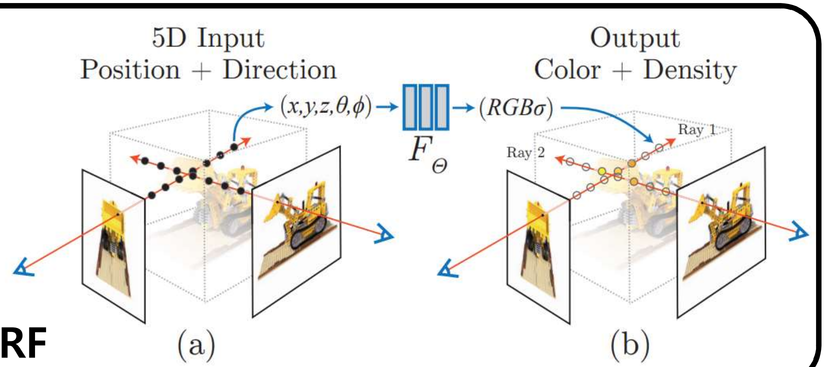
CS580: Student Presentation
mipNeRF & mipNeRF360

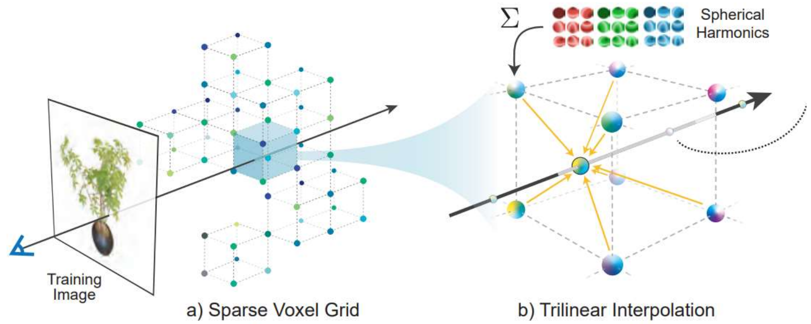
Dongyoung Choi
(최동영)

Recap. Previous Presentation



NeRF

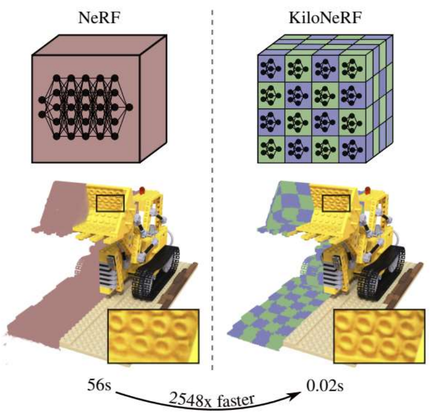
Acceleration by **sparse voxel grid**



Plenoxels

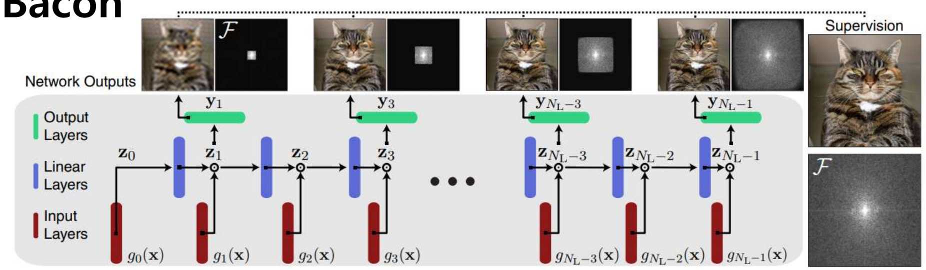
Acceleration by **thousand of small MLPs**

Can be applied to **neural radiance field**



KiloNeRF

Bacon



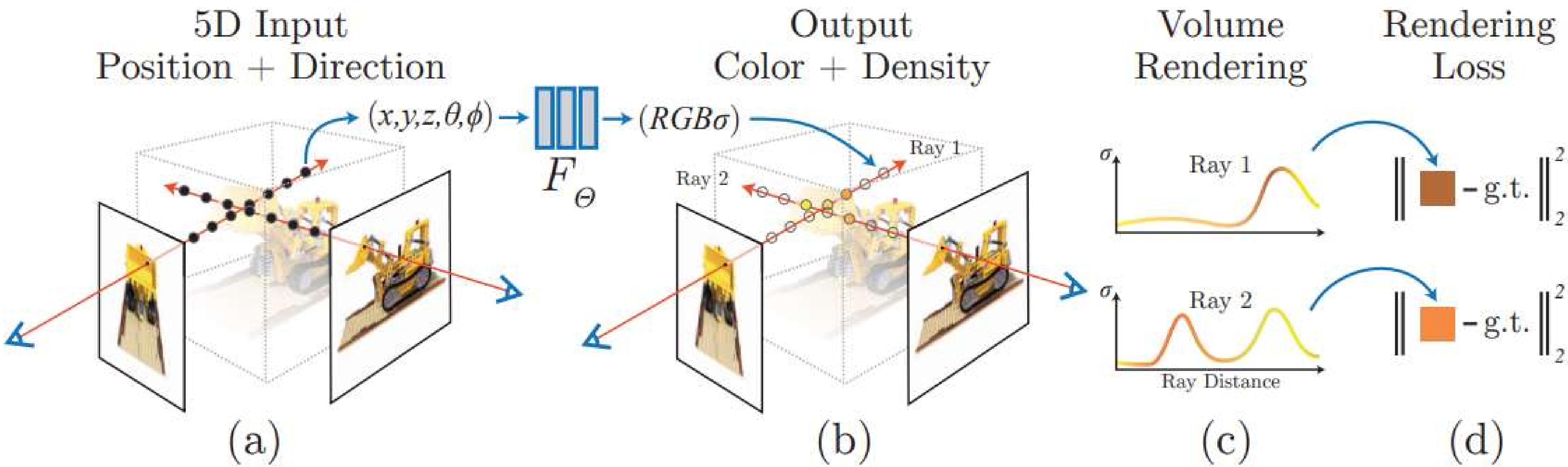
CS580: Student Presentation
mipNeRF

ICCV 2021 Oral, Best Paper Honorable Mention
Jonathan T. Barron et al.

KAIST

Recap. NeRF

- **Render 3D object using 2D Images with camera matrix by optimizing radiance field equation via neural network**



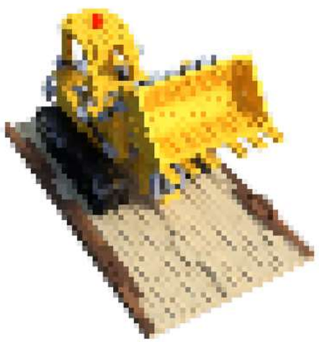
Recap. NeRF (conti.)

- NeRF can even generate unobserved view of trained object

Result of NeRF

Aliasing in multi-resolution images

- Aliasing occurs when NeRF learns **low resolution(1/2, ¼, …) images**



NeRF

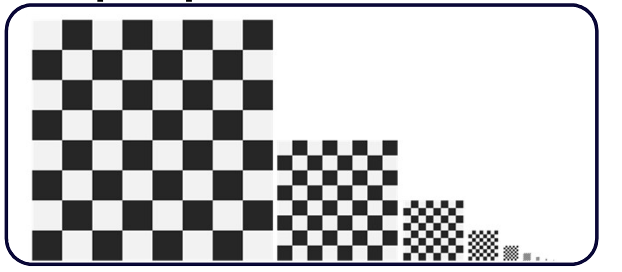
mipNeRF

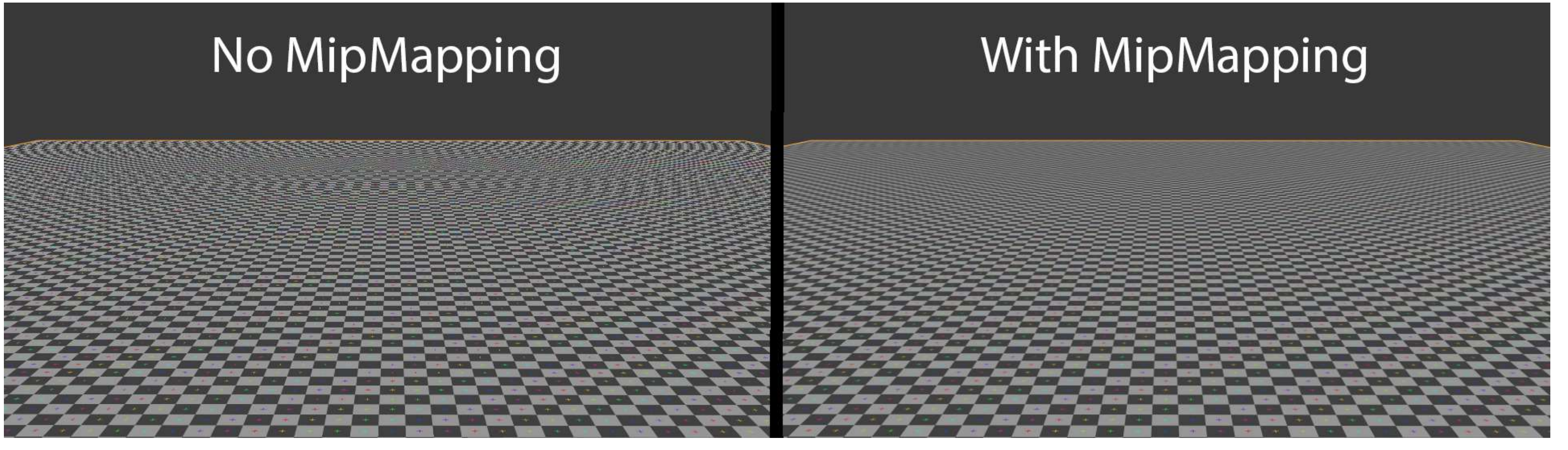
Ground Truth **NeRF**

Mipmap

- Aliasing effects can be reduced by using pre-computed gaussian filtered images **(prefiltering)**

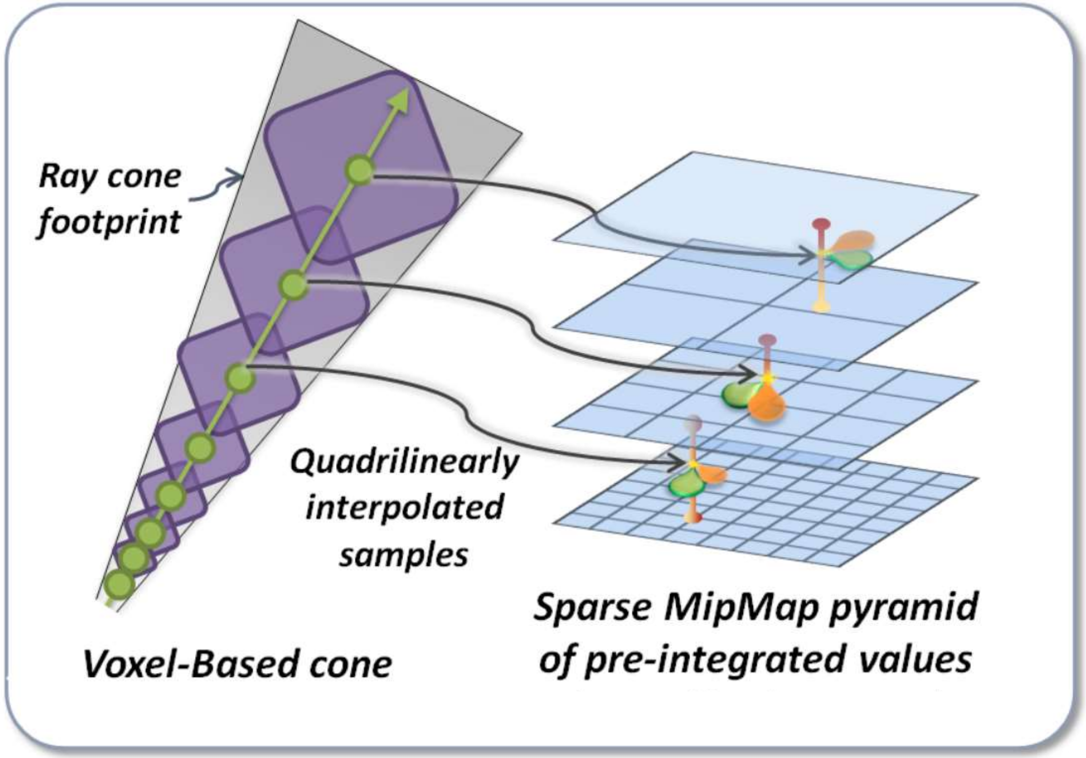
mipmap





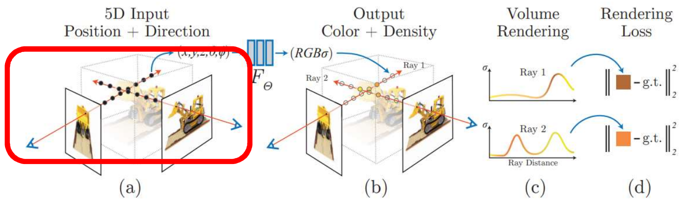
Cone tracing

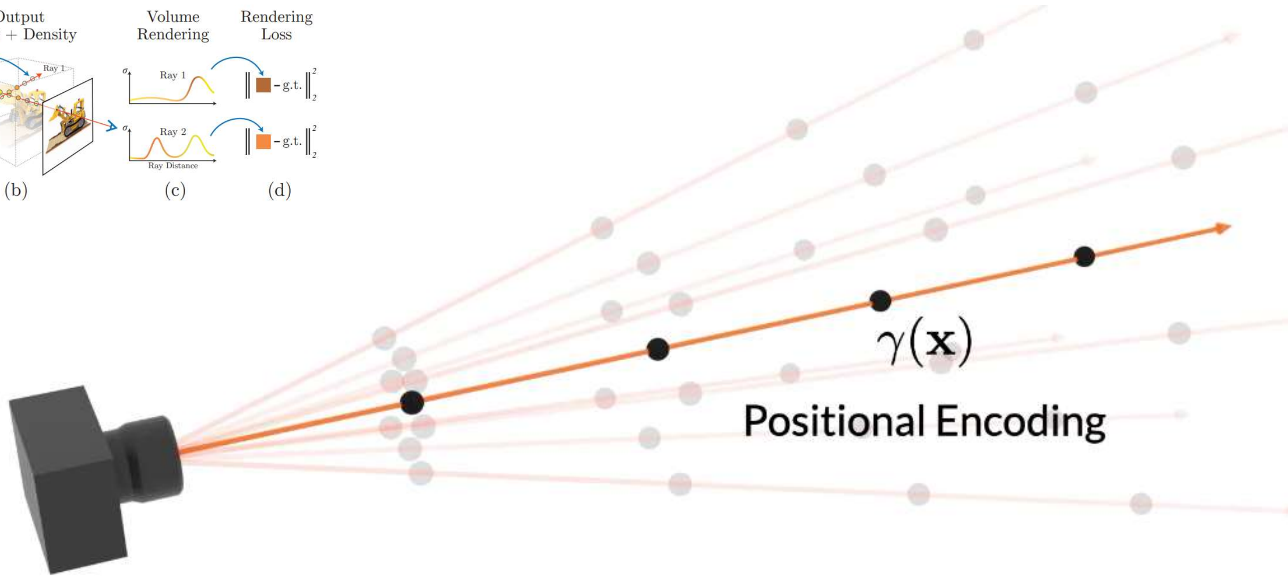
- mipNeRF uses **cone tracing** that acts like mipmap in rendering



Ray casting in NeRF

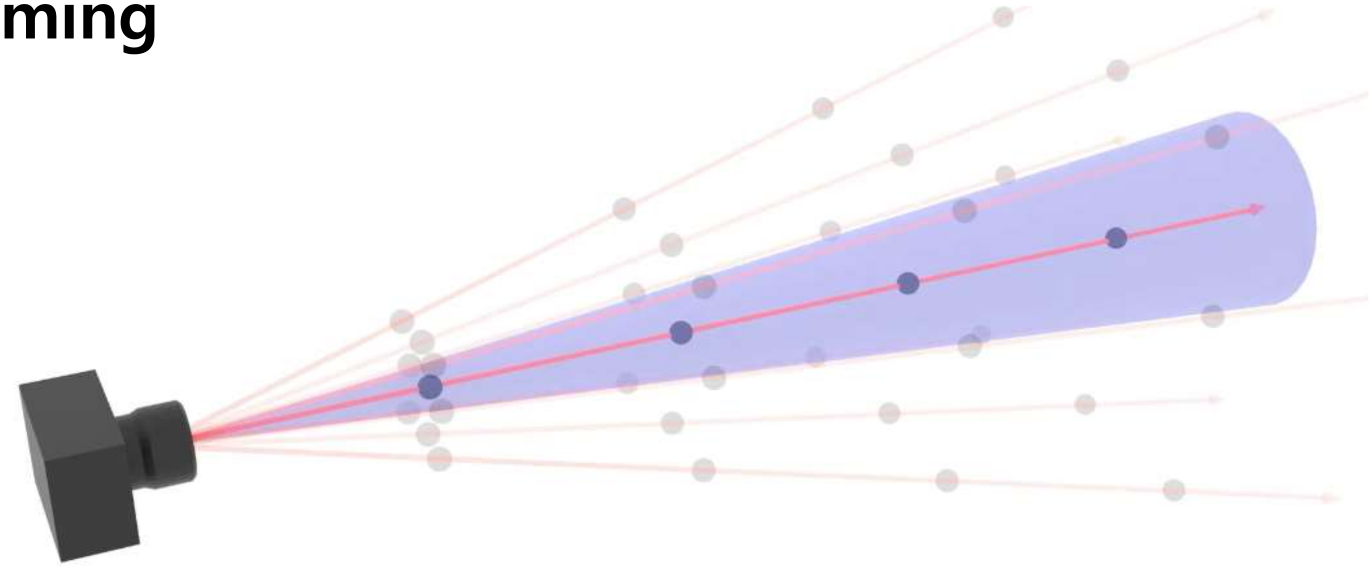
- NeRF casts a **ray** for sampling the points





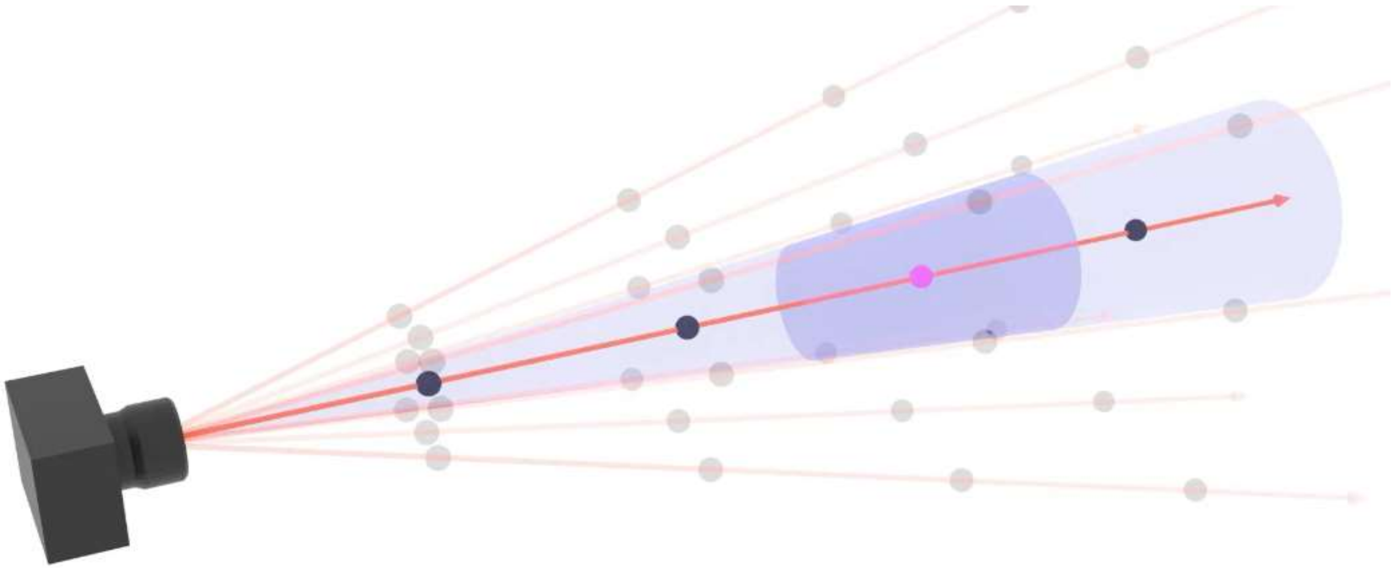
Cone casting in mipNeRF

- Instead, mipNeRF casts a **cone** to samples
- But sampling all points in the cones is **extremely time consuming**



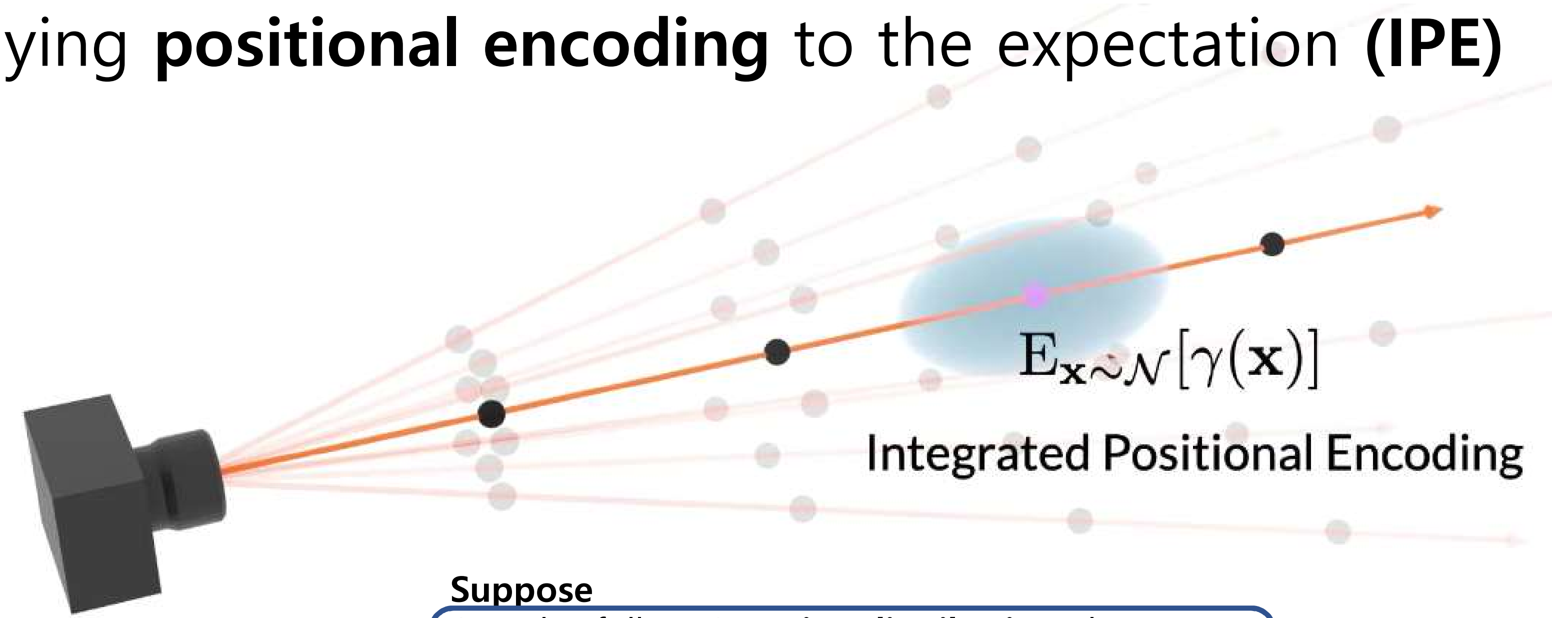
Cone casting in mipNeRF (cont.)

- Divide cone as **multiple conical frustum**



Cone casting in mipNeRF (cont.)

- Use the **expectation** of each conical frustum as samples!
- Suppose all samples follow the **Gaussian distribution**
- Applying **positional encoding** to the expectation **(IPE)**



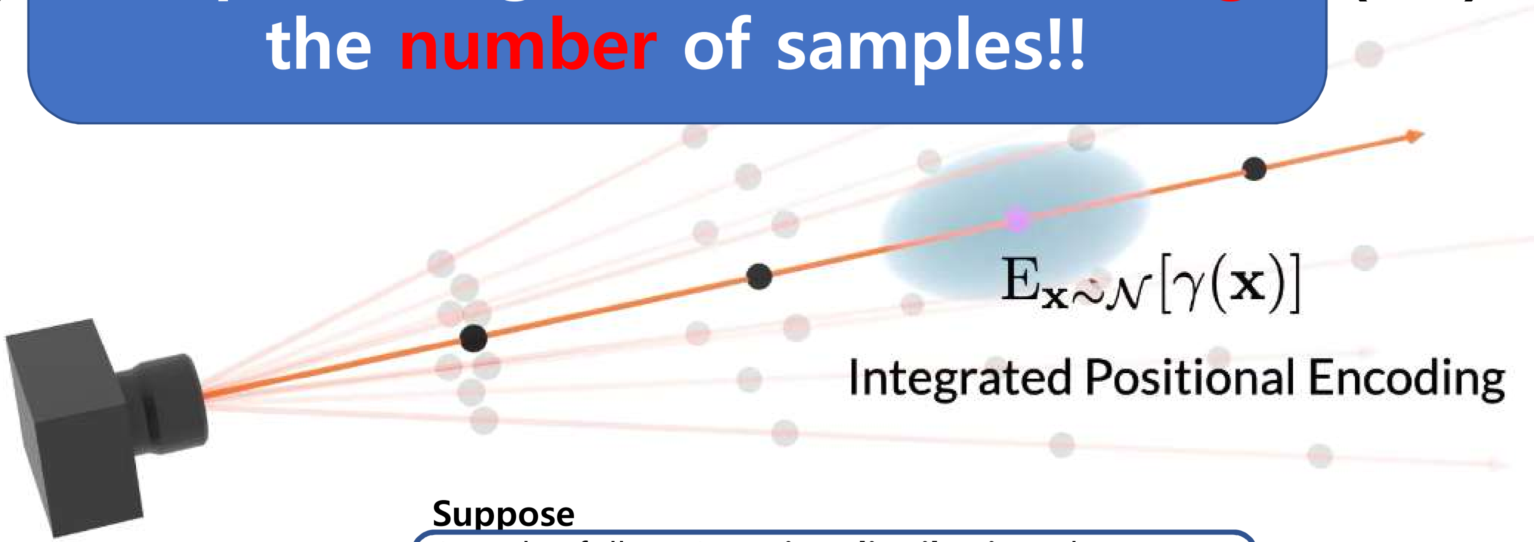
Suppose

Samples follow **Gaussian distribution** along a ray direction and it vertical one (multivariate Gaussian distribution)

Cone casting in mipNeRF (cont.)

- Use the **expectation** of each conical frustum as samples!
- Suppo … **ution**
- Applyi … **(IPE)**

Achieve higher performance with simple changes without increasing the number of samples!!



Suppose

Samples follow **Gaussian distribution** along a ray direction and it vertical one (multivariate Gaussian distribution)

Integrated Positional Encoding (IPE)

Positional Encoding

$$P = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 & 0 & & 2^{L-1} & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 & \cdots & 0 & 2^{L-1} & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & & 0 & 0 & 2^{L-1} \end{bmatrix}^T, \quad \gamma(\mathrm{x}) = \begin{bmatrix} \sin(\mathrm{Px}) \\ \cos(\mathrm{Px}) \end{bmatrix}$$

Conical frustum equation

Expectation

Positional encoding

$$E_{\mathrm{x} \sim N(\mu, \sigma^2)}[\gamma(\mathrm{x})]$$

Normal distribution

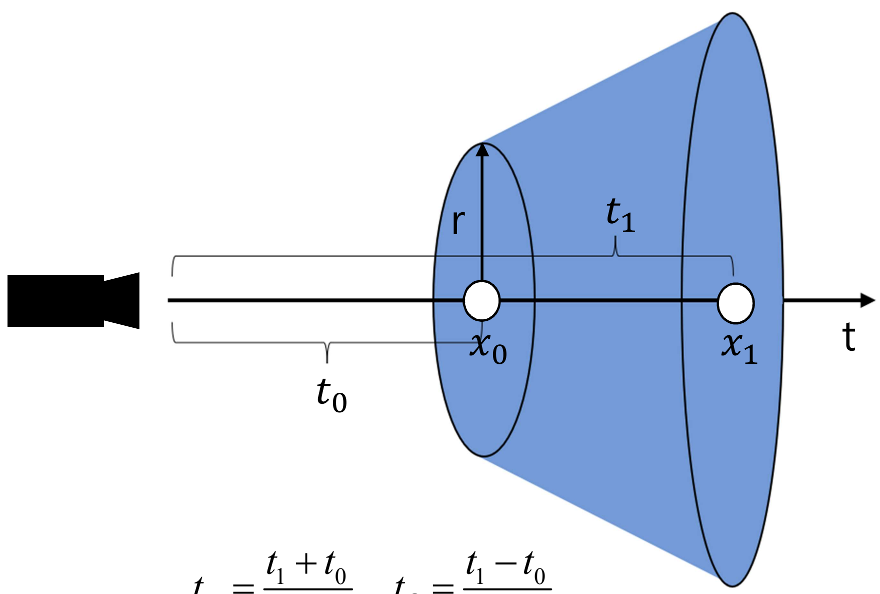
~

$$F(x, o, d, \dot{r}, t_0, t_1) = \left\{ \left(t_0 < \frac{d^T(x-o)}{\|d\|_2^2} < t_1 \right) \wedge \left(\frac{d^T(x-o)}{\|d\|_2 \|x-o\|_2} > \frac{1}{\sqrt{1 + (\dot{r} / \|d\|_2)^2}} \right) \right\}$$

x are the member of conical frustum

Integrated Positional Encoding (IPE) (cont.)

- Mean and **Var** is determined by the **distance** between samples!



$$t_\mu = \frac{t_1 + t_0}{2}, \quad t_\delta = \frac{t_1 - t_0}{2}$$

Mean, Covariance matrix of conical frustum

$$\mu_t = t_\mu + \frac{2 t_\mu t_\delta^2}{3t_\mu^2 + t_\delta^2}, \quad \sigma_t^2 = \frac{t_\delta^2}{3} - \frac{4t_\delta^2(12t_\mu^2 - t_\delta^2)}{15(t_\mu^2 + t_\delta^2)^2},$$

$$\sigma_r^2 = \dot{r}^2\left(\frac{t_\mu^2}{4} + \frac{5t_\delta^2}{12} - \frac{4t_\delta^4}{15(3t_\mu^2 + t_\delta^2)}\right)$$

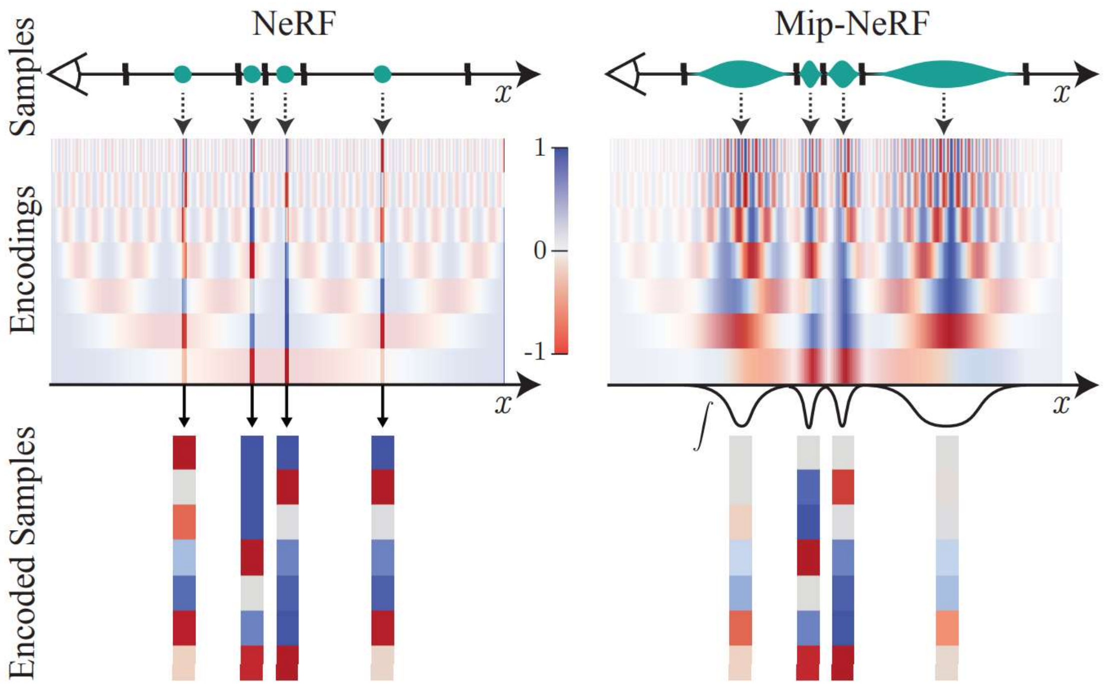
cone dir

radius dir

$$\mu = o + \mu_t d, \quad \Sigma = \sigma_t(dd^{\mathrm{T}}) + \sigma_r\left(I - \frac{dd^{\mathrm{T}}}{\|d\|_2^2}\right)$$

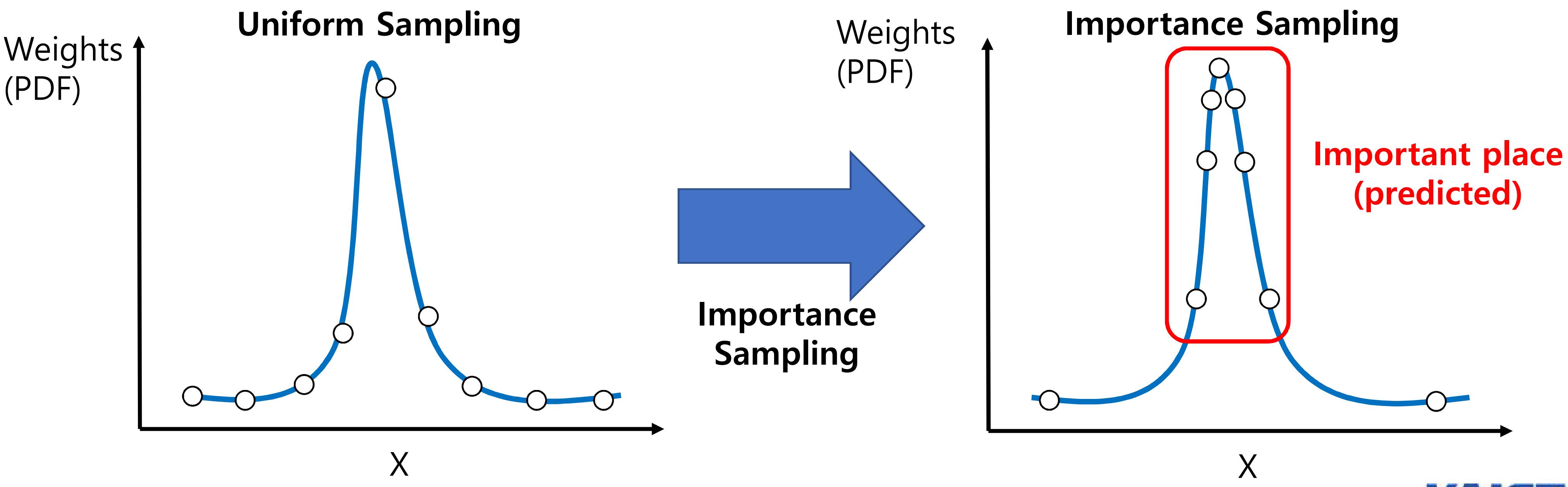
Result of IPE

- Get the samples including **variance information** implicitly



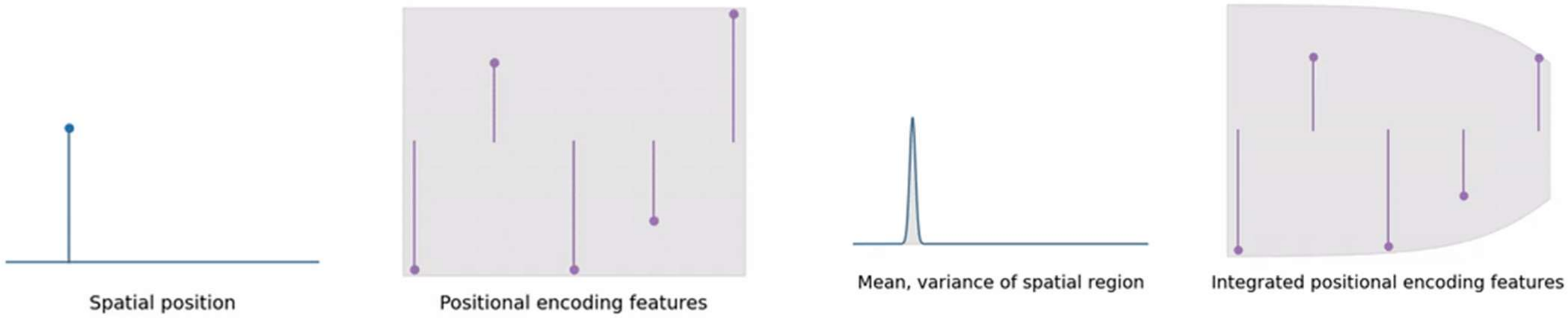
Importance Sampling

- Increase the number of samples predicted to be important
- **The important samples distance would be really short!**



Result of IPE (conti.)

- The distance between samples is **large**
  -> **cut off** the high freq. (**details**)
- The distance between samples is **short**
  -> **maintain** the high freq. (**details**)



KAIST

Result of IPE (conti.)

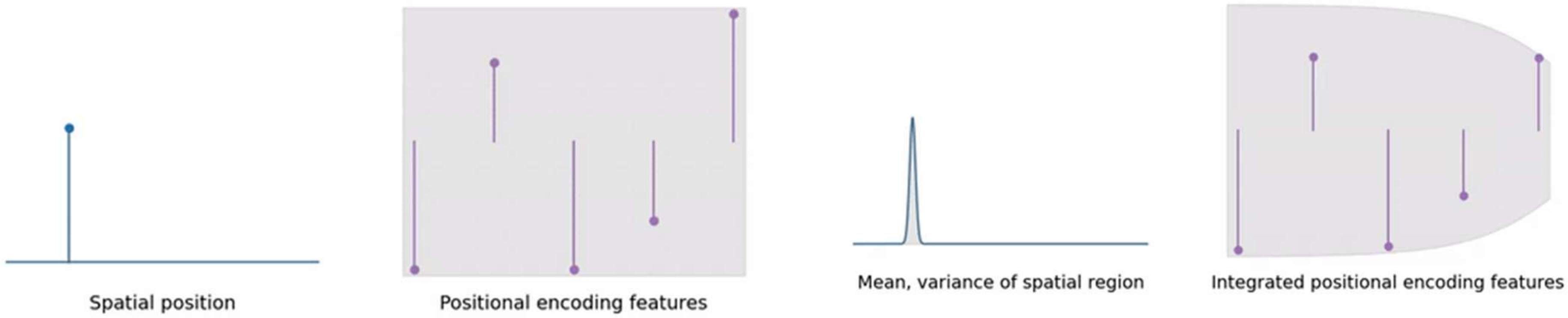
- The distance between samples is **large**

  -> **cu**

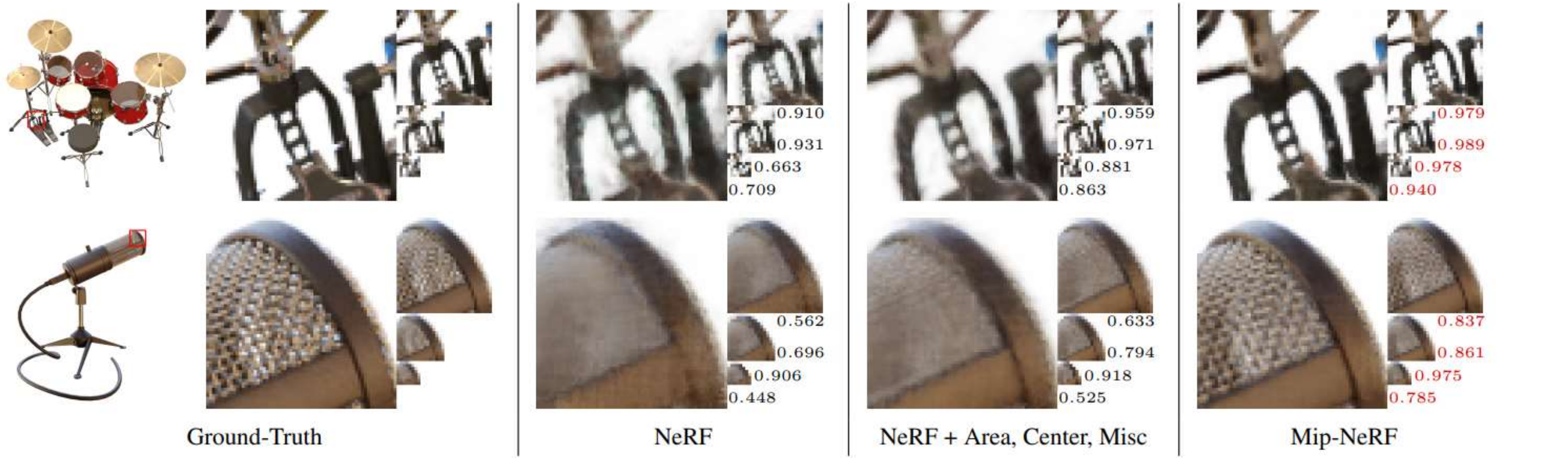
- The dista

  -> **m**

Keep the details of important samples and omit the details of unimportant samples autonomously!



Result



	PSNR ↑				SSIM ↑				LPIPS ↓						
	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Full Res.	1/2 Res.	1/4 Res.	1/8 Res.	Full Res.	1/2 Res.	1/4 Res.	1/8 Res	Avg. ↓	Time (hours)	# Params
NeRF (Jax Impl.) [11, 30]	31.196	30.647	26.252	22.533	0.9498	0.9560	0.9299	0.8709	0.0546	0.0342	0.0428	0.0750	0.0288	3.05 ± 0.04	1,191K
NeRF + Area Loss	27.224	29.578	29.445	25.039	0.9113	0.9394	0.9524	0.9176	0.1041	0.0677	0.0406	0.0469	0.0305	3.03 ± 0.03	1,191K
NeRF + Area, Centered Pixels	29.893	32.118	33.399	29.463	0.9376	0.9590	0.9728	0.9620	0.0747	0.0405	0.0245	0.0398	0.0191	3.02 ± 0.05	1,191K
NeRF + Area, Center, Misc.	29.900	32.127	33.404	29.470	0.9378	0.9592	0.9730	0.9622	0.0743	0.0402	0.0243	0.0394	0.0190	2.94 ± 0.02	1,191K
Mip-NeRF	32.629	34.336	35.471	35.602	0.9579	0.9703	0.9786	0.9833	0.0469	0.0260	0.0168	0.0120	0.0114	2.84 ± 0.01	612K
Mip-NeRF w/o Misc.	32.610	34.333	35.497	35.638	0.9577	0.9703	0.9787	0.9834	0.0470	0.0259	0.0167	0.0120	0.0114	2.82 ± 0.03	612K
Mip-NeRF w/o Single MLP	32.401	34.131	35.462	35.967	0.9566	0.9693	0.9780	0.9834	0.0479	0.0268	0.0169	0.0116	0.0115	3.40 ± 0.01	1,191K
Mip-NeRF w/o Area Loss	33.059	34.280	33.866	30.714	0.9605	0.9704	0.9747	0.9679	0.0427	0.0256	0.0213	0.0308	0.0139	2.82 ± 0.01	612K
Mip-NeRF w/o IPE	29.876	32.160	33.679	29.647	0.9384	0.9602	0.9742	0.9633	0.0742	0.0393	0.0226	0.0378	0.0186	2.79 ± 0.01	612K

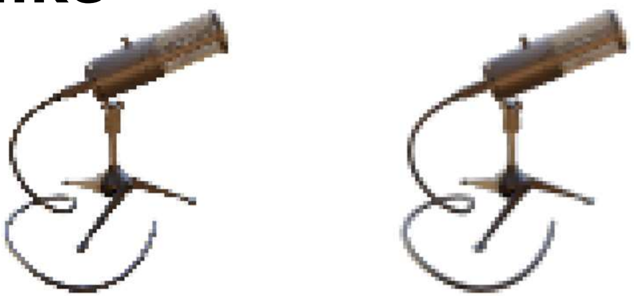
Result

- mipNeRF shows better performance in reducing aliasing!

Lego

Ship

Mike



Chair

Summary of mipNeRF

- Use **cone tracing** instead of ray for **mipmapping**
- Compute **expectations** of samples without sampling whole data
- Propose **Integrated Positional Encoding (IPE)** which autonomously adjusts the details of samples according to the importance

CS580: Student Presentation
mipNeRF 360

CVPR 2022 Oral
Jonathan T. Barron et al.

Obstacles in NeRF at 360˚ Scene

- **Parameterization**

  Samples of 360˚scene would be located from 0 to infinity $(0, \infty)$ (Unbounded scene)

  ← Scene and ray parameterization

- **Efficiency**

  Huge model would be required but need long training time

  ← Coarse-to-Fine Online Distillation

- **Ambiguity**

  Since the sample range is too broad, predicting object geometry is challenging

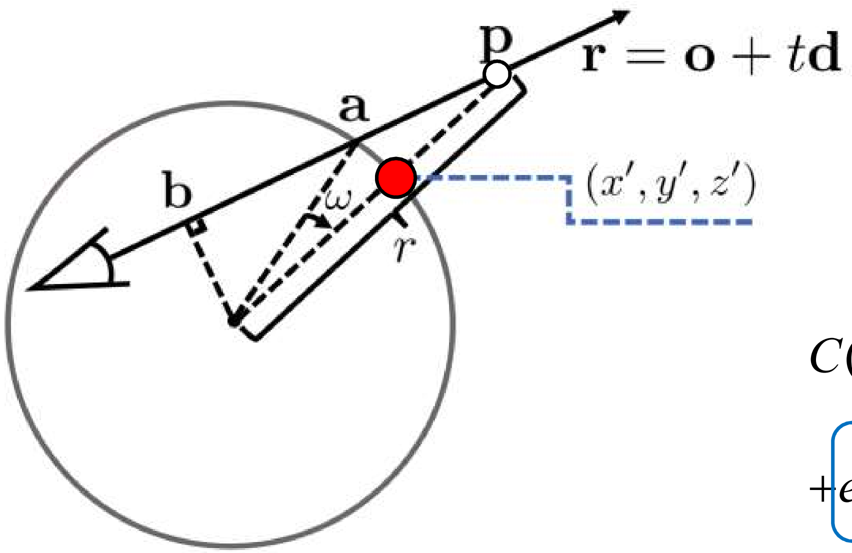
  ← Regularization for Interval-Based Models

KAIST

NeRF++: Unbounded scene (2020, arXiv)

Parameterization

- If sample distance > 1:
  use the color&density of samples projected to the sphere



Use the color and density of red points when computing white samples

$$C(r) = \underbrace{\int_{t=0}^{t'} \sigma(o+td)\cdot c(o+td,d)\cdot e^{-\int_{s=0}^{t}\sigma(o+sd)ds}\,dt}_{\text{Inside}}$$

$$+\underbrace{e^{-\int_{s=0}^{t'}\sigma(o+sd)ds}}_{\text{Inside}}\underbrace{\int_{t=t'}^{\infty} \sigma_{out}(o+td)\cdot c_{out}(o+td,d)\cdot e^{-\int_{s=t'}^{t}\sigma_{out}(o+sd)ds}\,dt.}_{\text{Outside}}$$

Inside

Outside

DONeRF: Unbounded scene (2021, Eurographics)

Parameterization

- Sampling **logarithmically** and then **warping** the samples

Uniform sampling

$$\mathbf{x}(d_i) = \mathbf{o} + d_i \cdot \mathbf{r}$$

$$d_i = \left(d_{min} + i \cdot \frac{(d_{max} - d_{min})}{N} \right), \; i = [0, 1, 2, \cdots, N],$$

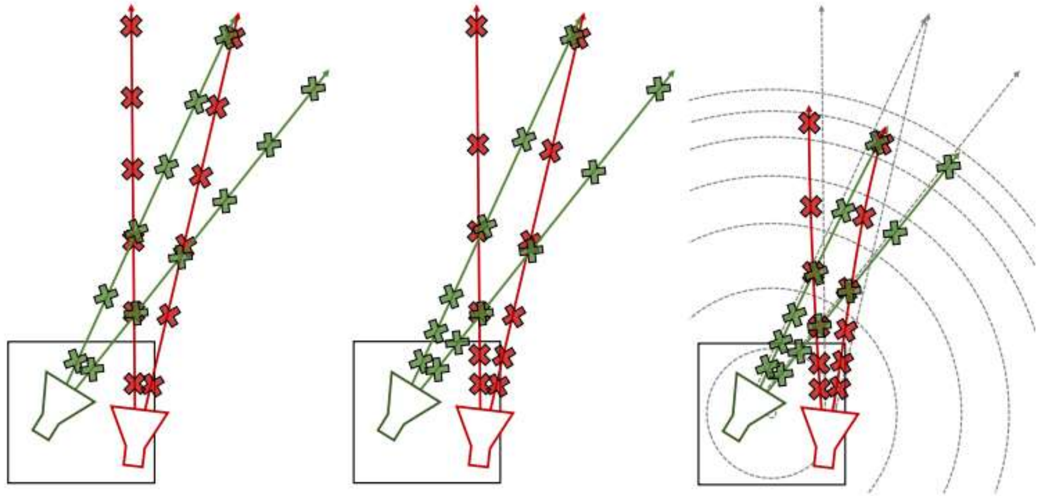
Logarithmic sampling

$$\tilde{d}_i = d_{min} + \frac{\log(d_i - d_{min} + 1)}{\log(d_{max} - d_{min} + 1)} \cdot (d_{max} - d_{min}).$$

Log + Warping

$$f = PE.\left(x(\tilde{d}_i - c) \cdot W(x(\tilde{d}_i - c) \right)$$

$$W(x) = \frac{1}{\sqrt{|x| \cdot d_{\max}}}$$



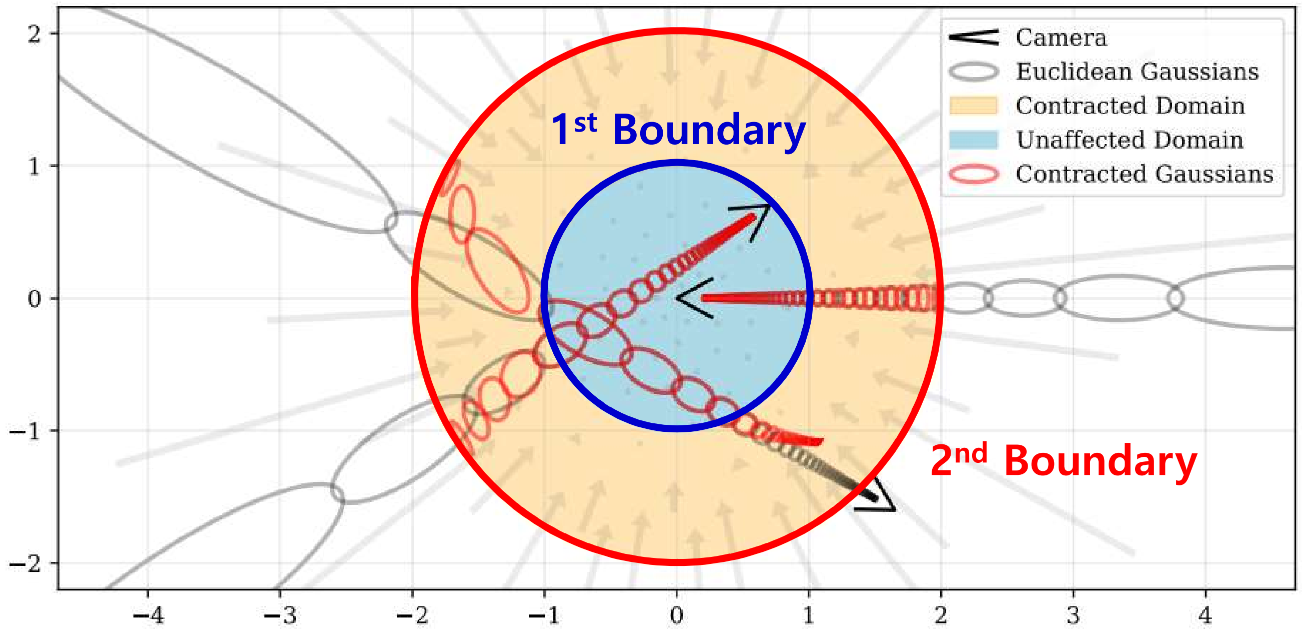
(a) *uniform* (b) *logarithmic* (c) *log+warp*

KAIST

mipNeRF360: Rescaling for Unbounded scene

Parameterization

- Similar to NeRF++, set the **boundaries (but two)**. If samples exceed the 1st boundary then would be converged on 2nd boundary!



Contraction function f (linear transformation)

$$f(x) \approx f(\mu) + J_f(\mu)(x-\mu)$$

$$f(\mu,\Sigma) = \left(f(\mu),\ J_f(\mu)\Sigma\, J_f(\mu)^T \right)$$

$$f = \text{contract}(x) = \begin{cases} x & \|x\| \le 1 \\ \left(2 - \frac{1}{\|x\|}\right)\left(\frac{x}{\|x\|}\right) & \|x\| > 1 \end{cases}$$

1st Boundary ($\|x\| \le 1$)

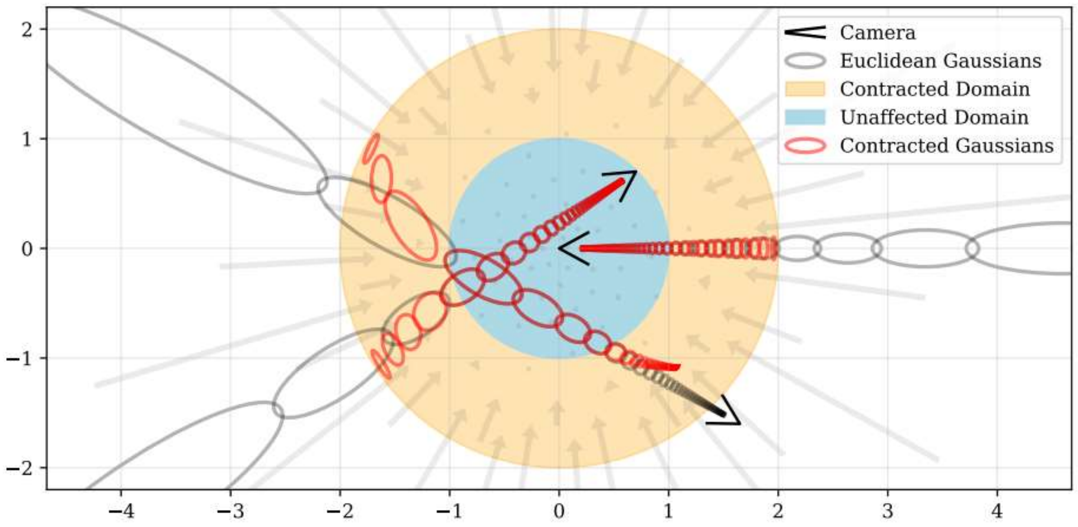
2nd Boundary (2)

PE of contraction: $\gamma\left(\text{contract}(\mu,\Sigma)\right)$

mipNeRF360: Sampling for Unbounded scene

Parameterization

- Similar to DONeRF, the farther away the wider the samples are
  - Uniform sampling on the **inverse of distance**
- Define **s-distance** to generalize sampling distance equation



$$s \in [0,1] \rightarrow [t_n, t_f]$$

$$s \triangleq \frac{g(t) - g(t_n)}{g(t_f) - g(t_n)}, \quad t \triangleq g^{-1}(s \cdot g(t_f) + (1-s) \cdot g(t_n))$$

t_n: near t, t_f: far t

$$g(x) = \frac{1}{x}$$

mipNeRF 360

$$g(x) = \log(x)$$

DONeRF spacing

mipNeRF360: Sampling for Unbounded scene

Parameterization

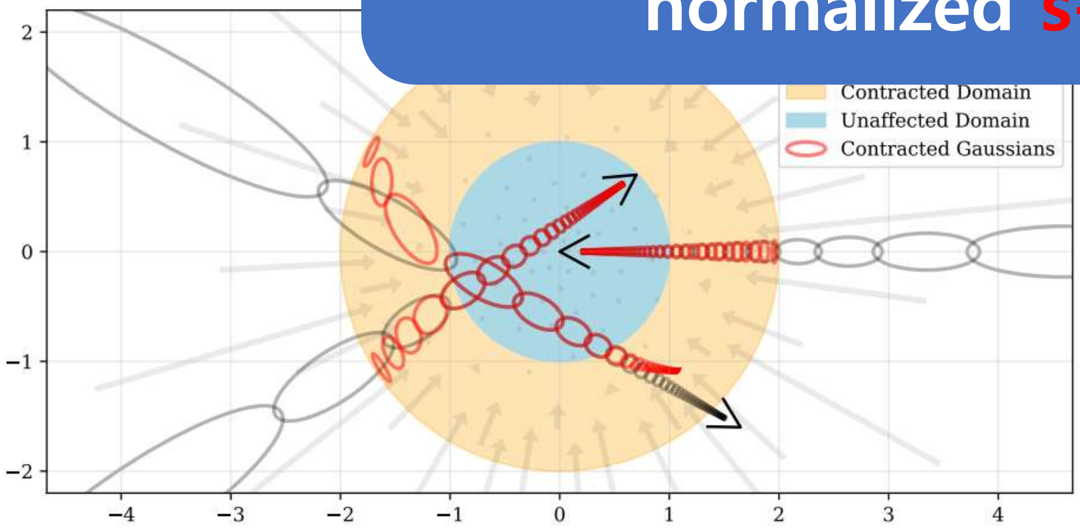
- Similar to DONeRF, the farther away the wider the samples are
  - Uniform
- Define **s-**

Contract coordinate effectively using 2 boundary warping and normalized s-distance

$[t_n, t_f]$

$$g(t_f) - g(t_n) \qquad (s \cdot g(t_f) + (1-s) \cdot g(t_n))$$

t_n: near t, t_f: far t



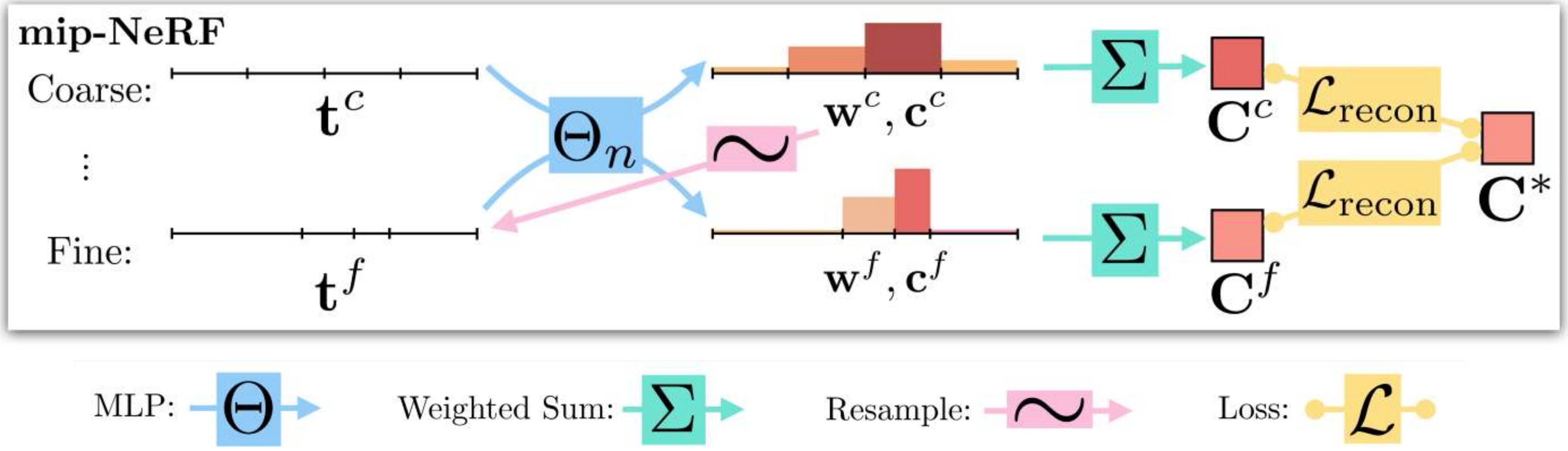
$g(x) = \frac{1}{x}$	$g(x) = \log(x)$
mipNeRF 360	**DONeRF spacing**

KAIST

mipNeRF: Importance sampling

Efficiency

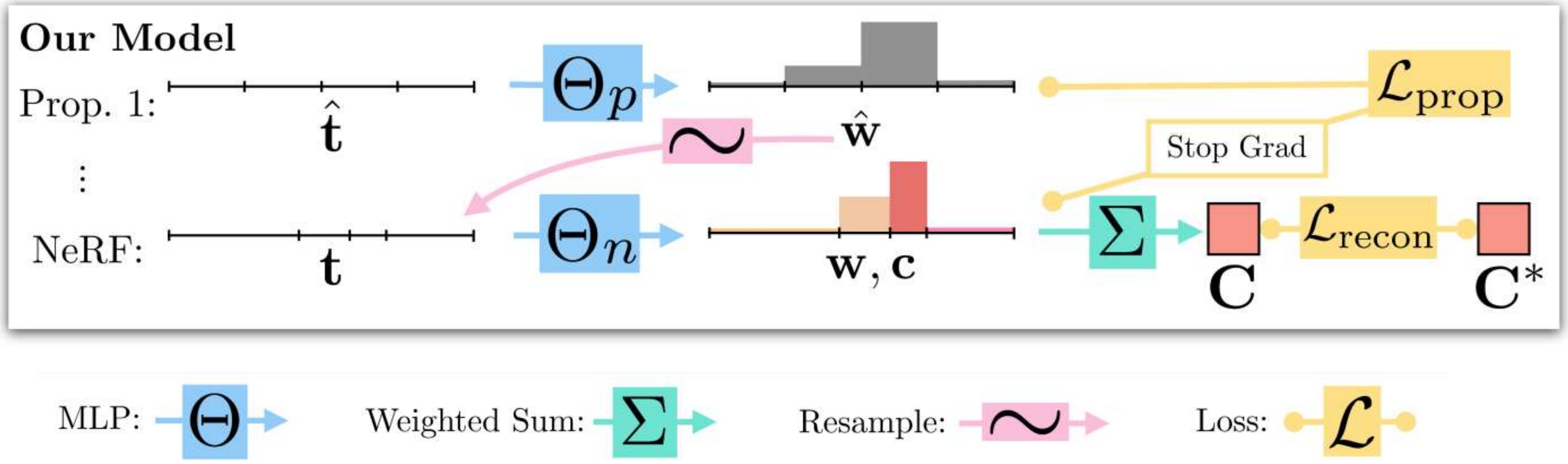
- First, **uniformly sample** the points and calculate **PDF** by its weight
- Second, perform **Importance sampling** based on the PDF of previous samples



mipNeRF360: Online distillation

Efficiency

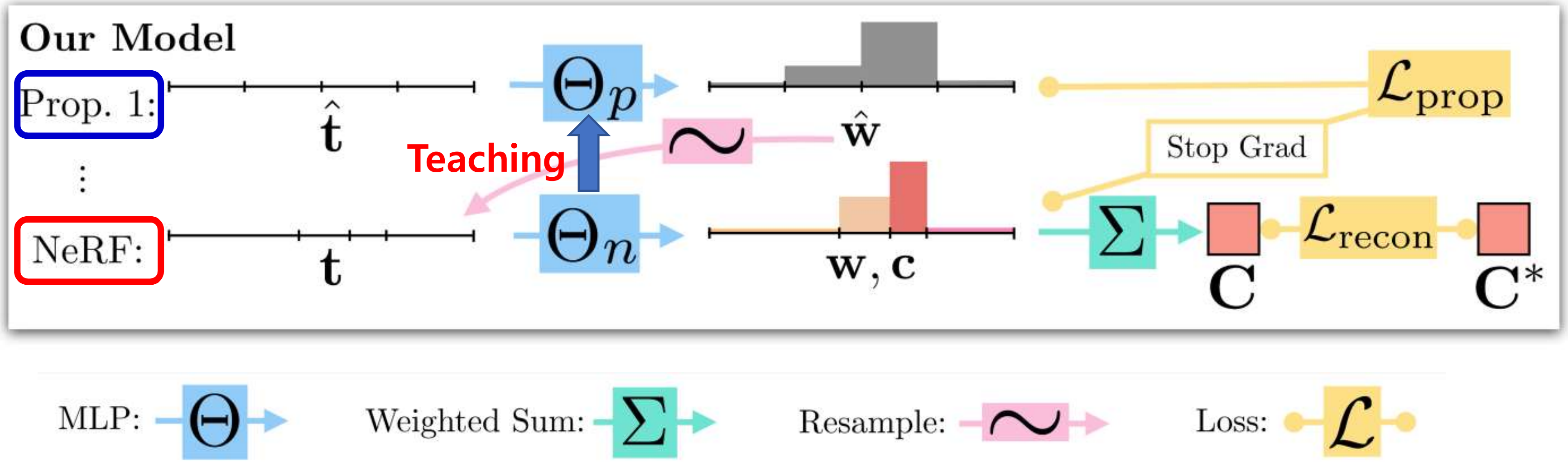
- **Distillation**: train a small model to imitate the huge model
  - Reduce the evaluation of huge model!



mipNeRF360: Online distillation

Efficiency

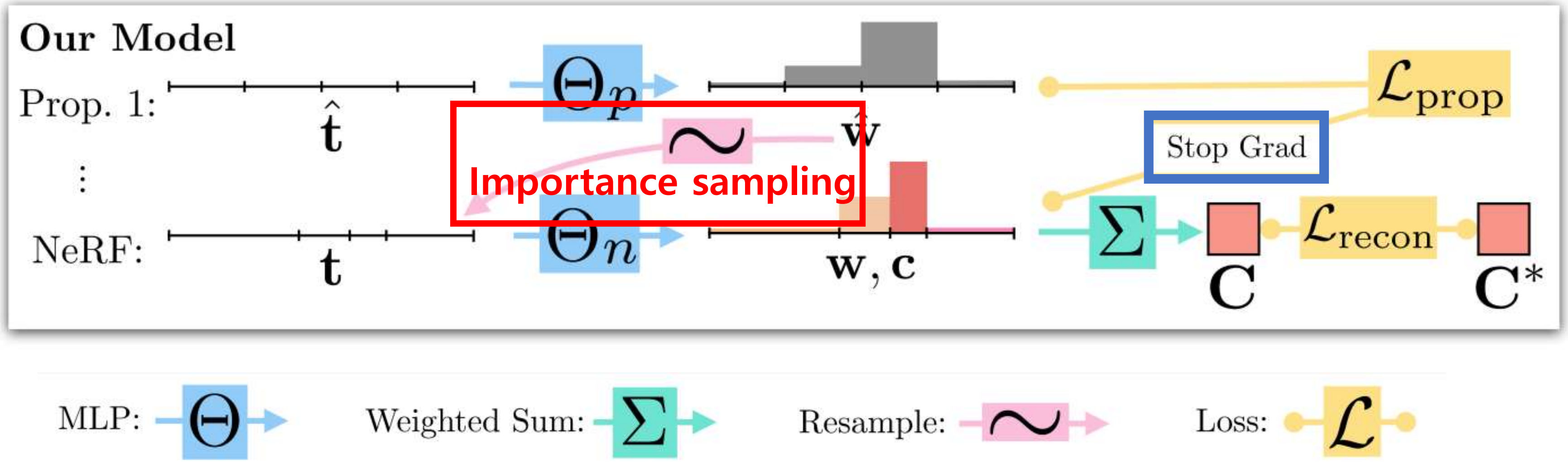
- Separate MLP as **proposal** MLP and **NeRF** MLP **(proposal ⋘ NeRF)**
- **Proposal** MLP does not predict the image directly
  - Only estimate the weights of samples



mipNeRF360: Online distillation

Efficiency

- Using **weights on proposal network** performs **importance sampling**
- Prevent the teaching MLP's update using "**Stop Gradient**" while updating student MLP

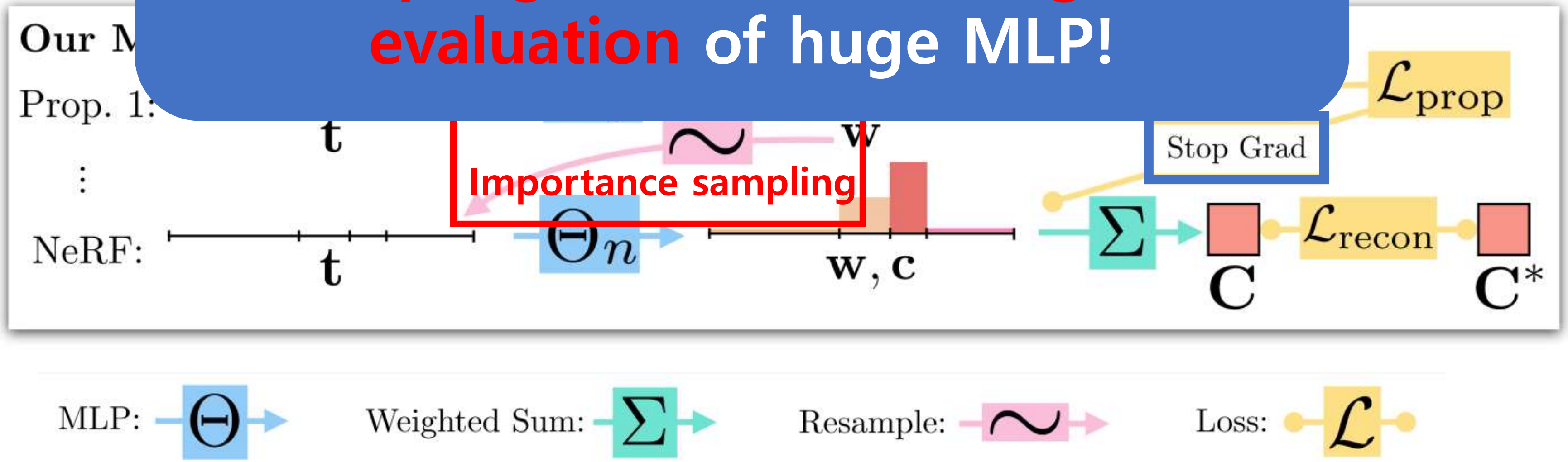


mipNeRF360: Online distillation

Efficiency

- Using **we** … **ampling**
- Prevent t … ile updating student N …

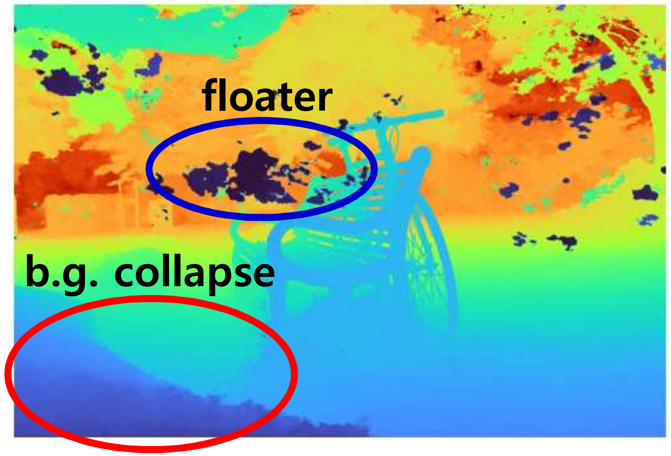
Online distillation enable mipNeRF360 to perform importance sampling while minimizing the evaluation of huge MLP!



mipNeRF: Characteristic Artifacts

Ambiguity

- **mipNeRF** does not have special function to discriminate **the surface or accurate geometry** -> 2 types of artifacts
  1. **Floater:** Floating discontinuity on a scene
  2. **Background Collapse:** predicting background as a set of semi-transparent cloud



Depth Result of **mipNeRF**

mipNeRF360: Regularization

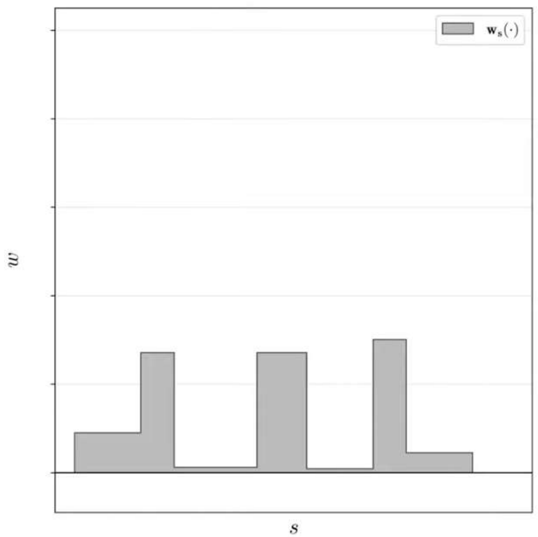
Ambiguity

- **Compress** the samples and its coefficients by adding **new loss**
- Make the weight distribution much sharper like delta function

  -> **Clarify the surface of contents**

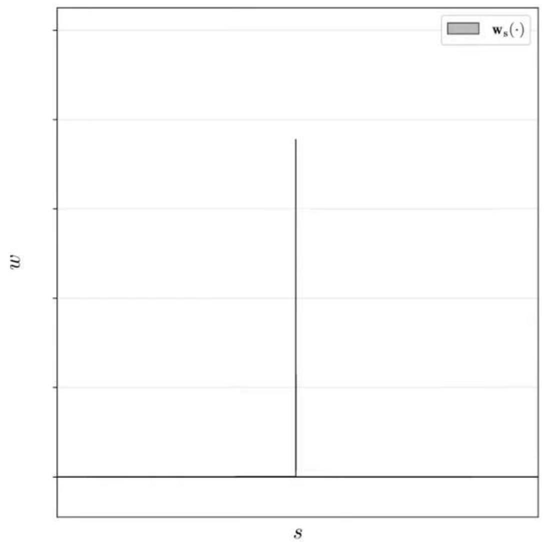
$$\mathcal{L}_{\text{dist}}(\mathbf{s}, \mathbf{w}) = \iint_{-\infty}^{\infty} \mathbf{w}_{\mathbf{s}}(u)\mathbf{w}_{\mathbf{s}}(v)|u - v| \, d_u \, d_v \,,$$

weight



Normalized s-distance

Distribution loss term



Normalized s-distance

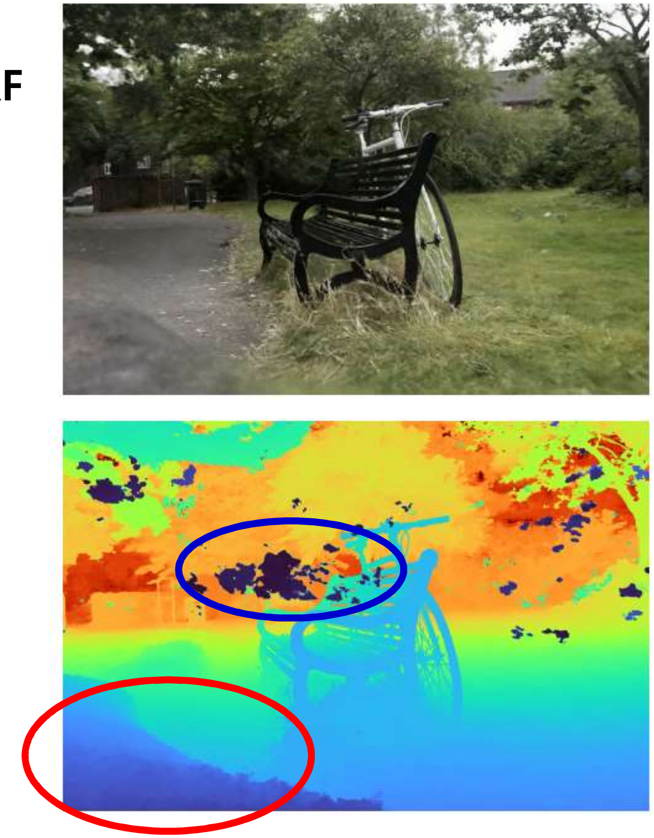
KAIST

mipNeRF360: Regularization

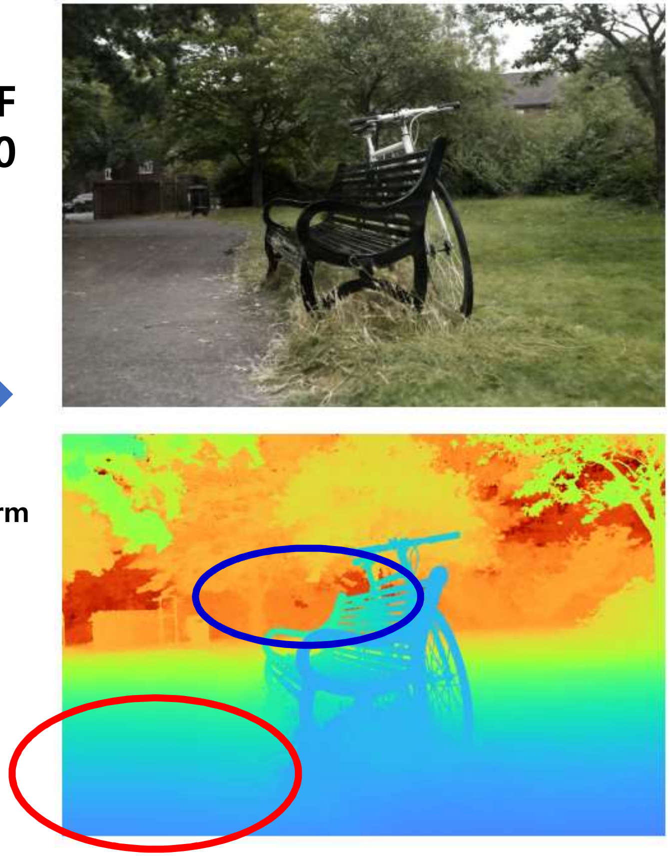
Ambiguity

- Successfully eliminate those artifacts!

mipNeRF



mipNeRF 360

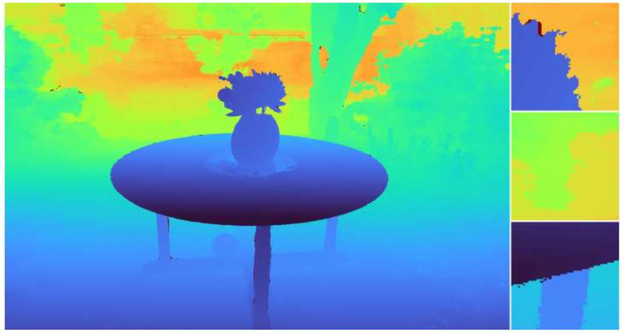


Distribution loss term

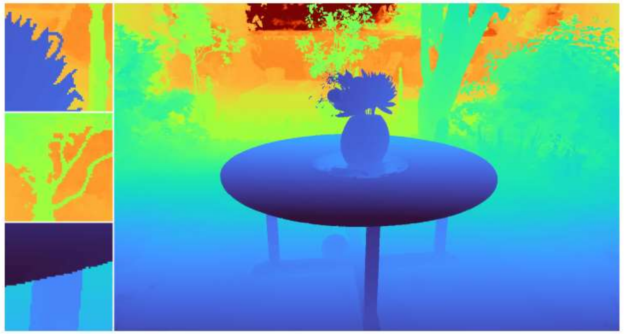
KAIST

mipNeRF360: Result

- Shows outstanding result in 360 scene!



mipNeRF, SSIM: 0.526



mipNeRF360, SSIM: 0.804

PSNR

	Outdoor					Indoor			
	bicycle	*flowers*	*garden*	*stump*	*treehill*	*room*	*counter*	*kitchen*	*bonsai*
NeRF [13, 33]	21.76	19.40	23.11	21.73	21.28	28.56	25.67	26.31	26.81
NeRF w/ DONeRF [34] param.	21.67	19.48	23.29	23.38	21.70	28.28	25.74	25.42	27.32
mip-NeRF [3]	21.69	19.31	23.16	23.10	21.21	28.73	25.59	26.47	27.13
NeRF++ [51]	22.64	20.31	24.32	24.34	22.20	28.87	26.38	27.80	29.15
Deep Blending [17]	21.09	18.13	23.61	24.08	20.80	27.20	26.28	25.02	27.08
Point-Based Neural Rendering [26]	21.64	19.28	22.50	23.90	20.98	26.99	25.23	24.47	28.42
Stable View Synthesis [41]	22.79	20.15	25.99	24.39	21.72	28.93	26.40	28.49	29.07
mip-NeRF [3] w/bigger MLP	22.90	20.79	25.85	23.64	21.71	30.67	28.61	29.95	31.59
NeRF++ [51] w/bigger MLPs	23.75	21.11	25.91	25.48	22.77	30.13	27.79	29.85	30.68
Our Model	24.37	21.73	26.98	26.40	22.87	31.63	29.55	32.23	33.46
Our Model w/GLO	23.95	21.60	25.09	25.98	21.99	28.24	28.40	30.81	30.27

SSIM

	Outdoor					Indoor			
	bicycle	*flowers*	*garden*	*stump*	*treehill*	*room*	*counter*	*kitchen*	*bonsai*
NeRF [13, 33]	0.455	0.376	0.546	0.453	0.459	0.843	0.775	0.749	0.792
NeRF w/ DONeRF [34] param.	0.454	0.379	0.542	0.522	0.461	0.841	0.776	0.678	0.813
mip-NeRF [3]	0.454	0.373	0.543	0.517	0.466	0.851	0.779	0.745	0.818
NeRF++ [51]	0.526	0.453	0.635	0.594	0.530	0.852	0.802	0.816	0.876
Deep Blending [17]	0.466	0.320	0.675	0.634	0.523	0.868	0.856	0.768	0.883
Point-Based Neural Rendering [26]	0.608	0.487	0.735	0.651	0.579	0.887	0.868	0.876	0.919
Stable View Synthesis [41]	0.663	0.541	0.818	0.683	0.606	0.905	0.886	0.910	0.925
mip-NeRF [3] w/bigger MLP	0.612	0.514	0.777	0.643	0.577	0.903	0.877	0.902	0.928
NeRF++ [51] w/bigger MLPs	0.630	0.533	0.761	0.687	0.597	0.883	0.857	0.888	0.913
Our Model	0.685	0.583	0.813	0.744	0.632	0.913	0.894	0.920	0.941
Our Model w/GLO	0.687	0.582	0.800	0.745	0.619	0.907	0.890	0.916	0.932

mipNeRF360: Result

Summary of mipNeRF360

- Use **contraction** and **warping** to **normalize** the sample distance
- Use **online distillation** for **efficient** training
- Use **regularization** for solving **ambiguity** in unbounded scene

Total Loss :

$$\mathcal{L}_{\text{recon}}(\mathbf{C}(\mathbf{t}), \mathbf{C}^*) + \lambda \mathcal{L}_{\text{dist}}(\mathbf{s}, \mathbf{w}) + \sum_{k=0}^{1} \mathcal{L}_{\text{prop}}(\mathbf{s}, \mathbf{w}, \hat{\mathbf{s}}^k, \hat{\mathbf{w}}^k)$$

Ordinary image recon loss

Regularization for ambiguity

Loss for online distillation

KAIST

Thank you!

Quiz

1. IPE in mipNeRF can improve the performance without increasing the number of samples if they are sampled by importance sampling (True / False)

2. Why mipNeRF360 uses regularization in 360° scene? ( )
   ① For parameterization
   ② For importance sampling
   ③ For efficiency
   ④ For resolving the ambiguity

Reference

1. *Jonathan T. Barron, et al. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields*
2. *Jonathan T. Barron, et al. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields*
3. *https://jonbarron.info/mipnerf/*
4. *https://jonbarron.info/mipnerf360*
5. *https://www.youtube.com/watch?v=zBSH-k9GbV4*

CS580:
mipNeRF & mipNeRF360
Appendix

Integrated Positional Encoding (IPE)

Positional Encoding

$$\mathrm{P} = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 & 0 & & 2^{L-1} & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 & \cdots & 0 & 2^{L-1} & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & & 0 & 0 & 2^{L-1} \end{bmatrix}^T, \quad \gamma(\mathrm{x}) = \begin{bmatrix} \sin(\mathrm{Px}) \\ \cos(\mathrm{Px}) \end{bmatrix}$$

Expectation

Positional encoding

$$E_{\mathrm{x} \sim N(\mu, \sigma^2)}[\gamma(\mathrm{x})]$$

Normal distribution

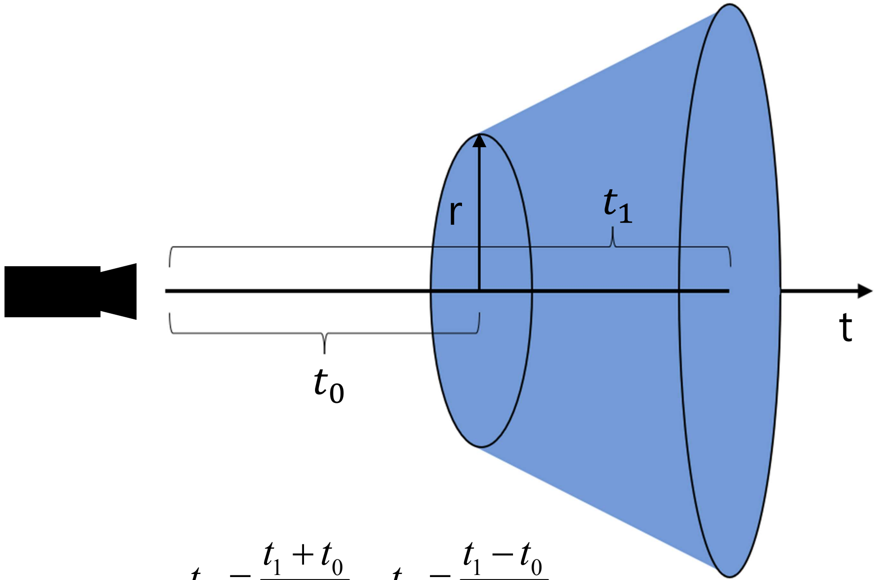
~

Conical frustum equation

$$F(x, o, d, \dot{r}, t_0, t_1) = \left\{ \left(t_0 < \frac{d^T(x-o)}{\|d\|_2^2} < t_1 \right) \wedge \left(\frac{d^T(x-o)}{\|d\|_2 \|x-o\|_2} > \frac{1}{\sqrt{1 + (\dot{r} / \|d\|_2)^2}} \right) \right\}$$

x are the member of conical frustum

Integrated Positional Encoding (IPE) (cont.)



$$t_\mu = \frac{t_1 + t_0}{2}, \quad t_\delta = \frac{t_1 - t_0}{2}$$

Mean, Covariance matrix of conical frustum

$$\mu_t = t_\mu + \frac{2t_\mu t_\delta^2}{3t_\mu^2 + t_\delta^2}, \quad \sigma_t^2 = \frac{t_\delta^2}{3} - \frac{4t_\delta^2(12t_\mu^2 - t_\delta^2)}{15(t_\mu^2 + t_\delta^2)^2},$$

$$\sigma_r^2 = \dot{r}^2\left(\frac{t_\mu^2}{4} + \frac{5t_\delta^2}{12} - \frac{4t_\delta^4}{15(3t_\mu^2 + t_\delta^2)}\right)$$

cone dir

radius dir

$$\mu = o + \mu_t d, \quad \Sigma = \sigma_t(dd^{\mathrm{T}}) + \sigma_r\left(I - \frac{dd^{\mathrm{T}}}{\|d\|_2^2}\right)$$

Proof of IPE

PDF

$$\varphi(r,t,\theta) = (rt\ cos\theta, rt\ sin\theta, t)$$

for $\theta \in [0,\ 2\pi),\ t \geq 0,\ |r| \leq \dot{r}$

$$dxdydz = |\det(D_\varphi)(r,t,\theta)|\, drdtd\theta$$

$$= \begin{vmatrix} t\cos\theta & t\sin\theta & 0 \\ r\cos\theta & r\sin\theta & 1 \\ -rt\sin\theta & rt\cos\theta & 0 \end{vmatrix} drdtd\theta$$

$$= (rt^2\cos^2\theta + rt^2\sin\theta)drdtd\theta$$

$$= rt^2 drdtd\theta$$

$$V = \int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{\dot{r}} rt^2 drdtd\theta = \pi\dot{r}^2\frac{t_1^3 - t_0^3}{3}.$$

$$P = \frac{1}{V}.$$

Var of w.r.t. "t"

$$E[t^2] = \frac{1}{V}\int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{\dot{r}} t^2 \cdot rt^2\, dr\, dt\, d\theta$$

$$= \frac{1}{V}\int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{\dot{r}} rt^4\, dr\, dt\, d\theta$$

$$= \frac{1}{V}\cdot\pi\dot{r}^2\frac{t_1^5 - t_0^5}{5}$$

$$= \frac{3(t_1^5 - t_0^5)}{5(t_1^3 - t_0^3)}.$$

$$E[t] = \frac{1}{V}\int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{\dot{r}} t \cdot rt^2\, dr\, dt\, d\theta$$

$$= \frac{1}{V}\int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{\dot{r}} rt^3\, dr\, dt\, d\theta$$

$$= \frac{1}{V}\cdot\pi\dot{r}^2\frac{t_1^4 - t_0^4}{4}$$

$$= \frac{3(t_1^4 - t_0^4)}{4(t_1^3 - t_0^3)}.$$

$$Var(t) = \sigma_t^2 = E[t^2] - (E[t])^2$$

$$= \frac{3(t_1^5 - t_0^5)}{5(t_1^3 - t_0^3)} - \mu_t^2$$

Var of w.r.t. "x"(radius)

$$E[x^2] = \frac{1}{V}\int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{\dot{r}} (rt\cos\theta)^2 \cdot rt^2\, dr\, dt\, d\theta$$

$$= \frac{1}{V}\int_{t_0}^{t_1}\int_0^{\dot{r}} r^3t^4 \int_0^{2\pi}\cos^2\theta\, d\theta\, dr\, dt$$

$$= \frac{1}{V}\cdot\frac{\dot{r}^4}{4}\cdot\frac{t_1^5 - t_0^5}{5}\cdot\pi$$

$$= \frac{\dot{r}^2}{4}\cdot\frac{3(t_1^5 - t_0^5)}{5(t_1^3 - t_0^3)}.$$

$$E[x] = \frac{1}{V}\int_0^{2\pi}\int_{t_0}^{t_1}\int_0^{r}(rt\cos\theta)\cdot rt\ drdtd\theta$$

$$= 0.$$

$$Var(x) = \sigma_r^2 = E[x^2] - (E[x])^2$$

$$= \dot{r}^2\left(\frac{3(t_1^5 - t_0^5)}{20(t_1^3 - t_0^3)}\right).$$

KAIST

Proof of IPE (conti.)

Positional encoding

$$P = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 & 0 & & 2^{L-1} & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 & \cdots & 0 & 2^{L-1} & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 & & 0 & 0 & 2^{L-1} \end{bmatrix}^T, \quad \gamma(x) = \begin{bmatrix} \sin(Px) \\ \cos(Px) \end{bmatrix}$$

$$x \rightarrow Px$$

$Cov[Ax,By] = A\, Cov[x,y]B^T$

$$\mu_\gamma = P\mu, \quad \Sigma_\gamma = P\Sigma P^T$$

Applying PE to x

Expectation of sin, cos following gaussian dist.

$$E_{x\sim N(\mu,\sigma^2)}[\sin(x)] = \sin(\mu)\exp(-(1/2)\sigma^2),$$
$$E_{x\sim N(\mu,\sigma^2)}[\cos(x)] = \cos(\mu)\exp(-(1/2)\sigma^2)$$

$$\gamma(\mu,\Sigma) = E_{x\sim N(\mu_\gamma,\Sigma_\gamma)}[\gamma(x)]$$
$$= \begin{bmatrix} \sin(\mu_\gamma)\circ\exp(-(1/2)diag(\Sigma_\gamma)), \\ \cos(\mu_\gamma)\circ\exp(-(1/2)diag(\Sigma_\gamma)), \end{bmatrix}$$

$$diag(\Sigma_\gamma) = \left[diag(\Sigma), 4diag(\Sigma), \cdots, 4^{L-1}diag(\Sigma)\right]^T$$
$$diag(\Sigma) = \sigma_t(d\circ d) + \sigma_r\left(I - \frac{d\circ d}{\|d\|_2^2}\right)$$

Proof of expectation of sin, cos following normal dist.

$$e^{ix} = \cos(x) + i\sin(x)$$ ········ **Euler formula**

$$E[e^{ix}] = E[\cos(x)] + iE[\sin(x)]$$

$$E_{x\sim \mathrm{N}(\mu,\sigma^2)}[e^{ix}] = \int_{-\infty}^{\infty} e^{ix} \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x-\mu)^2}{2\sigma^2}} dx$$

$$= \varphi_{x\sim N(\mu,\sigma^2)}(x)(t=1)$$ ········ **Characteristic function of normal dist.**

$$= e^{i\mu - \frac{1}{2}\sigma^2} \quad (\because e^{it\mu - \frac{1}{2}\sigma^2 t^2})$$

$$= \{\cos(\mu) + i\sin(\mu)\} e^{-\frac{1}{2}\sigma^2}$$

$$\therefore E[\cos(x)] = \cos(\mu) e^{-\frac{1}{2}\sigma^2}$$

$$E[\sin(x)] = \sin(\mu) e^{-\frac{1}{2}\sigma^2}$$

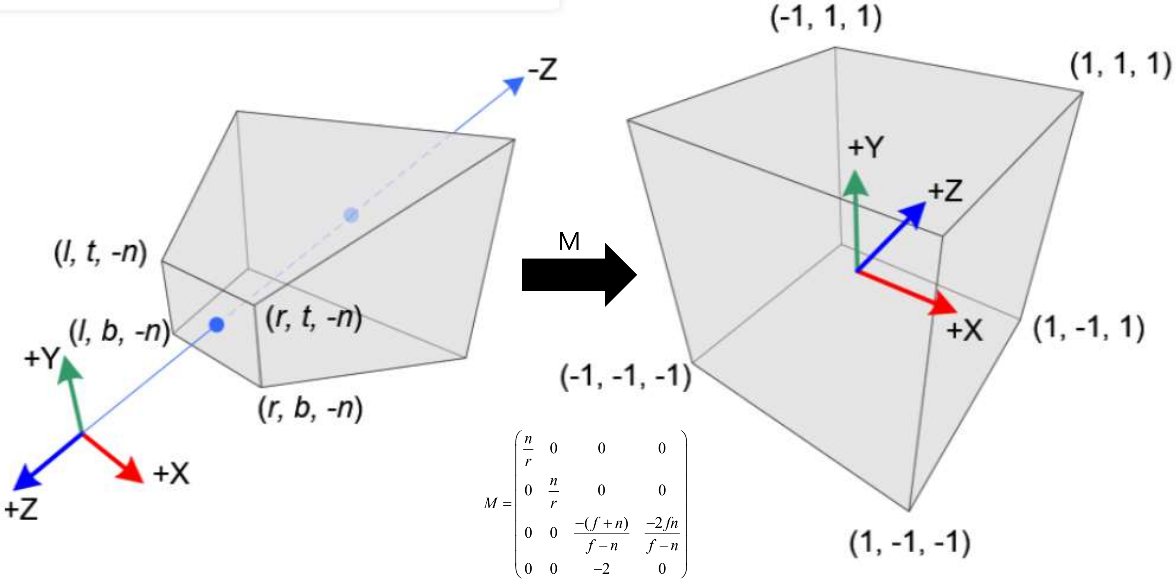
$$\therefore \quad E_{x\sim \mathrm{N}(\mu,\sigma^2)}[\sin(x)] = \sin(\mu)\exp(-(1/2)\sigma^2),$$

$$E_{x\sim \mathrm{N}(\mu,\sigma^2)}[\cos(x)] = \cos(\mu)\exp(-(1/2)\sigma^2)$$

NeRF in Unbounded scene

- ◆ 기존 NeRF dataset에서는 “facing forward” 라는 unbounded dataset이 존재
- ◆ 이를 효과적으로 렌더링하기 위해 ray의 원점과 방향을 계산한 후, Normalized device coordinate (NDC) 변환을 거친 다음 PE를 진행

NeRF applied to
“**facing forward scene**”



$$M = \begin{pmatrix} \frac{n}{r} & 0 & 0 & 0 \\ 0 & \frac{n}{r} & 0 & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -2 & 0 \end{pmatrix}$$

Normalized Device Coordinate (**NDC**)

NDC in NeRF $(o+t\cdot d \;\rightarrow\; o'+t'\cdot d')$

r = right, t= top,
n = near, f = far

$$\begin{pmatrix} \frac{n}{r} & 0 & 0 & 0 \\ 0 & \frac{n}{t} & 0 & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{n}{r}x \\ \frac{n}{t}y \\ \frac{-(f+n)}{f-n}z - \frac{-2fn}{f-n} \\ -z \end{pmatrix}$$

NDC projection matrix

$$\text{project} \rightarrow \begin{pmatrix} \frac{n}{r}\frac{x}{-z} \\ \frac{n}{t}\frac{y}{-z} \\ \frac{(f+n)}{f-n} - \frac{2fn}{f-n}\frac{1}{-z} \end{pmatrix}$$

$a_x = -\frac{n}{r},\ a_y = -\frac{n}{t},$
$a_z = -\frac{f+n}{f-n},\ b_z = -\frac{2fn}{f-n}$
$(x, y, z) = o + t\cdot d$

$$\begin{pmatrix} a_x \frac{o_x+td_x}{o_z+td_z} \\ a_y \frac{o_y+td_y}{o_z+td_z} \\ a_z + \frac{b_z}{o_z+td_z} \end{pmatrix} = \begin{pmatrix} o'_x + t'd'_x \\ o'_y + t'd'_y \\ o'_z + t'd'_z \end{pmatrix}$$

Suppose if t=0, then t'=0

$$\mathbf{o}' = \begin{pmatrix} o'_x \\ o'_y \\ o'_z \end{pmatrix} = \begin{pmatrix} a_x \frac{o_x}{o_z} \\ a_y \frac{o_y}{o_z} \\ a_z + \frac{b_z}{o_z} \end{pmatrix} = \pi(\mathbf{o})$$

$$\begin{pmatrix} t'd'_x \\ t'd'_y \\ t'd'_z \end{pmatrix} = \begin{pmatrix} a_x \frac{o_x+td_x}{o_z+td_z} - a_x \frac{o_x}{o_z} \\ a_y \frac{o_y+td_y}{o_z+td_z} - a_y \frac{o_y}{o_z} \\ a_z + \frac{b_z}{o_z+td_z} - a_z - \frac{b_z}{o_z} \end{pmatrix} = \begin{pmatrix} a_x \frac{o_z(o_x+td_x)-o_x(o_z+td_z)}{(o_z+td_z)o_z} \\ a_y \frac{o_z(o_y+td_y)-o_y(o_z+td_z)}{(o_z+td_z)o_z} \\ b_z \frac{o_z-(o_z+td_z)}{(o_z+td_z)o_z} \end{pmatrix}$$

$$= \begin{pmatrix} a_x \frac{td_z}{o_z+td_z}\left(\frac{d_x}{d_z} - \frac{o_x}{o_z}\right) \\ a_y \frac{td_z}{o_z+td_z}\left(\frac{d_y}{d_z} - \frac{o_y}{o_z}\right) \\ -b_z \frac{td_z}{o_z+td_z}\frac{1}{o_z} \end{pmatrix}$$

$$t' = \frac{td_z}{o_z+td_z} = 1 - \frac{o_z}{o_z+td_z}$$

$$\mathbf{d}' = \begin{pmatrix} a_x\left(\frac{d_x}{d_z} - \frac{o_x}{o_z}\right) \\ a_y\left(\frac{d_y}{d_z} - \frac{o_y}{o_z}\right) \\ -b_z \frac{1}{o_z} \end{pmatrix}.$$

https://blog.naver.com/PostView.naver?blogId=wnxodnr&logNo=10122064855&parentCategoryNo=26&categoryNo=&viewDate=&isShowPopularPosts=true&from=search NDC matrix proof

NDC in NeRF (cont.)

r = right, t= top, n = near, f = far

$$a_x = -\frac{n}{r},\ a_y = -\frac{n}{t},\ a_z = -\frac{f+n}{f-n},\ b_z = -\frac{2fn}{f-n}$$

Supposing pinhole camera

$f \to \infty$

$$a_x = -\frac{f_{cam}}{W/2},\ a_y = -\frac{f_{cam}}{H/2} \qquad a_z = 1,\ b_z = 2n$$

$$\mathbf{o}' = \begin{pmatrix} o'_x \\ o'_y \\ o'_z \end{pmatrix} = \begin{pmatrix} a_x \frac{o_x}{o_z} \\ a_y \frac{o_y}{o_z} \\ a_z + \frac{b_z}{o_z} \end{pmatrix} \qquad t' = \frac{td_z}{o_z + td_z} = 1 - \frac{o_z}{o_z + td_z}$$

$$\mathbf{d}' = \begin{pmatrix} a_x \left(\frac{d_x}{d_z} - \frac{o_x}{o_z} \right) \\ a_y \left(\frac{d_y}{d_z} - \frac{o_y}{o_z} \right) \\ -b_z \frac{1}{o_z} \end{pmatrix}.$$

$$\mathbf{o}' = \begin{pmatrix} -\frac{f_{cam}}{W/2} \frac{o_x}{o_z} \\ -\frac{f_{cam}}{H/2} \frac{o_y}{o_z} \\ 1 + \frac{2n}{o_z} \end{pmatrix} \qquad t' = \frac{td_z}{o_z + td_z} = 1 - \frac{o_z}{o_z + td_z}$$

$$\mathbf{d}' = \begin{pmatrix} -\frac{f_{cam}}{W/2} \left(\frac{d_x}{d_z} - \frac{o_x}{o_z} \right) \\ -\frac{f_{cam}}{H/2} \left(\frac{d_y}{d_z} - \frac{o_y}{o_z} \right) \\ -2n \frac{1}{o_z} \end{pmatrix}$$

NDC Projection $o + t \cdot d \ \to \ o' + t' \cdot d'$ ($t \sim (0, \infty) \to t' \sim (0,1)$)

https://blog.naver.com/PostView.naver?blogId=wnxodnr&logNo=10122064855&parentCategoryNo=26&categoryNo=&viewDate=&isShowPopularPosts=true&from=search NDC matrix proof