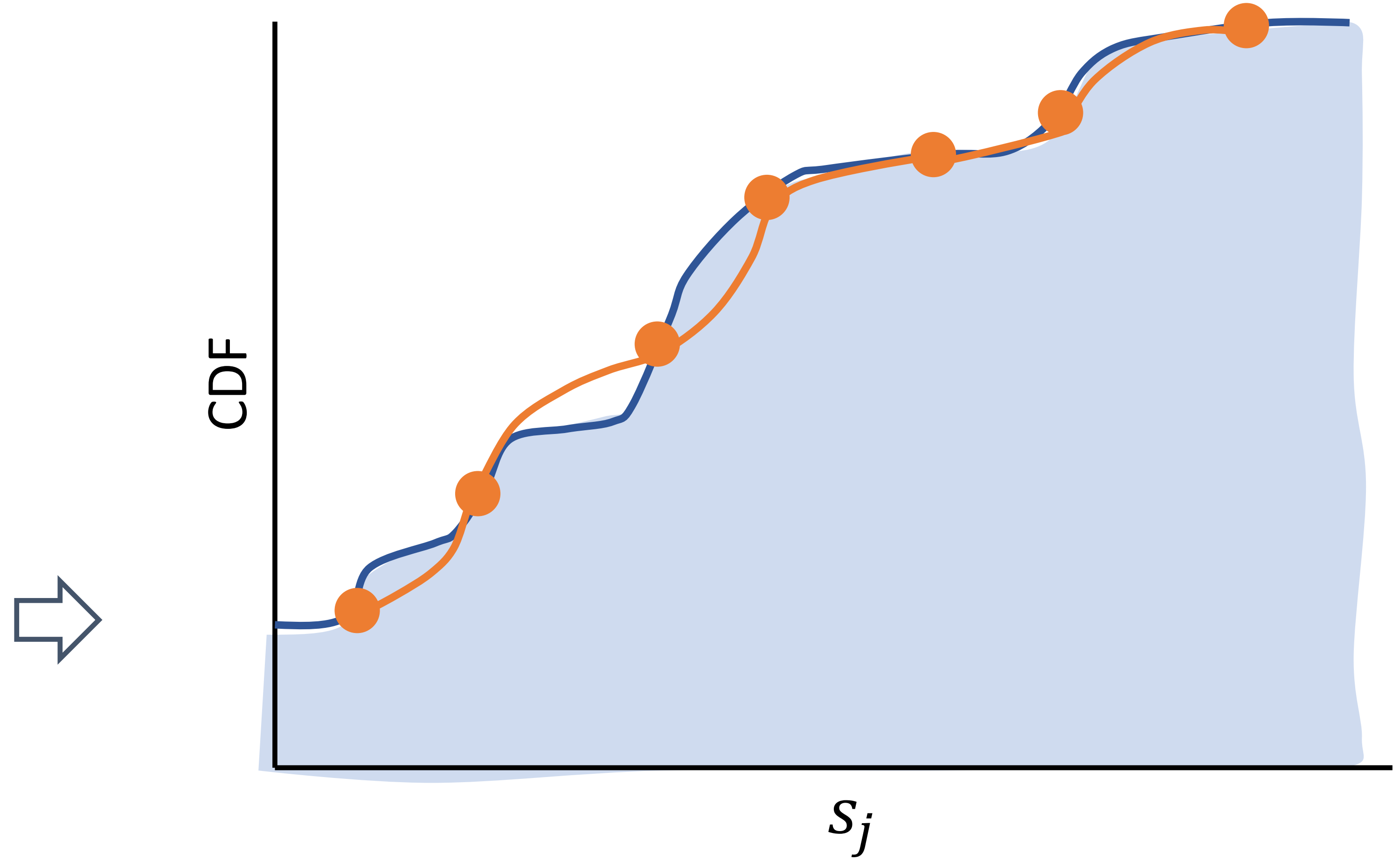
$$\boxed{P_j = T(s_j) \cdot \left(1 - \exp\left[-\frac{(\tau_{j+1} + \tau_j)(s_{j+1} - s_j)}{2}\right]\right).}$$

$$\boxed{T(s_j) = \prod_{k=1}^{i} \exp\left[-\frac{(\tau_k + \tau_{k-1})(s_k - s_{k-1})}{2}\right].}$$

Our Precise Importance Sampling

- The CDF is increasing and continuous, thus it is invertible!

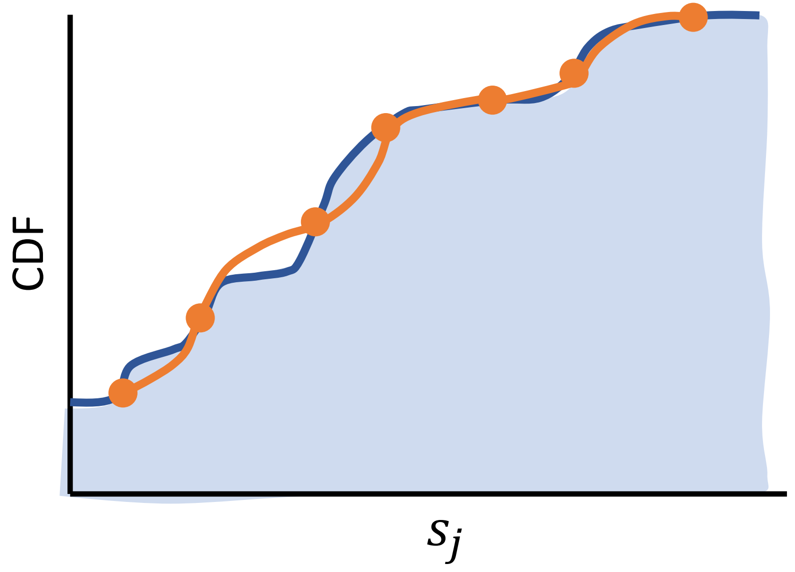




Invertibility

- The CDF is increasing and continuous, thus it is invertible!

$$F(t) = \int_0^t p(s)\,\mathrm{d}s = \sum_{s_j<t} P_j + \int_{s_j}^t p(s)\,\mathrm{d}s = \sum_{s_j<t} P_j + \int_{s_j}^t \tau(s)T(s)\,\mathrm{d}s$$

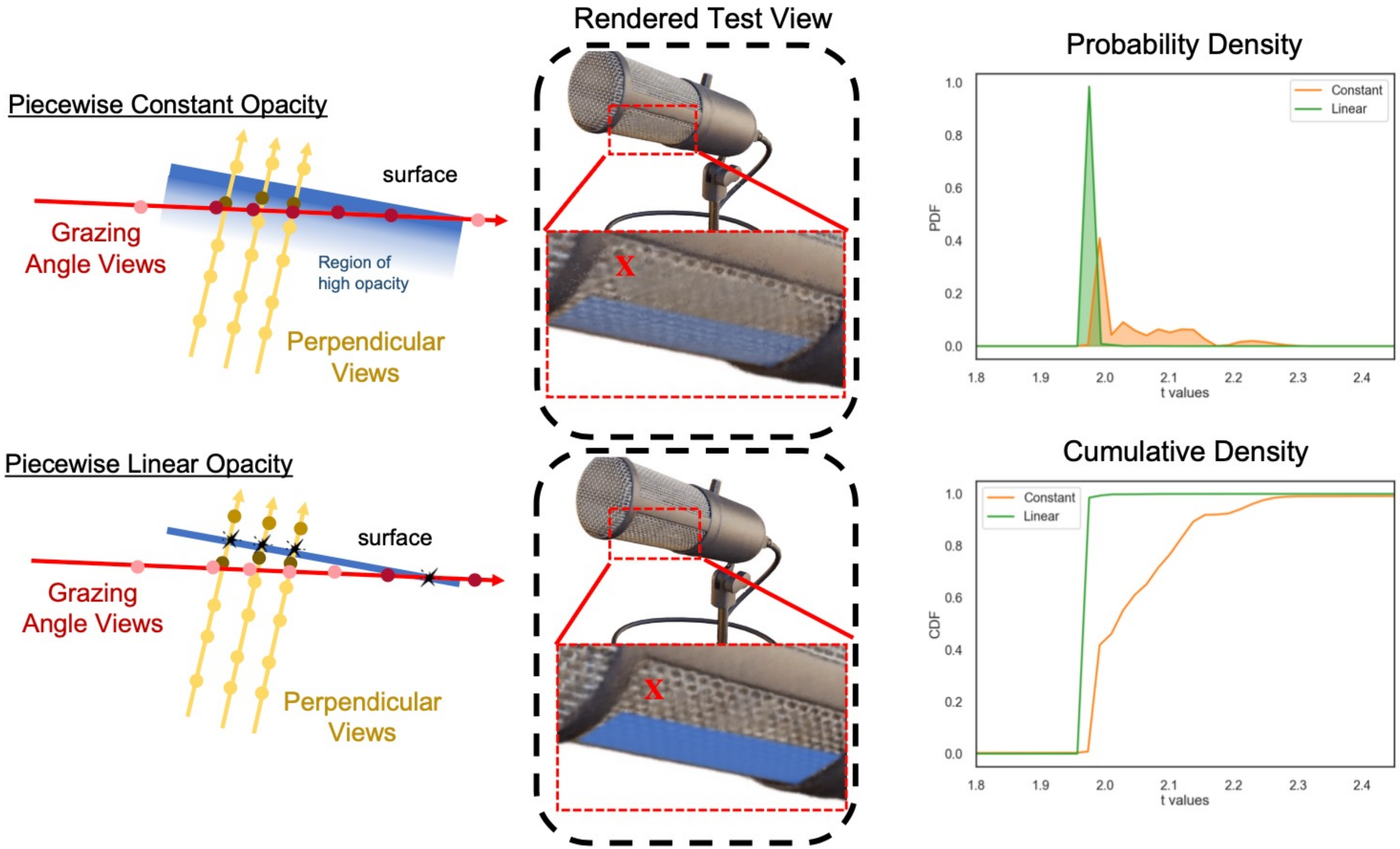


- We are also able to derive a closed-form solution for inverse transform sampling -- our **precise importance sampling**.

$$\boxed{t = \frac{s_{k+1}-s_k}{\tau_{k+1}-\tau_k}\left[-\tau_k + \sqrt{\tau_k^2 + \frac{2(\tau_{k+1}-\tau_k)\left(-\ln\frac{1-u}{T(s_k)}\right)}{(s_{k+1}-s_k)}}\right].}$$

Which also leads to more effective supervision on samples, e.g. depth.

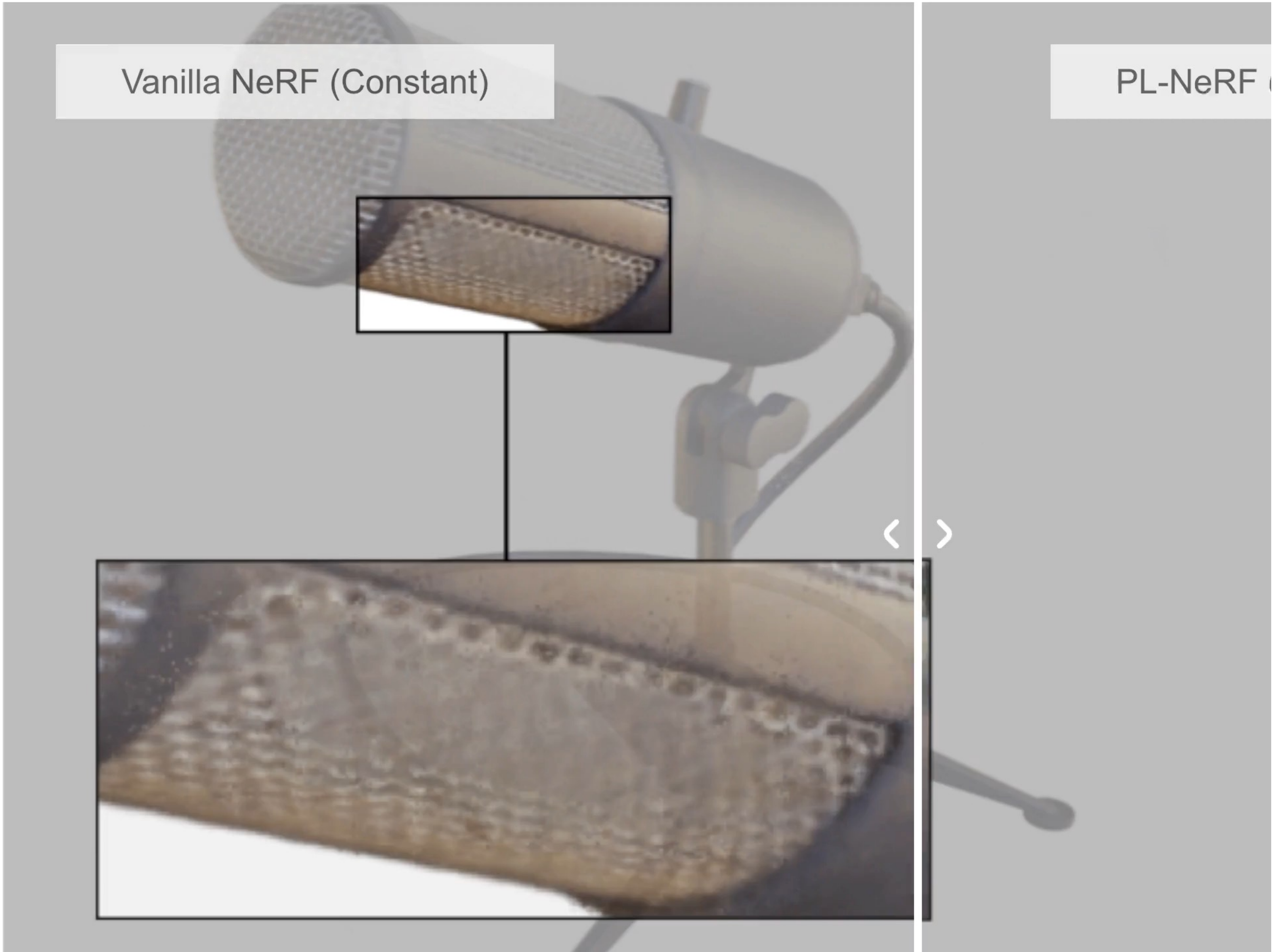
Our Precise Importance Sampling



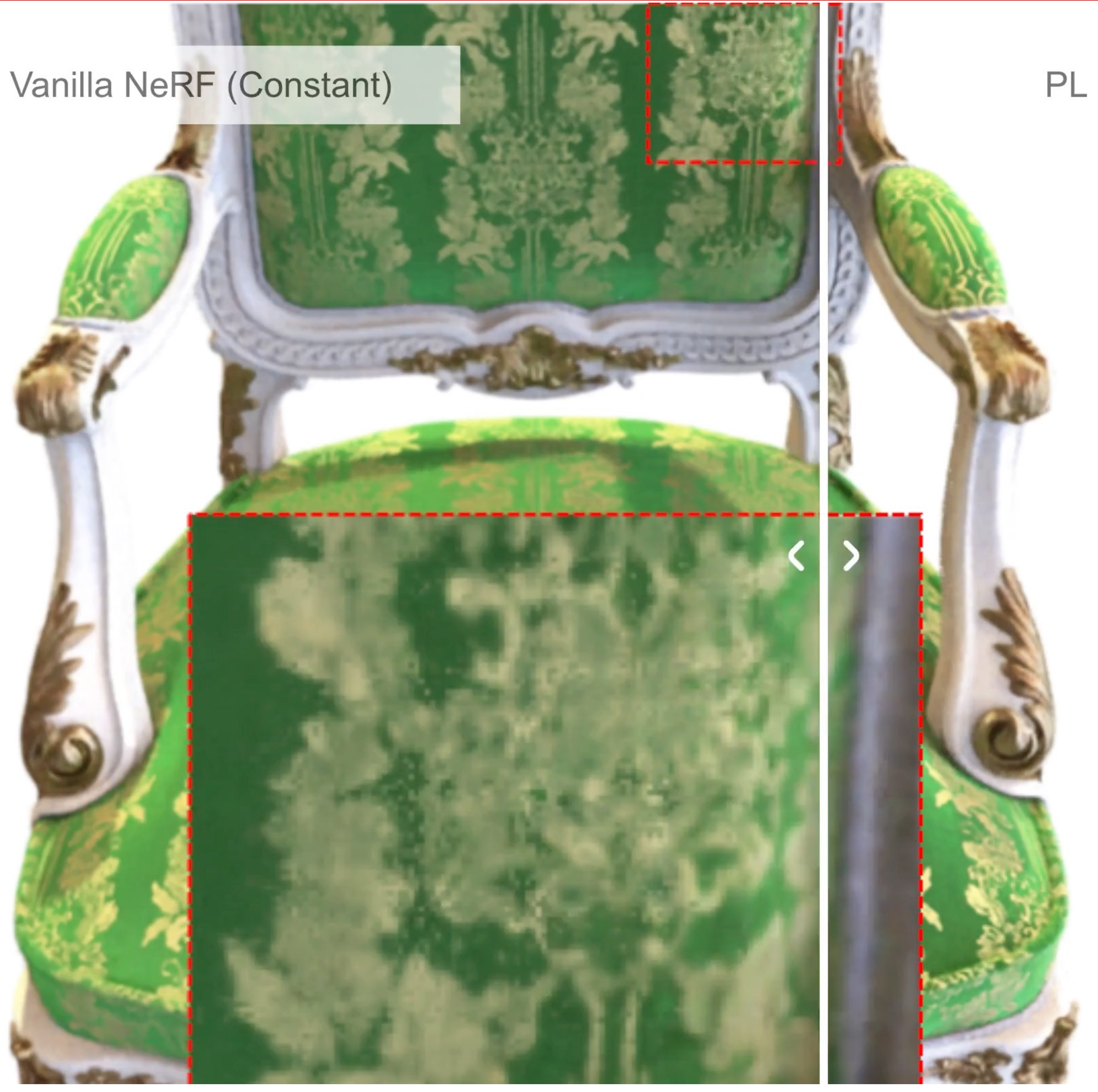
Results: Alleviating Ray Conflicts

Ray Conflicts

Results: Less Fuzzy Surfaces



Results: Crisper Textures



Results: Quantitative

Blender		Avg.	Chair	Drums	Ficus	Hotdog	Lego	Mat.	Mic	Ship
PSNR↑	Const. (Vanilla)	30.61	32.54	24.79	29.63	36.08	32.01	29.31	32.55	27.95
	Linear (Ours)	**31.10**	**32.92**	**25.07**	**30.18**	**36.46**	**32.90**	**29.52**	**33.08**	**28.71**
SSIM↑	Const. (Vanilla)	0.943	0.966	0.918	0.960	0.975	0.959	0.943	0.978	0.846
	Linear (Ours)	**0.948**	**0.969**	**0.923**	**0.965**	**0.977**	**0.966**	**0.948**	**0.981**	**0.857**
LPIPS↓	Const. (Vanilla)	5.17	3.19	7.97	4.14	2.48	2.33	4.32	2.16	14.8
	Linear (Ours)	**4.39**	**2.85**	**7.10**	**3.03**	**2.28**	**1.81**	**3.21**	**1.73**	**13.1**
RFF		Avg.	Fern	Flower	Fortress	Horns	Leaves	Orchid	Room	Trex
PSNR↑	Const. (Vanilla)	27.53	26.79	28.23	32.53	28.54	22.35	21.20	33.03	27.58
	Linear (Ours)	**28.05**	**26.85**	**28.71**	**32.95**	**29.38**	**22.51**	**21.25**	**33.99**	**28.79**
SSIM↑	Const. (Vanilla)	0.874	0.746	0.886	0.925	0.893	0.816	0.746	0.956	0.916
	Linear (Ours)	**0.885**	**0.863**	**0.902**	**0.932**	**0.911**	**0.826**	**0.754**	**0.961**	**0.933**
LPIPS↓	Const. (Vanilla)	7.37	9.67	6.34	2.92	7.26	11.0	11.8	4.33	5.66
	Linear (Ours)	**6.06**	**7.92**	**4.93**	**2.46**	**5.51**	**9.59**	**10.2**	**3.54**	**4.38**

Table 1: **Quantitative Results on Blender and Real Forward Facing Datasets.** Reported LPIPS scores are multiplied by 10^2

Results: Qualitative

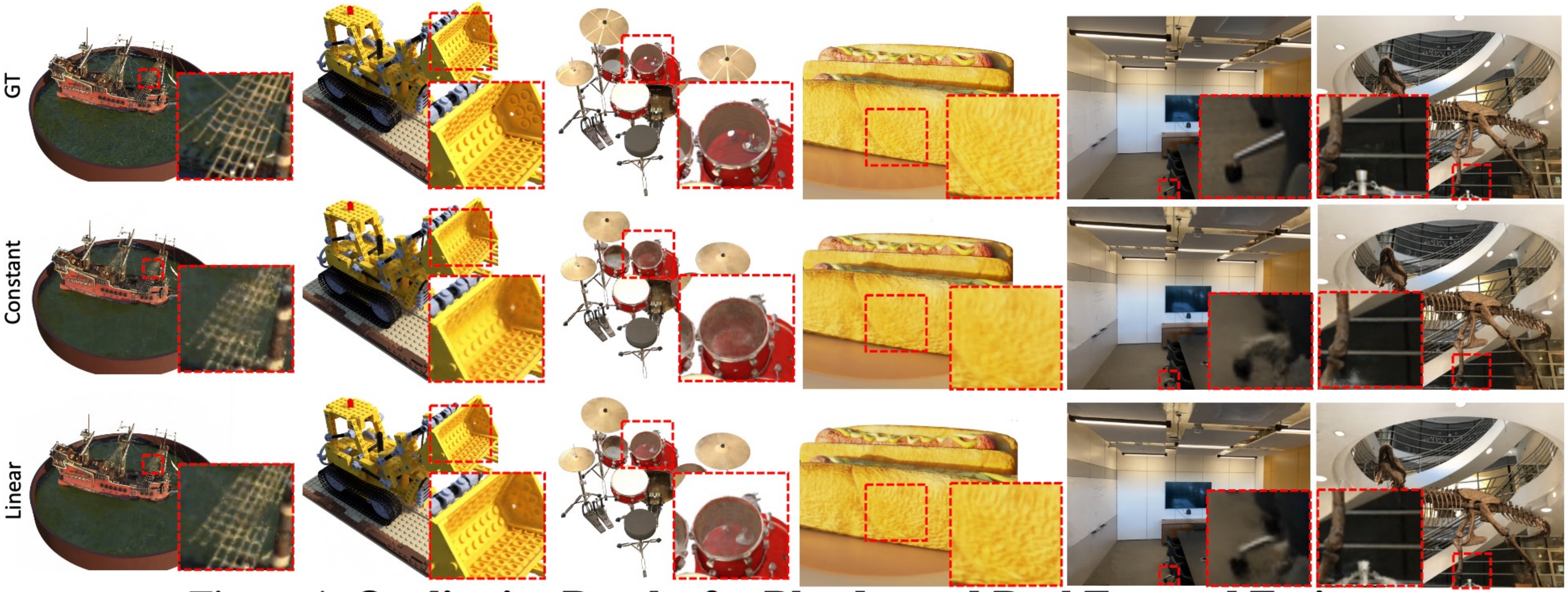
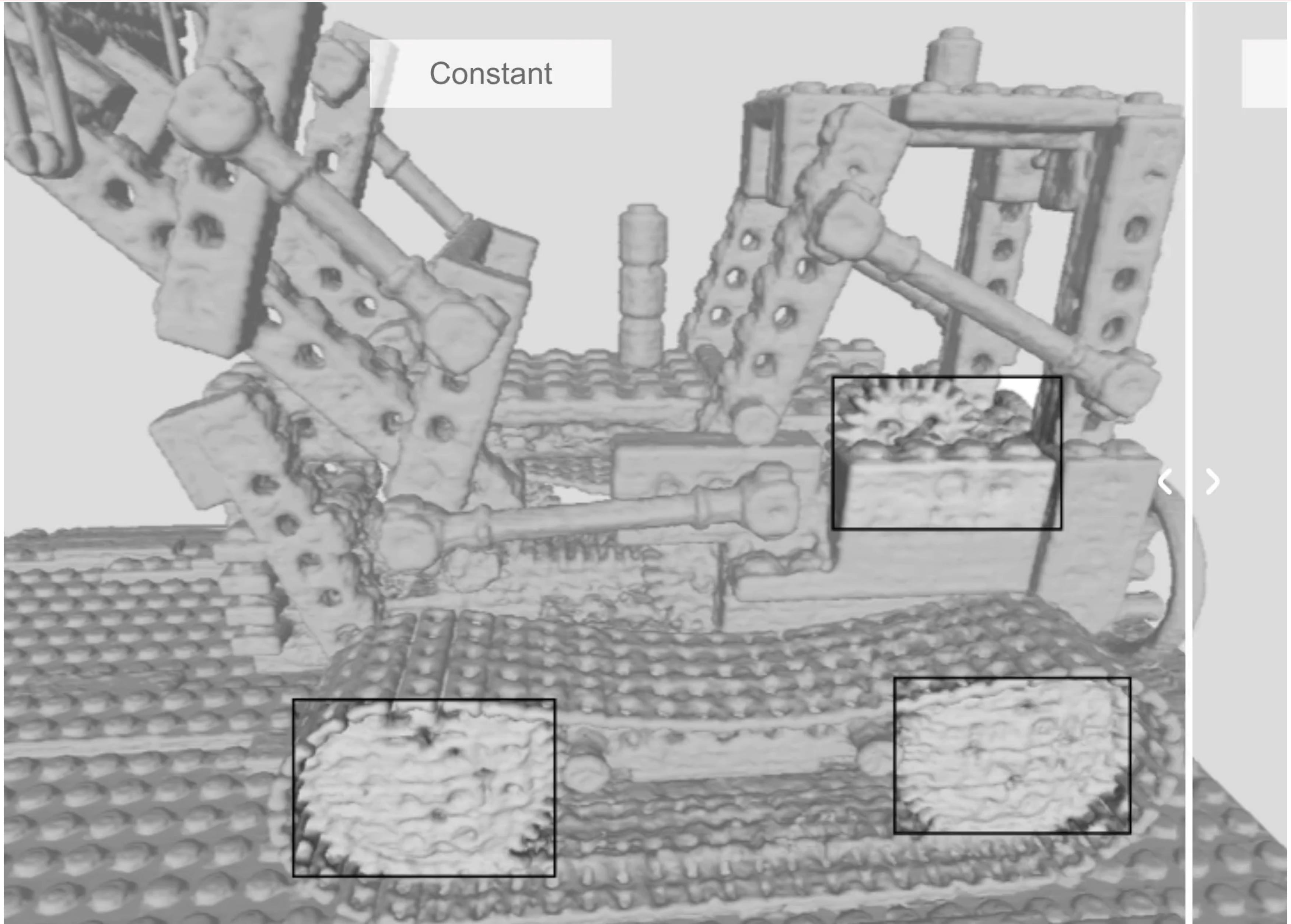
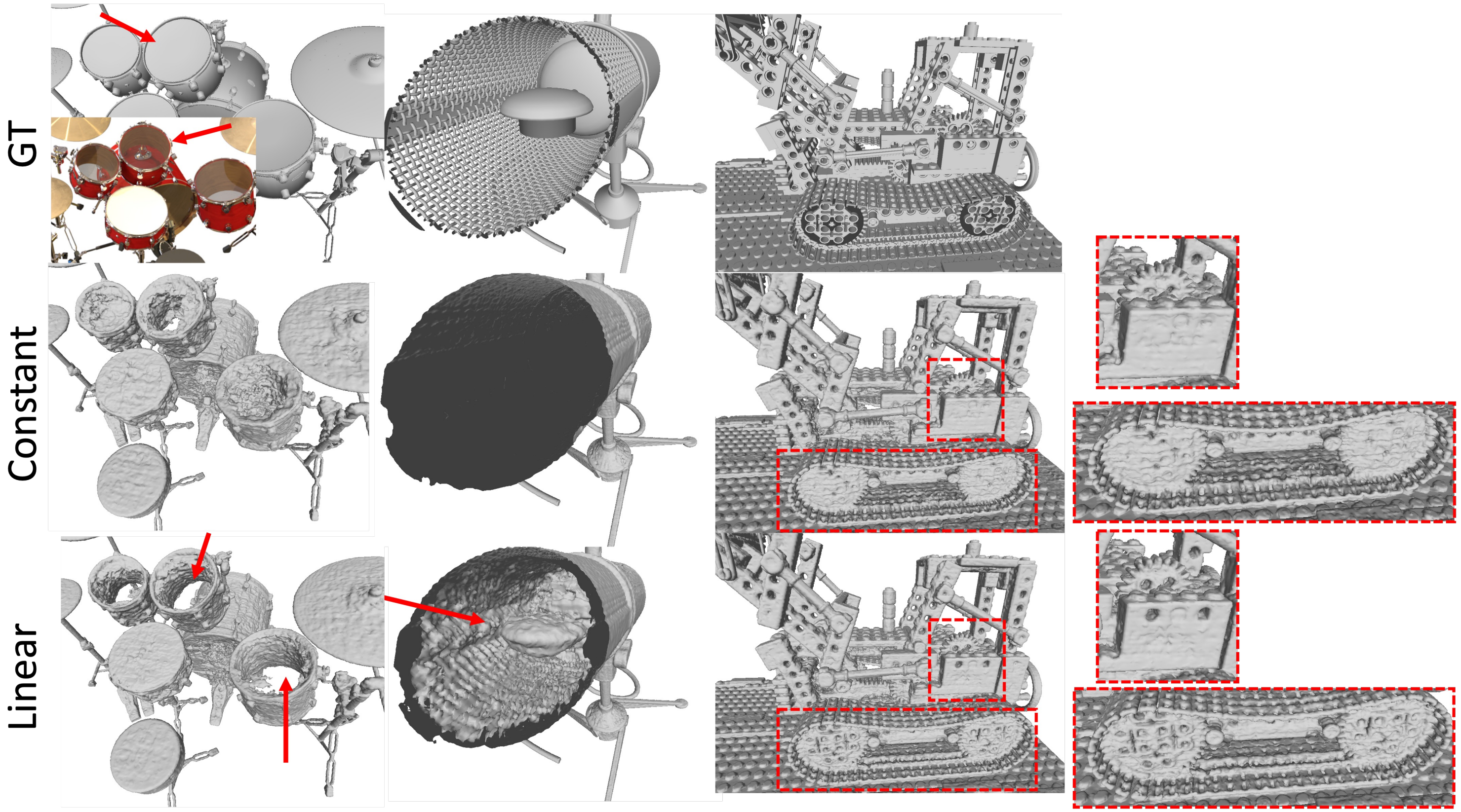


Figure 4: **Qualitative Results for Blender and Real Forward Facing.**

Results: Better Geometry Extraction



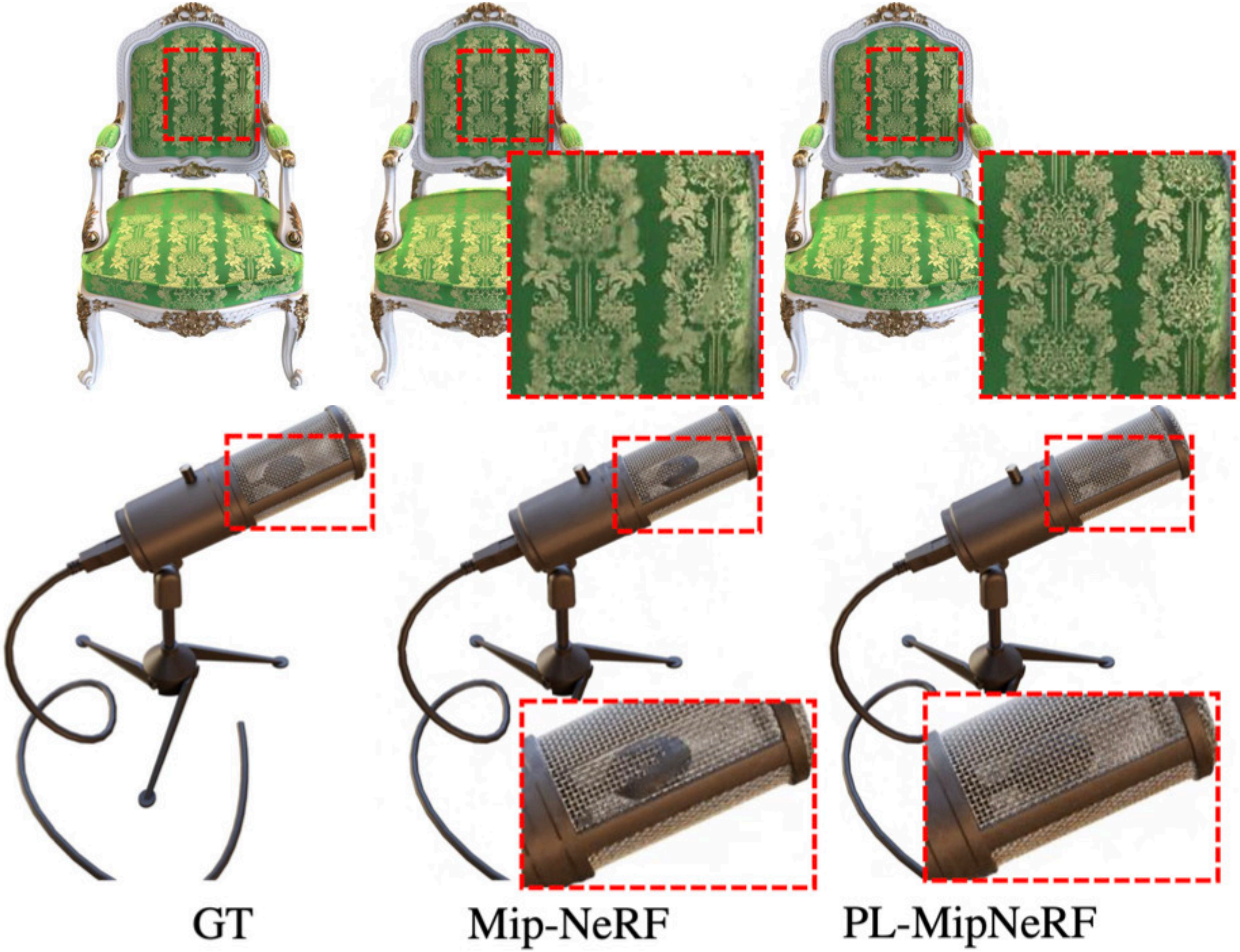
Results: Better Geometry Extraction



Blender		Avg.	Chair	Drums	Ficus	Hotdog	Lego	Mat.	Mic	Ship
CD↓	Vanilla NeRF	10.43	5.162	**6.842**	29.94	7.555	7.474	6.833	5.214	11.44
	PL-NeRF	**10.10**	**4.676**	7.754	**29.58**	**7.004**	**6.825**	**6.061**	**5.213**	**10.44**

Table 3: **Geometry Extraction Error** Distance between the surface of the GT to the predicted meshes. Scores are $\times 10^3$

Drop-in Replacement in Existing Methods



Drop-in Replacement in Existing Methods

Blender		Avg.	Chair	Drums	Ficus	Hotdog	Lego	Mat.	Mic	Ship
PSNR↑	Mip-NeRF	31.76	33.95	24.39	31.20	36.12	33.84	30.55	34.63	29.41
	PL-MipNeRF	**32.48**	**35.11**	**24.92**	**32.25**	**36.51**	**35.15**	**30.69**	**35.22**	**30.00**
SSIM↑	Mip-NeRF	0.955	0.975	0.921	0.971	0.978	0.971	0.957	0.987	0.876
	PL-MipNeRF	**0.959**	**0.981**	**0.928**	**0.977**	**0.980**	**0.976**	**0.959**	**0.989**	**0.882**
LPIPS↓	Mip-NeRF	3.64	1.80	6.82	2.35	1.97	1.44	2.39	0.973	11.4
	PL-MipNeRF	**3.09**	**1.32**	**5.78**	**1.66**	**1.67**	**1.07**	**2.09**	**0.788**	**10.3**

Table 1: **Quantitative Results of Mip-NeRF v.s. PL-MipNeRF** Reported LPIPS scores are multiplied by 10^2

Blender		Avg.	Chair	Drums	Ficus	Hotdog	Lego	Mat.	Mic	Ship
PSNR↑	DIVeR	30.78	32.01	**24.72**	30.1	**35.94**	29.03	29.31	32.10	**29.08**
	PL-DIVeR	**30.88**	**32.92**	24.7	**30.23**	**35.94**	**33.42**	**32.06**	**33.08**	28.99
SSIM↑	DIVeR	0.956	0.959	**0.917**	**0.963**	0.974	0.965	**0.977**	0.978	0.870
	PL-DIVeR	**0.947**	**0.969**	0.916	**0.963**	**0.966**	**0.966**	**0.977**	**0.981**	**0.871**
LPIPS↓	DIVeR	3.39	2.79	6.13	2.34	1.92	**1.46**	**1.77**	2.16	**7.77**
	PL-DIVeR	**3.28**	**2.85**	**6.01**	**2.12**	**1.83**	1.49	**1.77**	**1.73**	7.82

Table 2: **Quantitative Results of DIVeR v.s. PL-DIVeR** Reported LPIPS scores are multiplied by 10^2

Thank you!

Visit our
Project Page!