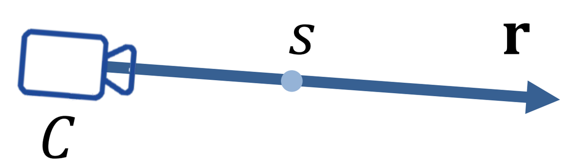
NeRF Revisited: Fixing Quadrature Instability in Volume Rendering

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

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

VOLUME RENDERING/NeRF -- AS WE KNOW IT

- Each infinitesimal particle in space (s) is with a certain **opacity** (τ_s) that emits a scalar **color** (c_s) for each direction.



- A pixel's value is the **expected color** $C(r)$ along the ray.

$$\mathbb{E}_{s\sim p(s)}[c(s)] = \int_0^\infty p(s)c(s)\,\mathrm{d}s = \int_0^\infty \tau(s)T(s)c(s)\,\mathrm{d}s\,.$$

  - Opacity $\tau(s)$: probability of hitting a particle at location
  - Transparency $\mathrm{T}(s)$: probability of not hitting anything until that location

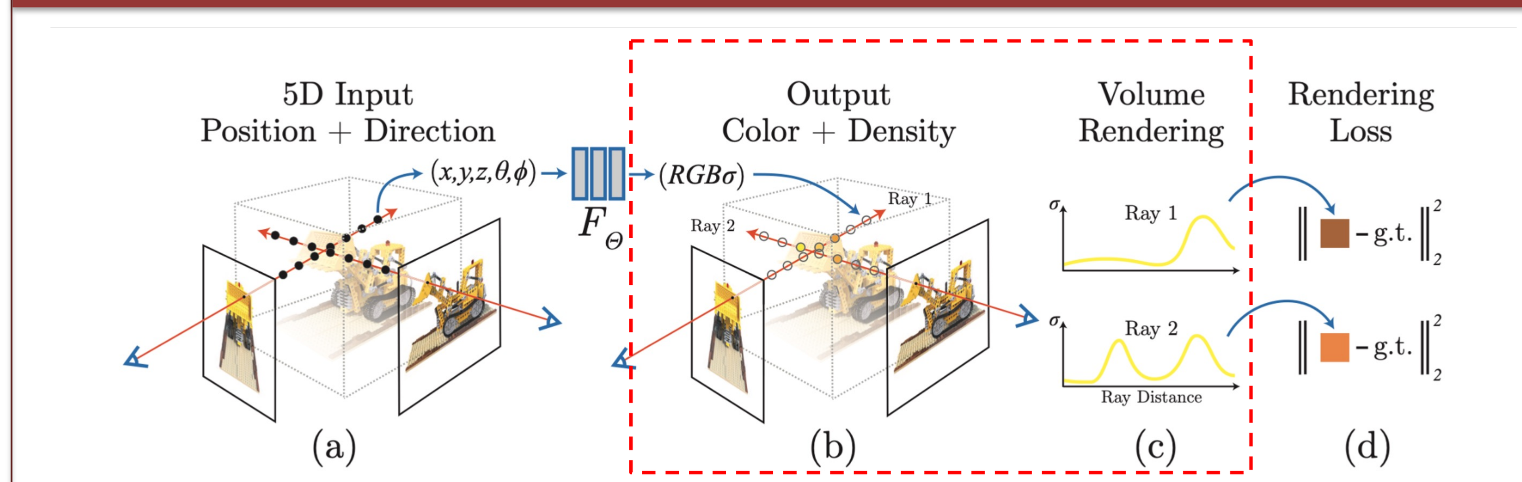
- Continuous:

$$C(\mathbf{r}) = \int_0^\infty \underline{T(s)}\,\underline{\tau(\mathbf{r}(s))}\,\underline{c(\mathbf{r}(s),\mathbf{d})}\mathrm{d}s, \quad \text{where } T(s) = \exp\left(-\int_0^s \tau(\mathbf{r}(t))\mathrm{d}t\right)$$

- Discrete (**the equation as we all know it**) :

$$\boxed{\hat{C}(\mathbf{r}) = \sum_{i=0}^{N} \underline{T_i}\Big(\underline{1-\exp\left(-\tau_i\delta_i\right)}\Big)\underline{c_i}, \quad \text{where } T_i = \exp\Big(-\sum_{j=0}^{i-1}\tau_j\delta_j\Big)}$$

OUR CONTRIBUTION



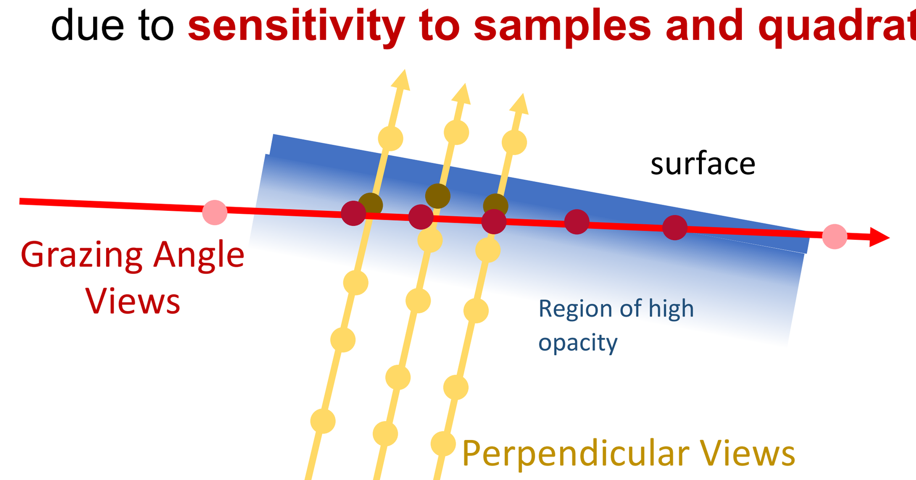
- We look into the actual **volume rendering equation**.
- The phenomena we investigate is general and **independent** of the model, the input, or the defined loss.

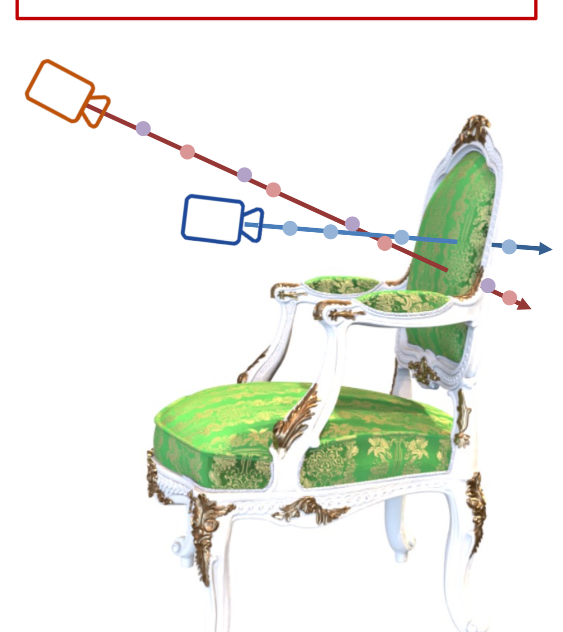
PROBLEM: QUADRATURE INSTABILITY

- The volume rendering equation as we all know it, assumes **piecewise constant opacity** at each interval.

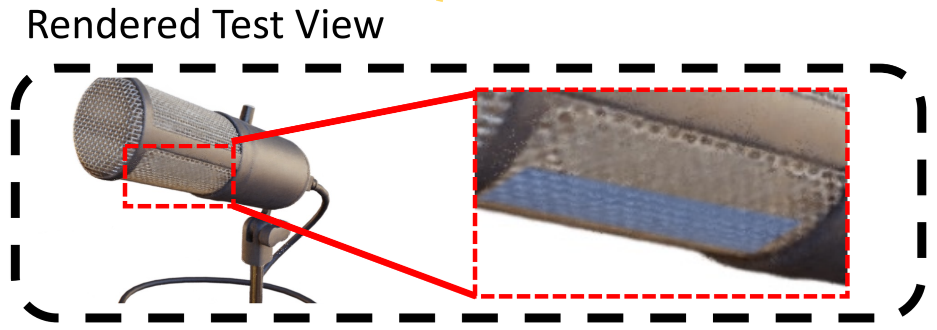
- Piecewise constant opacity introduces **ray conflicts** in NeRF optimization due to **sensitivity to samples and quadrature**.

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



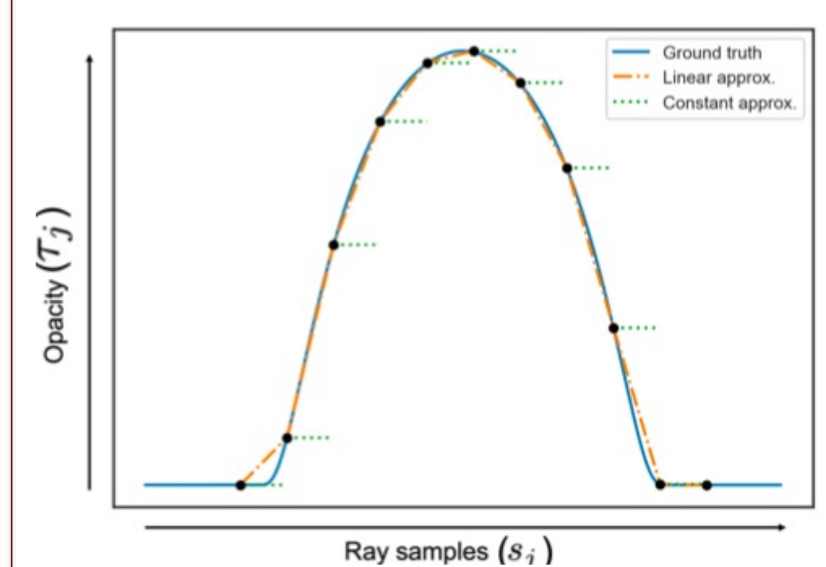


Rendered Test View

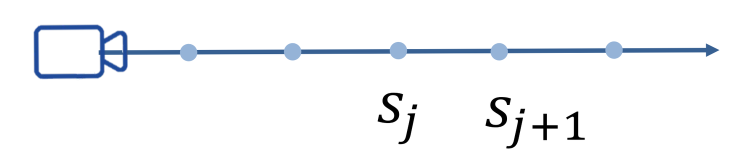


OUR PL-NERF : PIECEWISE LINEAR OPACITY

- To alleviate these concerns, we propose to use **piecewise linear opacity** and piecewise constant color at each interval.



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

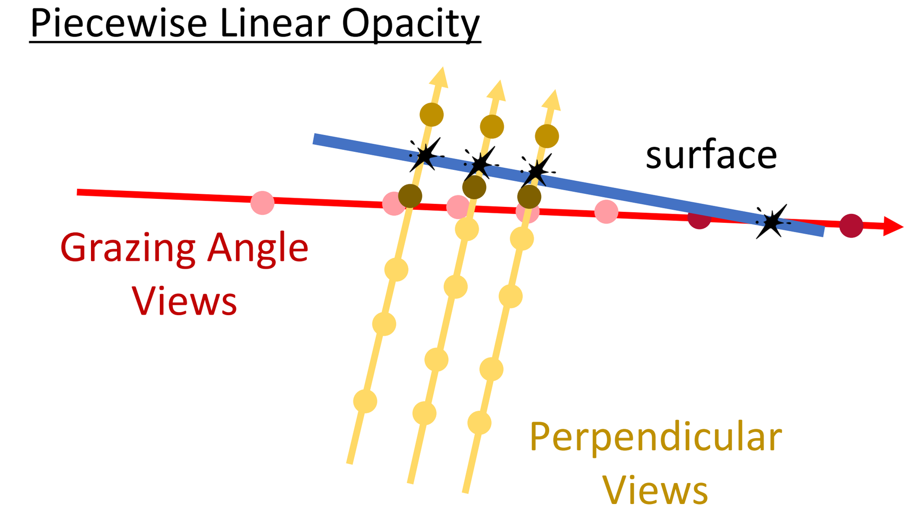


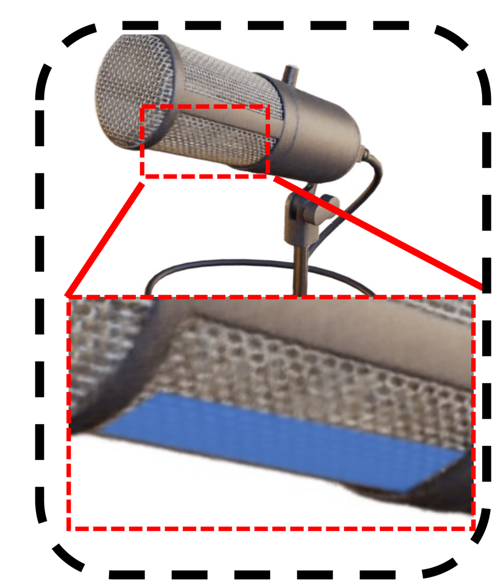
- We arrive at the following simple expressions:

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

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

Piecewise Linear Opacity





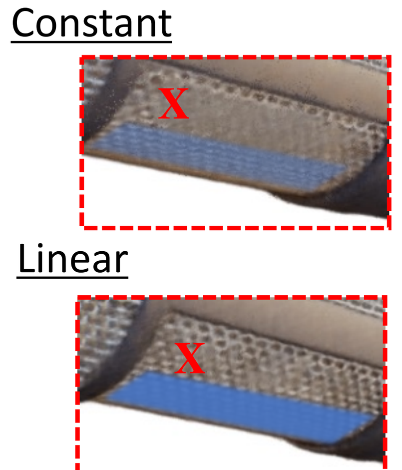
PRECISE IMPORTANCE SAMPLING

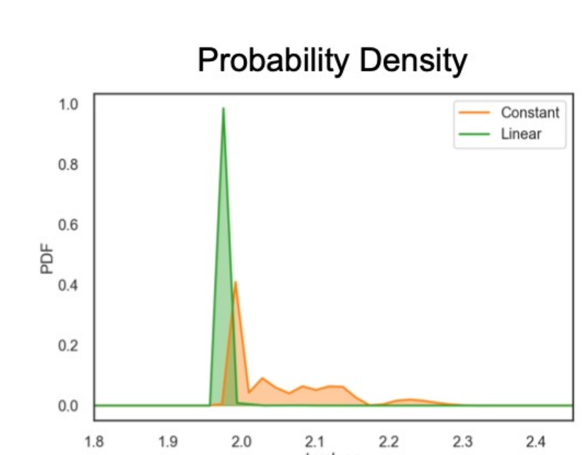
- One gets important samples from the probability density function (pdf) through **inverse transform sampling**.
- Under the piecewise constant opacity assumption, since the pdf is non-continuous, then its cdf is **non-invertible**.
- Under our piecewise linear opacity assumption, the cdf is **invertible** resulting in a closed-form solution for inverse transform sampling:

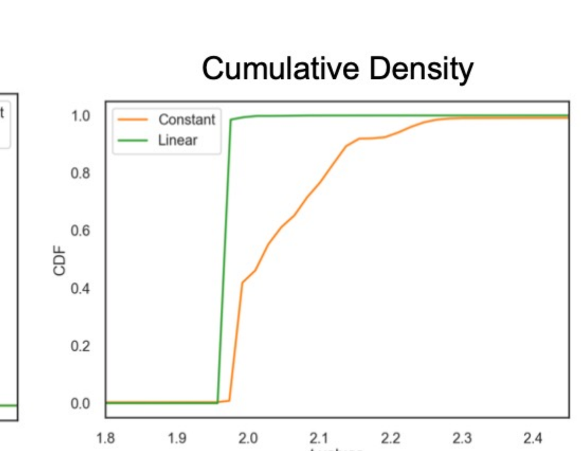
$$\boxed{t = \frac{s_{k+1}-s_k}{\tau_{k+1}-\tau_k}\left[-\tau_k + \sqrt{\tau_k^2 + \frac{2(\tau_{k+1}-\tau_k)\left(-\ln\frac{1-u}{T(s_k)}\right)}{(s_{k+1}-s_k)}}\,\right].}$$

Constant

Linear

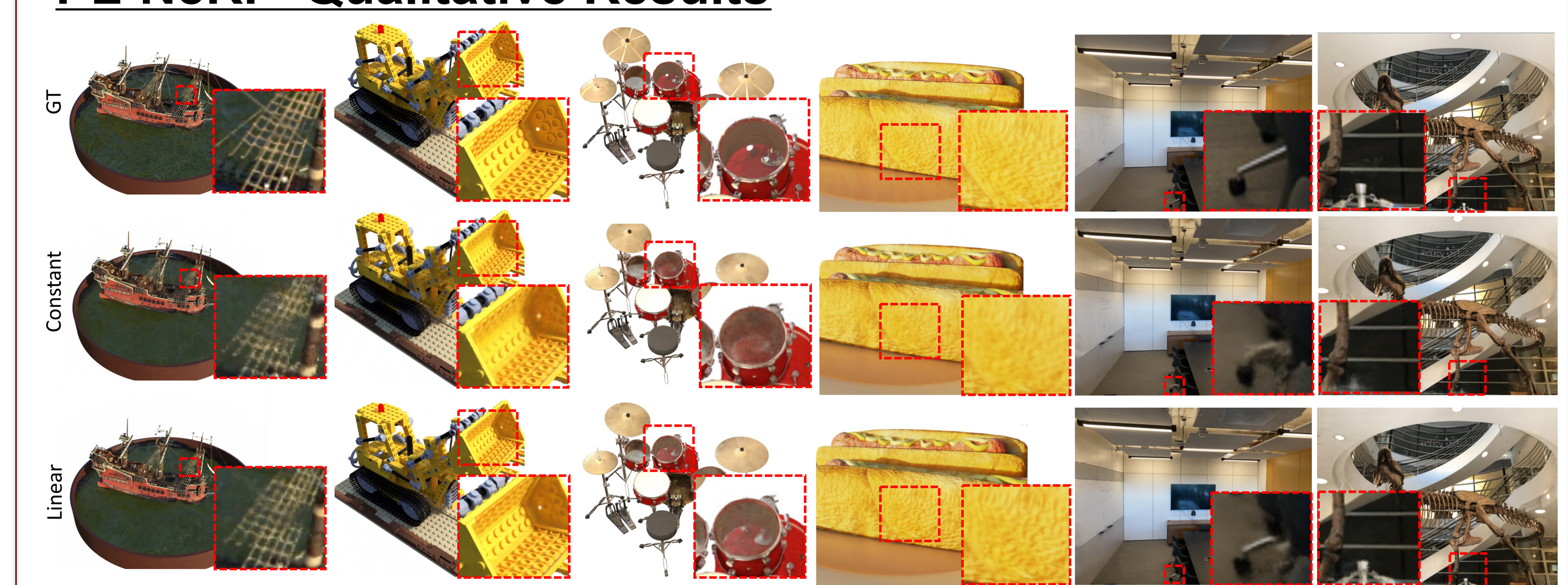






RESULTS

PL-NeRF Qualitative Results



PL- MipNeRF Qualitative Results

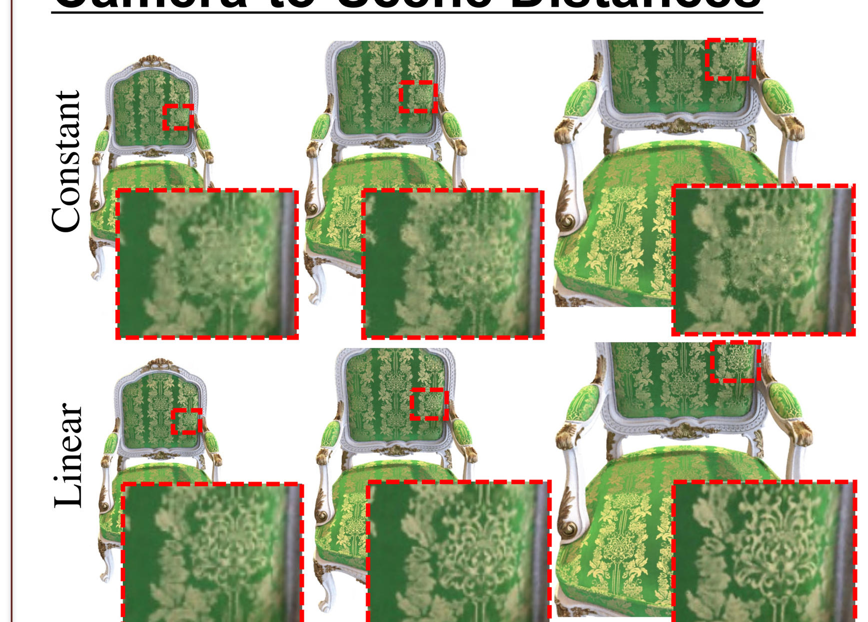
Quantitative Results

	Blender	Avg.
PSNR↑	Const. (Vanilla)	30.61
	Linear (Ours)	**31.10**
SSIM↑	Const. (Vanilla)	0.943
	Linear (Ours)	**0.948**
LPIPS↓	Const. (Vanilla)	5.17
	Linear (Ours)	**4.39**

	Blender	Avg.
PSNR↑	Mip-NeRF	31.76
	PL-MipNeRF	**32.48**
SSIM↑	Mip-NeRF	0.955
	PL-MipNeRF	**0.959**
LPIPS↓	Mip-NeRF	3.64
	PL-MipNeRF	**3.09**

	Blender	Avg.
PSNR↑	DIVeR	30.78
	PL-DIVeR	**30.88**
SSIM↑	DIVeR	0.956
	PL-DIVeR	**0.947**
LPIPS↓	DIVeR	3.39
	PL-DIVeR	**3.28**

Camera-to-Scene Distances



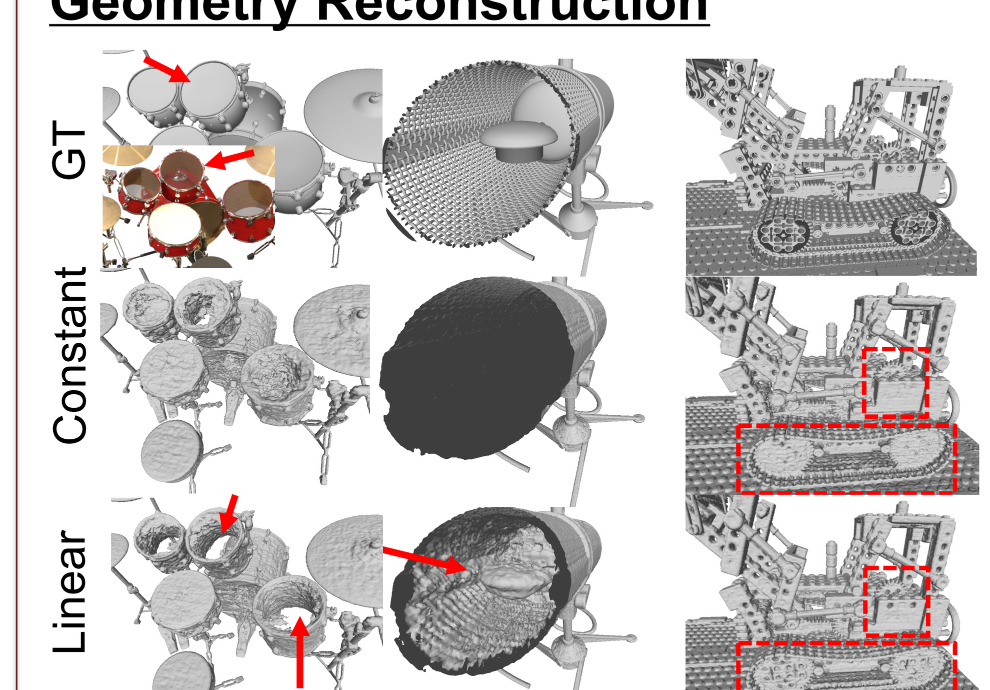
LLFF Results

	RFF	Avg.
PSNR↑	Const. (Vanilla)	27.53
	Linear (Ours)	**28.05**
SSIM↑	Const. (Vanilla)	0.874
	Linear (Ours)	**0.885**
LPIPS↓	Const. (Vanilla)	7.37
	Linear (Ours)	**6.06**

DTU Results

	PSNR↑	SSIM↑	LPIPS↓
Const. (Vanilla)	27.96	0.909	8.58
Linear (Ours)	**28.43**	**0.918**	**7.73**

Geometry Reconstruction



	Blender	Avg.
CD↓	Vanilla NeRF	10.43
	PL-NeRF	**10.10**

Depth Supervision

	PSNR↑	SSIM↑	LPIPS↓	RMSE↓
Const. (Vanilla)	29.20	0.898	11.2	0.178
Linear (Ours)	**29.54**	**0.905**	**10.4**	**0.147**

Table 3: **Depth Supervision.** Reported LPIPS score is multiplied by 10^2.