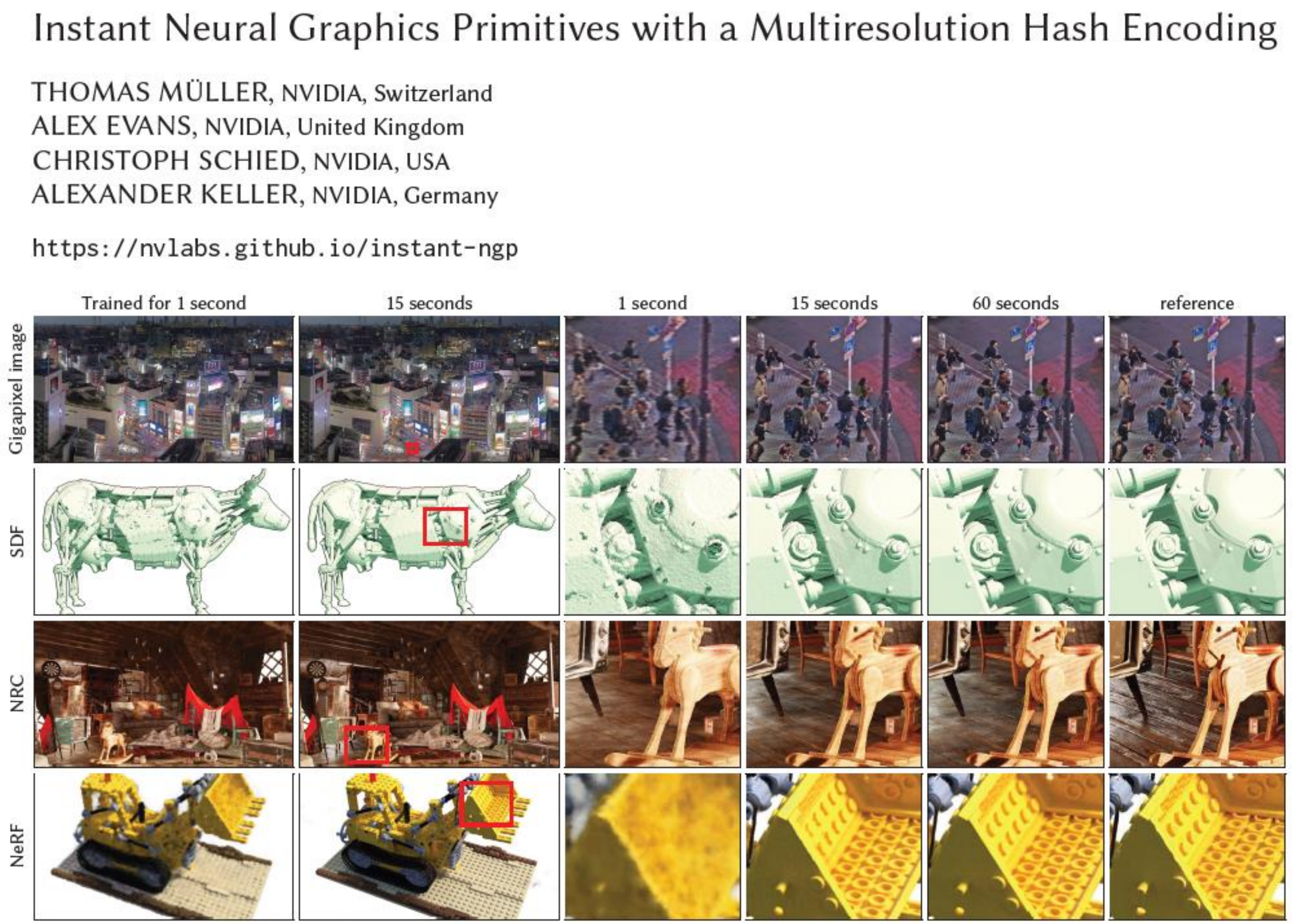
NVIDIA Instant NeRF

Using neural radiance fields to generate
new views of 2D

Instant NGP

- Instant Neural Graphics Primitives with a Multiresolution Hash Encoding
  - 2022, SIGGRAPH
    - Best paper award

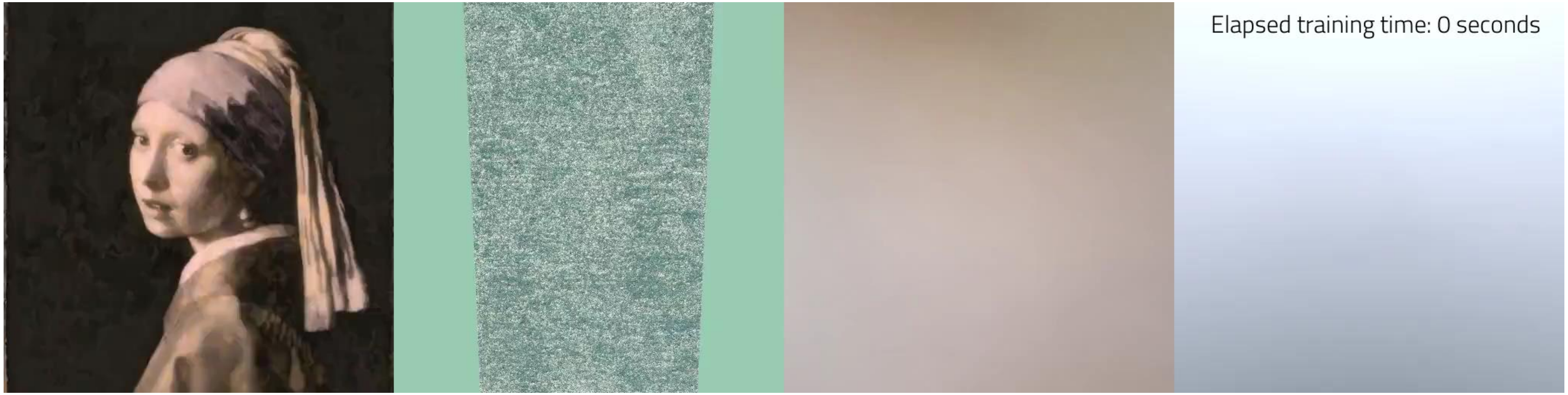

Instant Neural Graphics Primitives with a Multiresolution Hash Encoding

THOMAS MÜLLER, NVIDIA, Switzerland
ALEX EVANS, NVIDIA, United Kingdom
CHRISTOPH SCHIED, NVIDIA, USA
ALEXANDER KELLER, NVIDIA, Germany

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

Instant NGP

- Neural Graphics Primitives의 4가지 Task
  - Gigapixel image
  - SDF (Signed Distance Function)
  - **NeRF (Neural Radiance Field)**
  - NRC (Neural Radiance Caching)



Gigapixel image **SDF** **NeRF** **NRC**

NeRF

- NeRF : Representing Scenes as Neural Radiance Fields for View Synthesis
  - ECCV 2020 Oral, Best Paper Honorable Mention

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

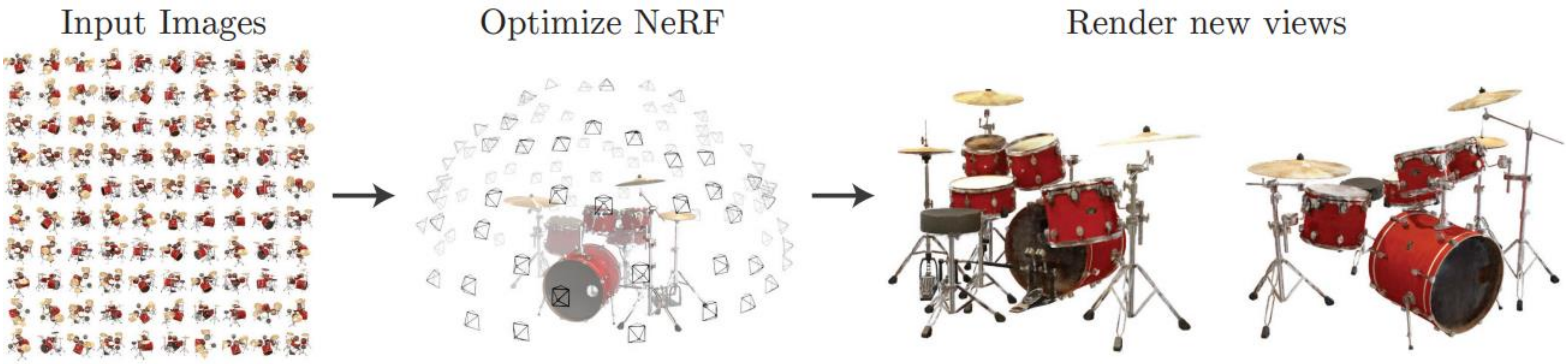
Ben Mildenhall[1*] Pratul P. Srinivasan[1*] Matthew Tancik[1*]
Jonathan T. Barron[2] Ravi Ramamoorthi[3] Ren Ng[1]

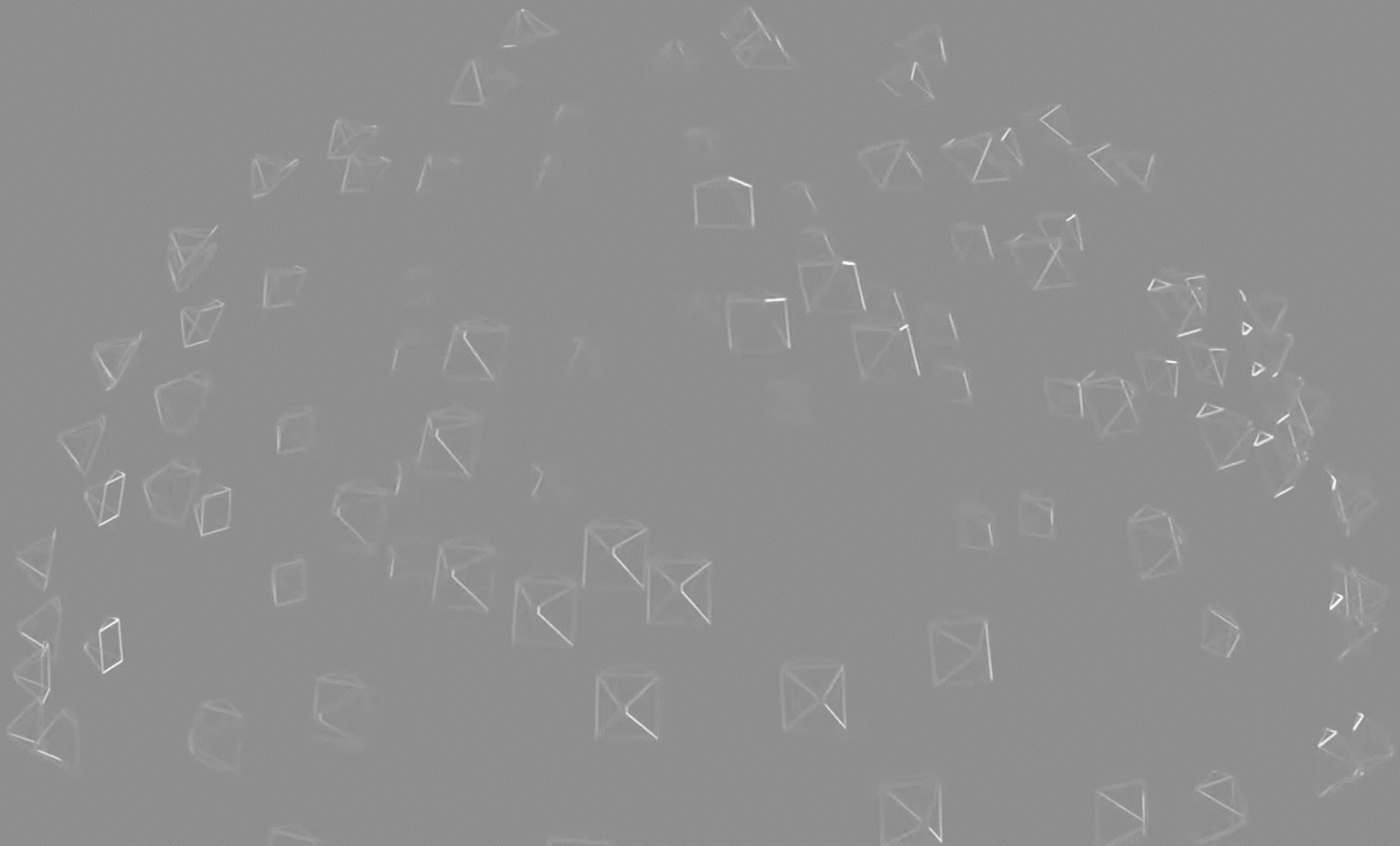
[1]UC Berkeley [2]Google Research [3]UC San Diego

Abstract. We present a method that achieves state-of-the-art results for synthesizing novel views of complex scenes by optimizing an underlying continuous volumetric scene function using a sparse set of input views. Our algorithm represents a scene using a fully-connected (non-convolutional) deep network, whose input is a single continuous 5D coordinate (spatial location (x, y, z) and viewing direction (θ, ϕ)) and whose

NeRF

- 목표
  - 1. 다양한 위치와 방향에서 사진 촬영.
  - 2. Structure from Motion을 통해 카메라 파라미터 추출.
  - 3. 학습 및 렌더링을 통해 새로운 시점에서의 장면을 생성.

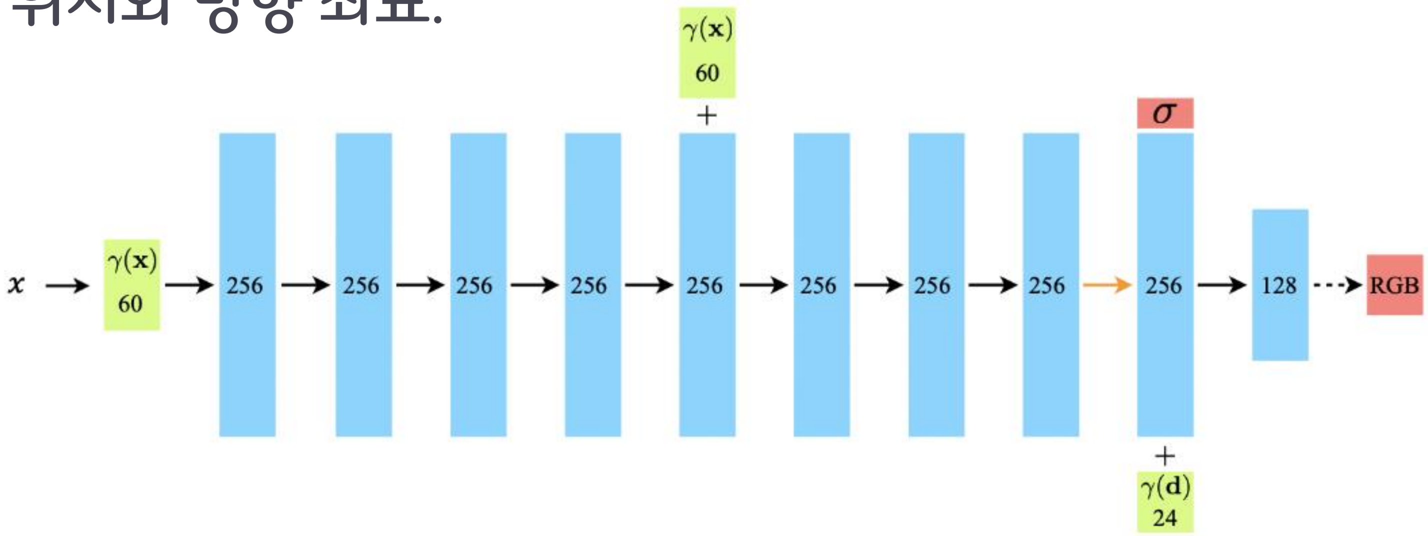




네트워크 구조

- 9개의 Fully Connected Layer
  - ReLU
- 256차원으로 구성된 Feature
- Green box
  - **Positional encoding**된 위치와 방향 좌표.
- Red box
  - Color와 Density를 출력.

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

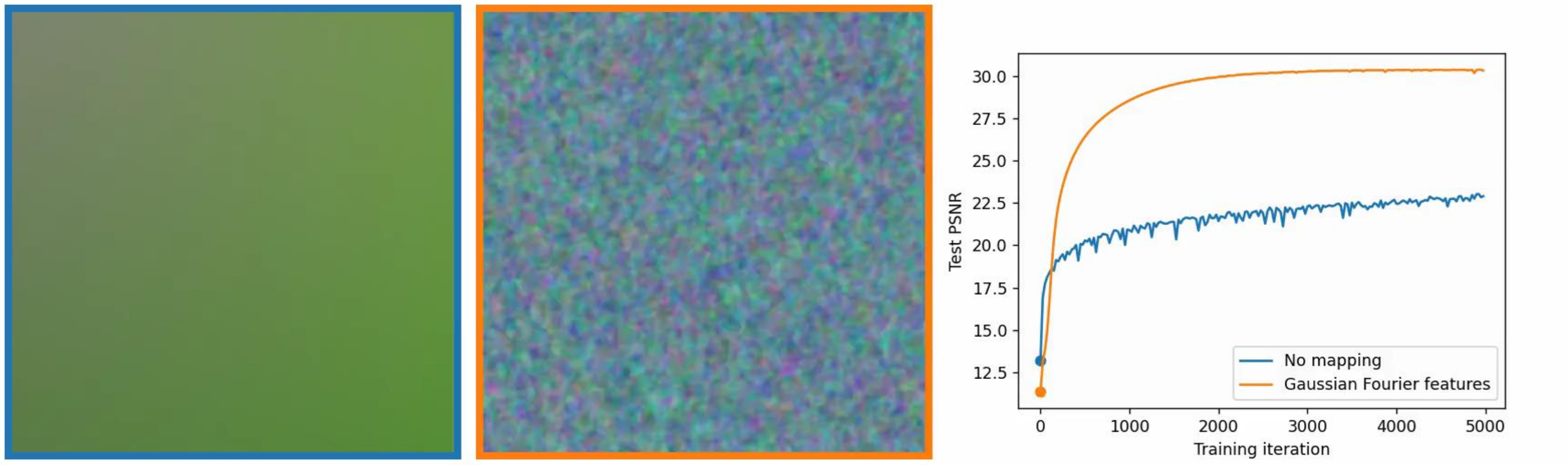


Positional encoding 이 하는 일.

No Positional encoding

Positional encoding

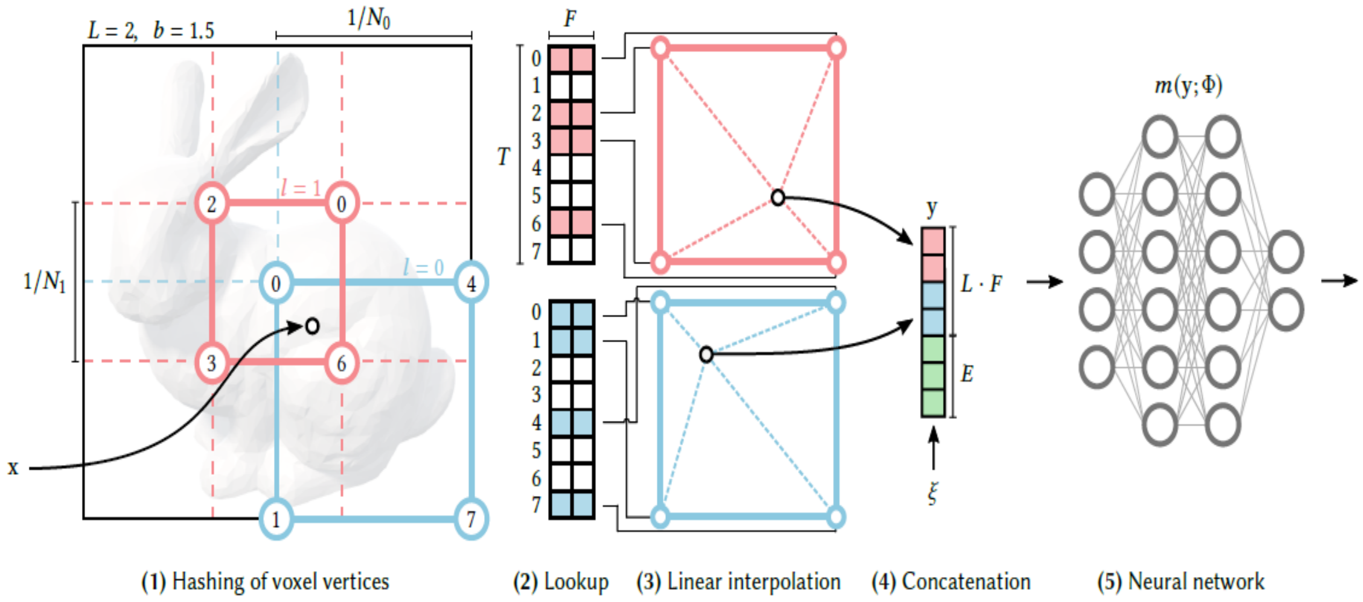
$$\begin{pmatrix} \sin(\mathbf{x}), \cos(\mathbf{x}) \\ \sin(2\mathbf{x}), \cos(2\mathbf{x}) \\ \sin(4\mathbf{x}), \cos(4\mathbf{x}) \\ \vdots \\ \sin(2^N\mathbf{x}), \cos(2^N\mathbf{x}) \end{pmatrix}$$



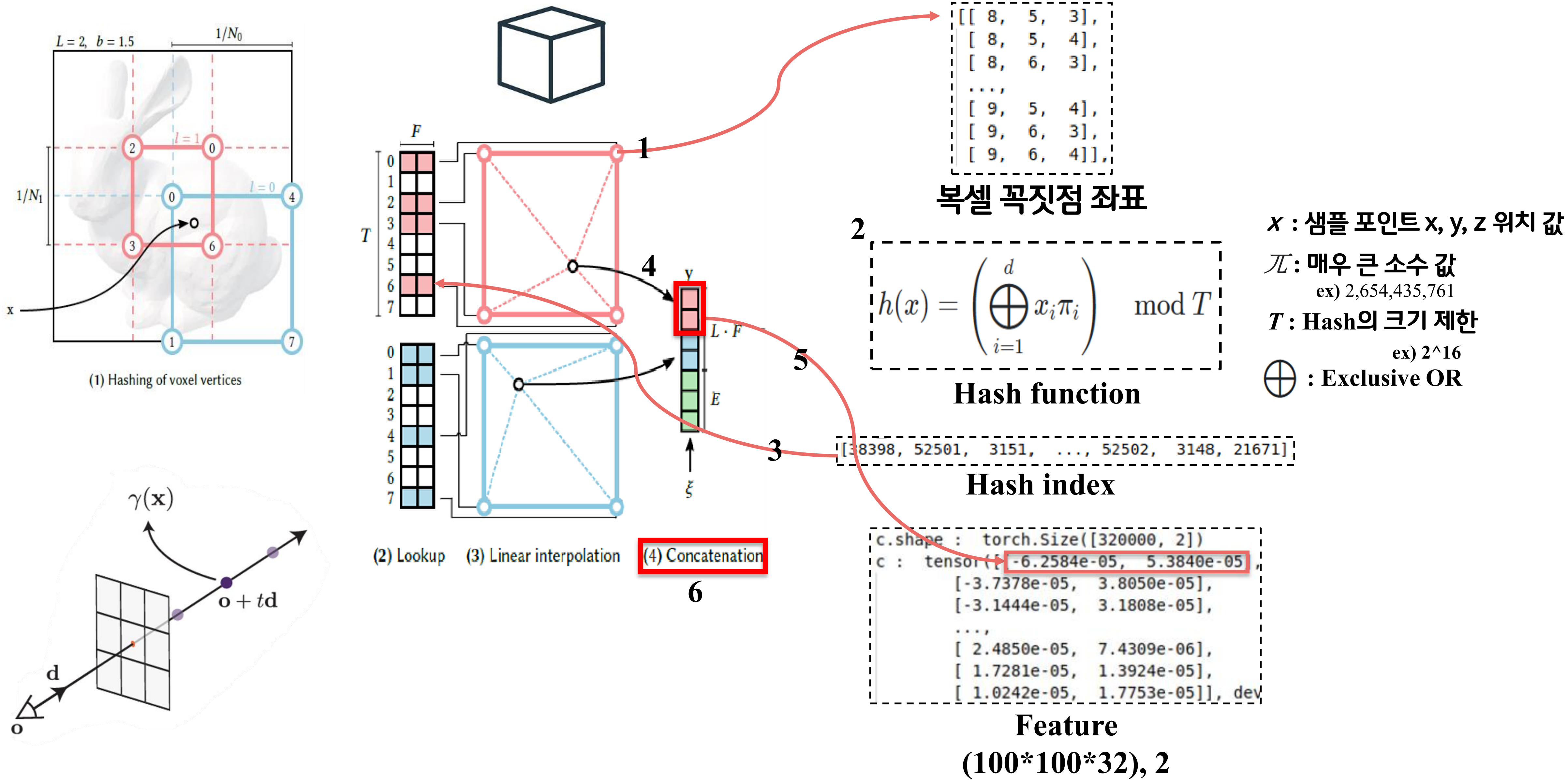
No Positional encoding **Positional encoding**

Main contribution in Instant NeRF

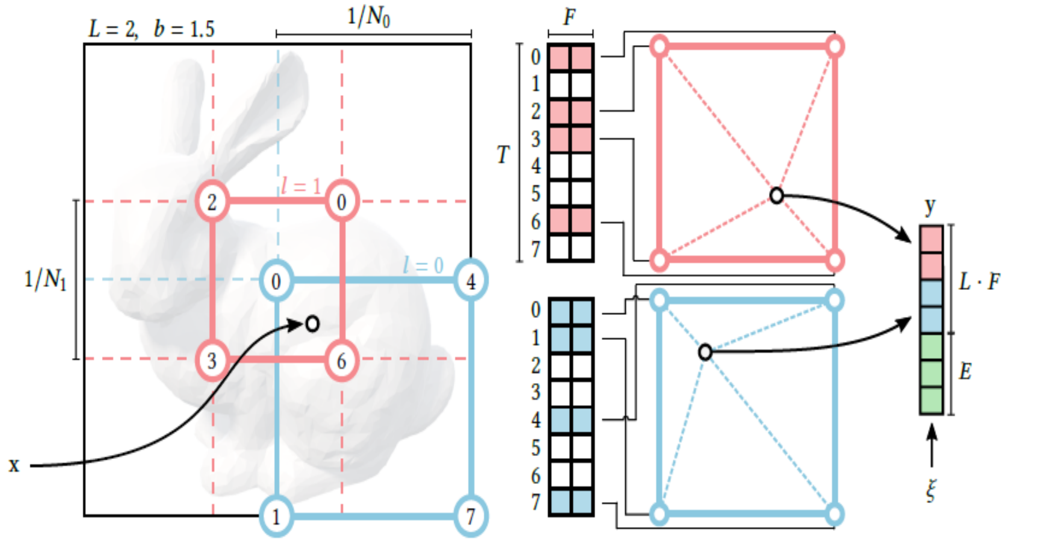
- **Positional encoding 대신에 Multi-resolution Hash Encoding 적용**
  - 보다 빠르게 NeRF를 학습하고 해상도를 높임.
    - **추가로, CUDA, C++ 기반의 cuda-tiny-nn library와 Simple MLP를 통해 더 빠른 속도 가능.**



Multi-resolution Hash Encoding



$L = 2,\ b = 1.5$
$1/N_0$
$1/N_1$
$l = 1$
$l = 0$
x
(1) Hashing of voxel vertices
$\gamma(\mathbf{x})$
$\mathbf{o} + t\mathbf{d}$
d
o
F
T
$L \cdot F$
E
y
ξ
(2) Lookup
(3) Linear interpolation
(4) Concatenation
1
2
3
4
5
6
[[8, 5, 3],
[8, 5, 4],
[8, 6, 3],
...,
[9, 5, 4],
[9, 6, 3],
[9, 6, 4]],
복셀 꼭짓점 좌표
$h(\mathbf{x}) = \left(\bigoplus_{i=1}^{d} x_i \pi_i \right) \mod T$
Hash function
x : 샘플 포인트 x, y, z 위치 값
π : 매우 큰 소수 값
ex) 2,654,435,761
T : Hash의 크기 제한
ex) 2^16
$\bigoplus$: Exclusive OR
[38398, 52501, 3151, ..., 52502, 3148, 21671]
Hash index
c.shape : torch.Size([320000, 2])
c : tensor([[-6.2584e-05, 5.3840e-05],
[-3.7378e-05, 3.8050e-05],
[-3.1444e-05, 3.1808e-05],
...,
[2.4850e-05, 7.4309e-06],
[1.7281e-05, 1.3924e-05],
[1.0242e-05, 1.7753e-05]], dev
Feature
(100*100*32), 2



(1) Hashing of voxel vertices (2) Lookup (3) Linear interpolation (4) Concatenation

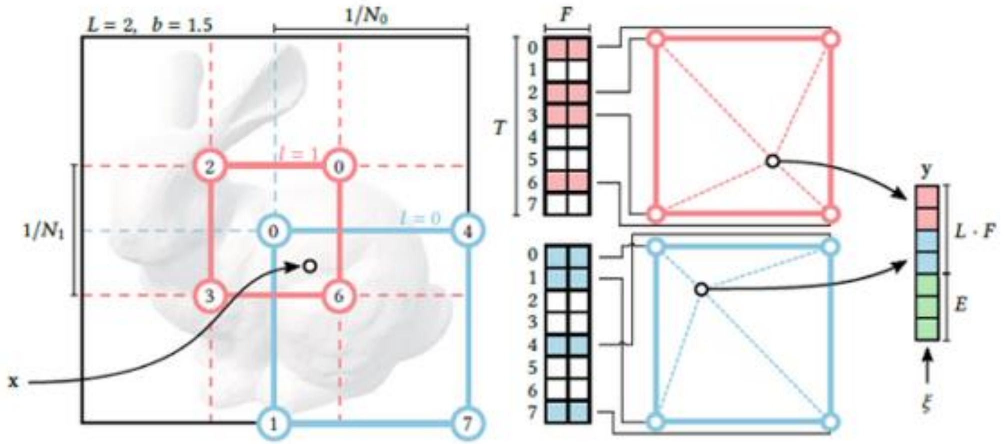
Table 1. Hash encoding parameters and their ranges in our results. Only the hash table size T and max. resolution $N_{\max}$ need to be tuned to the task.

Parameter	Symbol	Value
Number of levels	L	16
Max. entries per level (hash table size)	T	2^{14} to 2^{24}
Number of feature dimensions per entry	F	2
Coarsest resolution	$N_{\min}$	16
Finest resolution	$N_{\max}$	512 to 524288

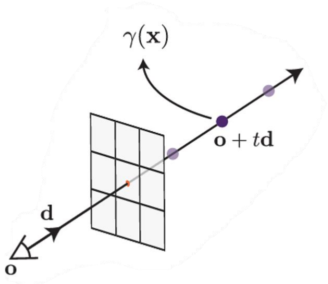
$$L := \text{Number of Levels}$$
$$N_{\min} := \text{Coarsest Resolution}$$
$$N_{\max} := \text{Finest Resolution}$$

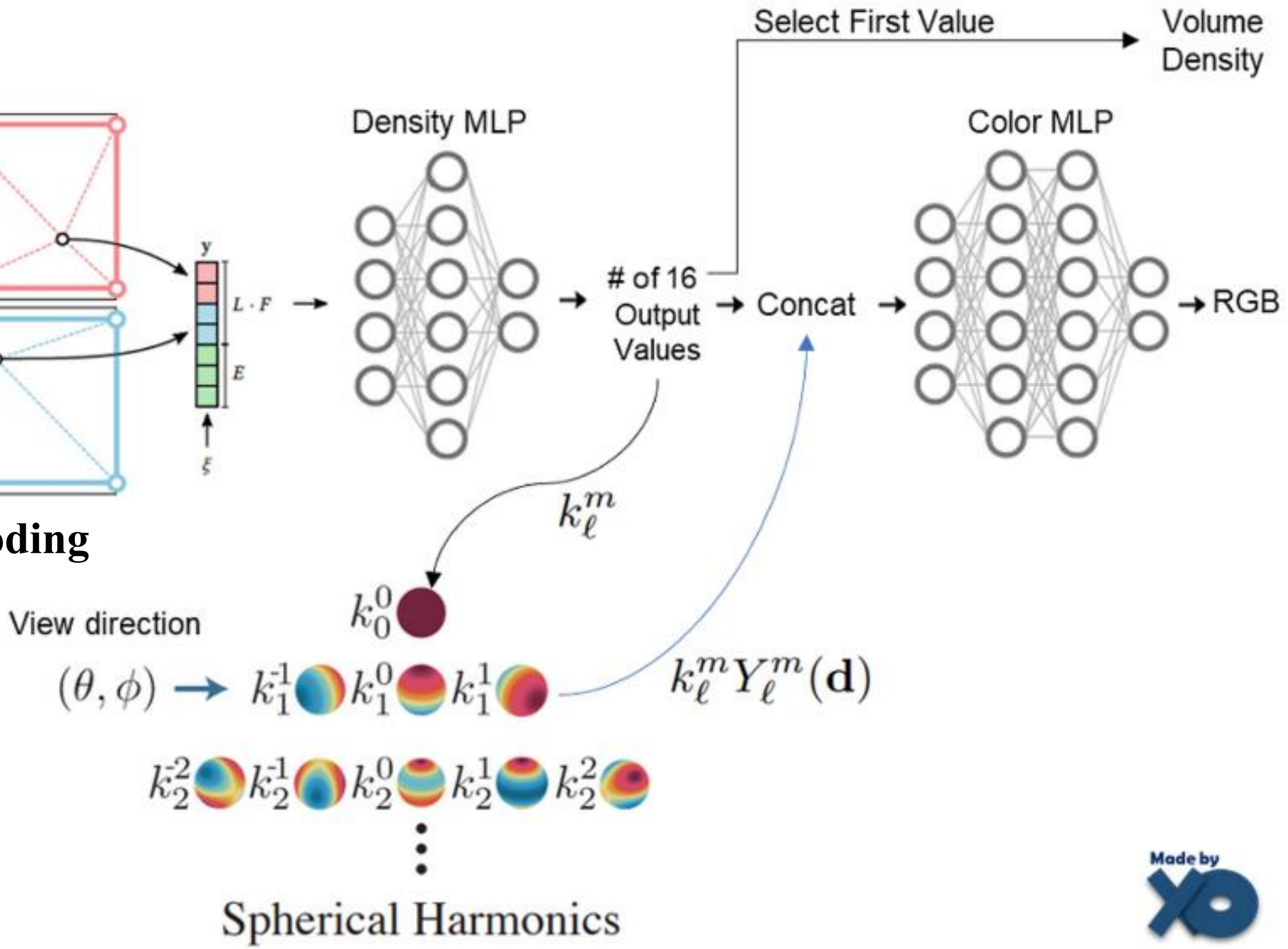
$$b := \exp\left(\frac{\ln N_{\max} - \ln N_{\min}}{L - 1}\right)$$

Instant NeRF 의 Pipeline

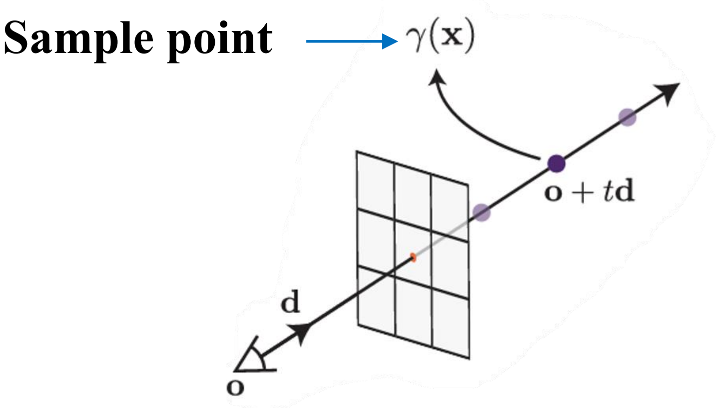


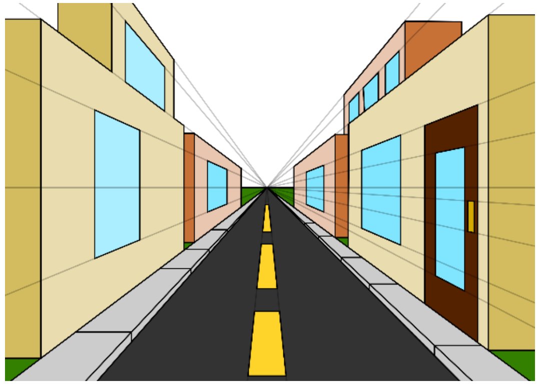
Multi-resolution hash encoding





Volume rendering and Loss Function





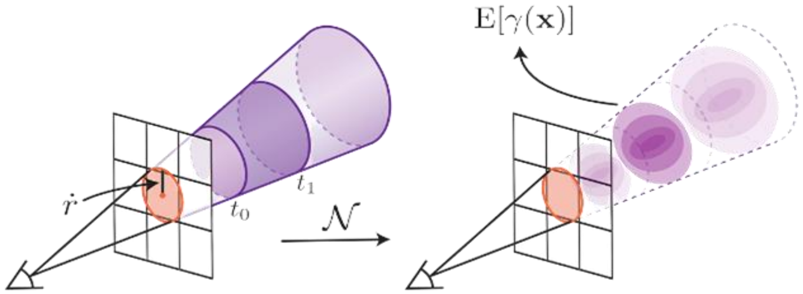
- Volume rendering

$$\underbrace{C(\mathbf{r})}_{\text{Color}} = \int_{t_n\ (\text{near})}^{t_f\ (\text{far})} T(t)\underbrace{\sigma(\mathbf{r}(t))}_{\text{Density}}\underbrace{\mathbf{c}(\mathbf{r}(t),\mathbf{d})}_{\text{Color}}dt\,, \text{ where } \underbrace{T(t)}_{\text{Transmittance}} = \exp\left(-\int_{t_n}^{t}\sigma(\mathbf{r}(s))ds\right)$$

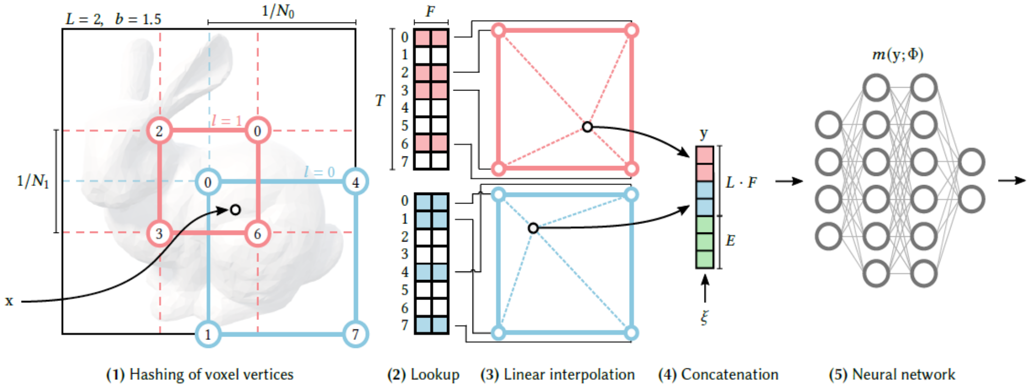
- Loss Function

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

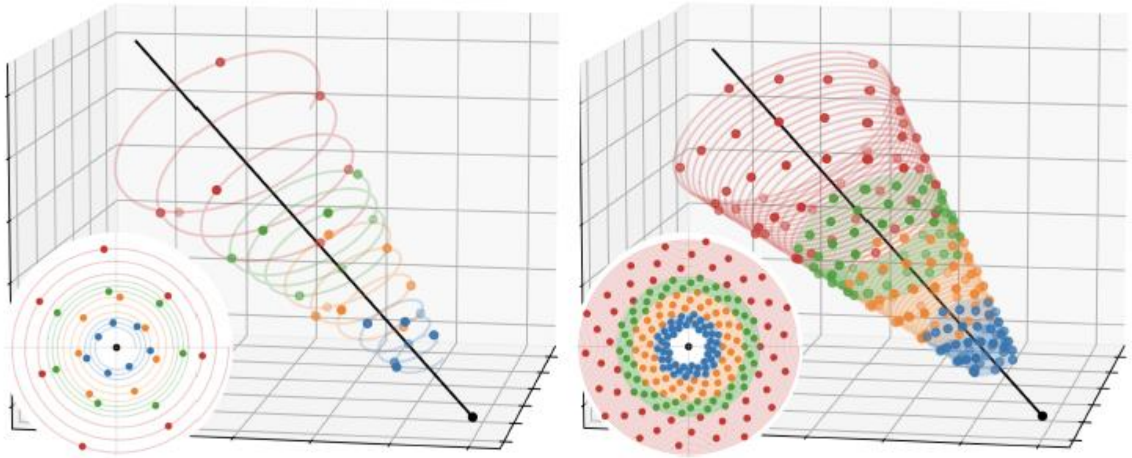
현재의 SOTA, Zip-NeRF



Cone Tracing
in Mip-NeRF360



Multi-resolution hash encoding
in Instant NGP



Zip-NeRF
arXiv, April 2023