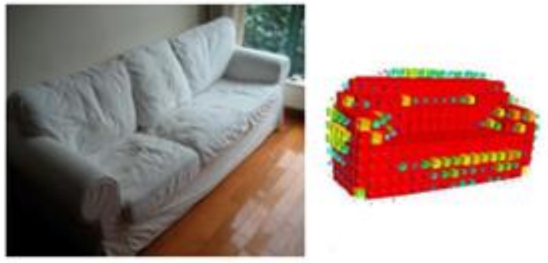
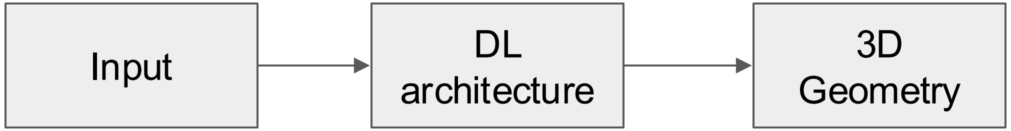
Neural Radiance Fields

Single View 3D Reconstruction

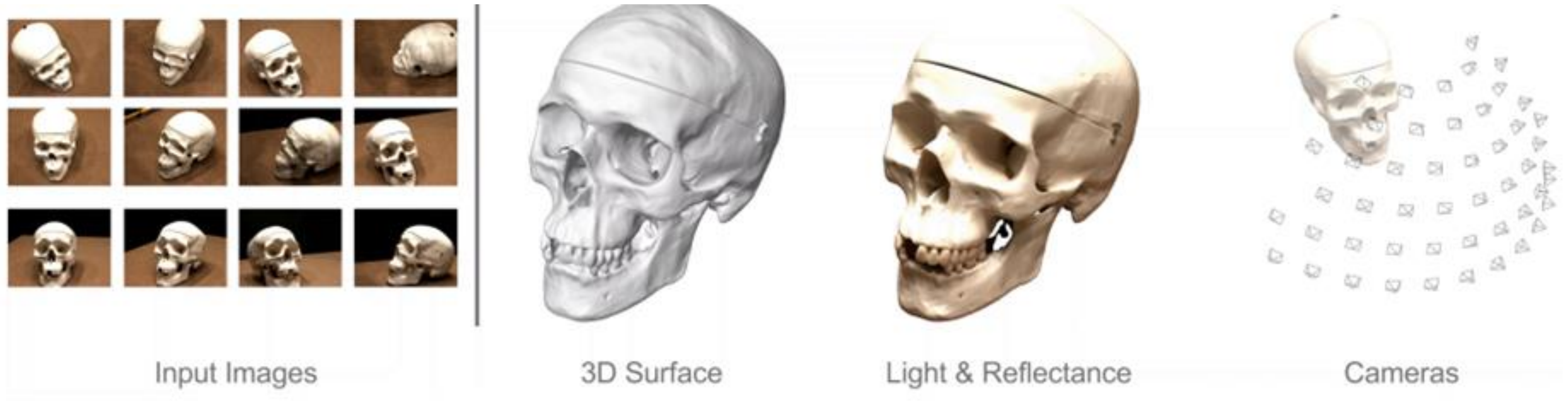


Objective : Given a single view (RGB / Grayscale/ ..) , estimate the 3D geometry of the object



Methods : Occupancy grid estimation, Truncated Signed Distance Function (TSDF) regression, parametric models like SMAL & SMPL , etc.

Multi-View 3D Reconstruction



What happens when we have multiple views?

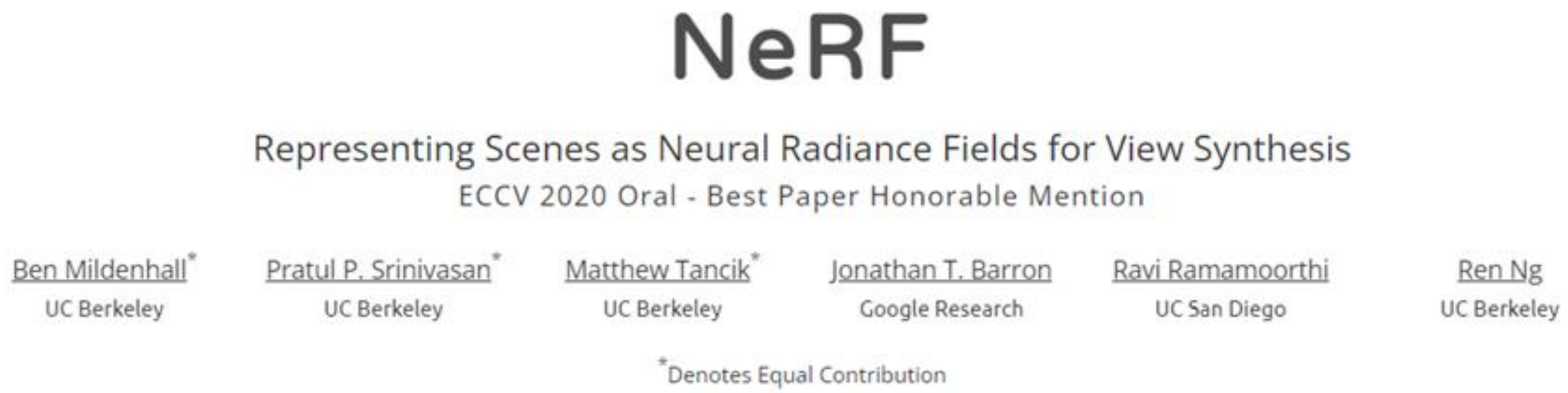
Objective : Find a 3D representation which is consistent with input views.

Application : Rendering novel views (can observe freely from any point in 3D space)

Methods : Structure from Motion (SfM), COLMAP, **Neural Radiance Fields (NeRFs)**

Citations So Far….

- Original NeRF paper - ECCV 2020 Oral - 2694 citations in 3 years



NeRF

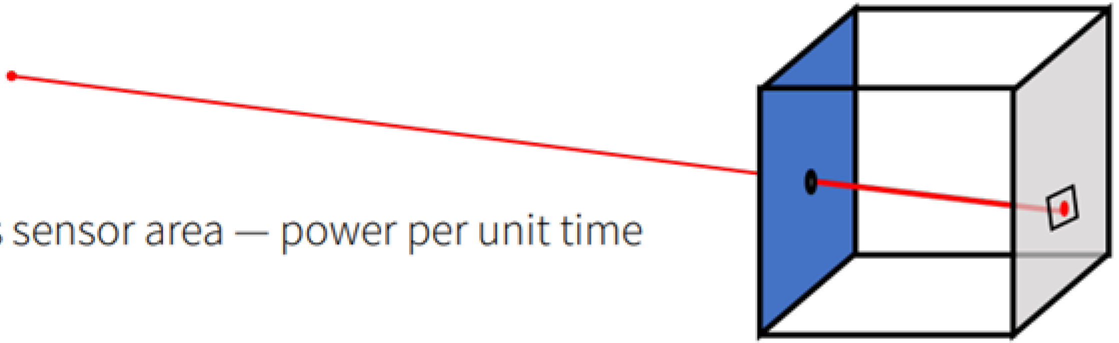
Representing Scenes as Neural Radiance Fields for View Synthesis

ECCV 2020 Oral - Best Paper Honorable Mention

Ben Mildenhall*	Pratul P. Srinivasan*	Matthew Tancik*	Jonathan T. Barron	Ravi Ramamoorthi	Ren Ng
UC Berkeley	UC Berkeley	UC Berkeley	Google Research	UC San Diego	UC Berkeley

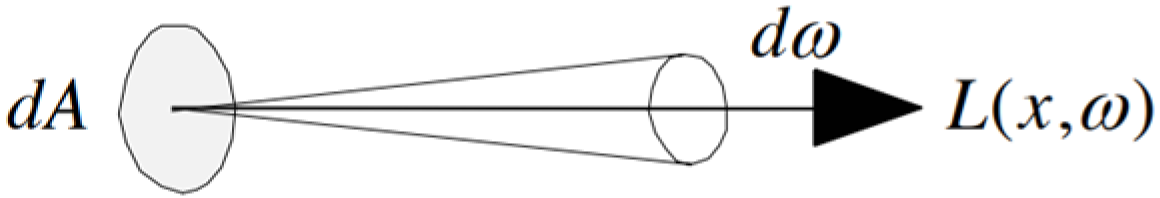
*Denotes Equal Contribution

What does a pixel Measure ?

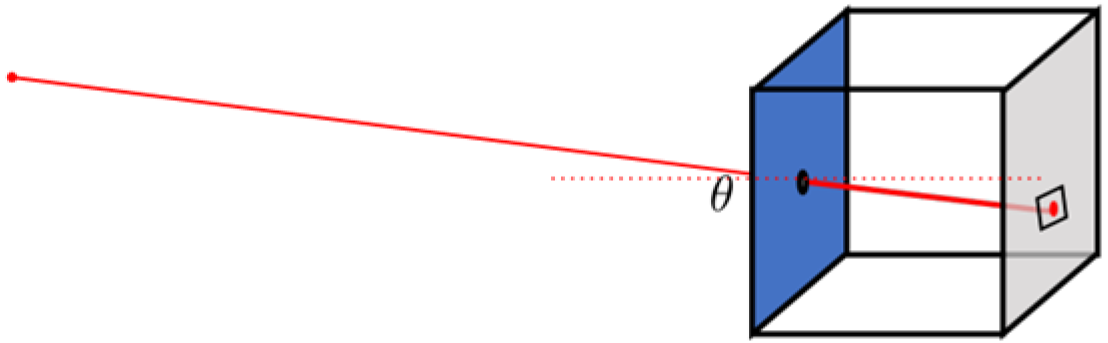


Flux across sensor area — power per unit time

<u>Definition</u>: The field *radiance* (*luminance*) at a point in space in a given direction is the power per unit solid angle per unit area perpendicular to the direction



What does a pixel Measure ?



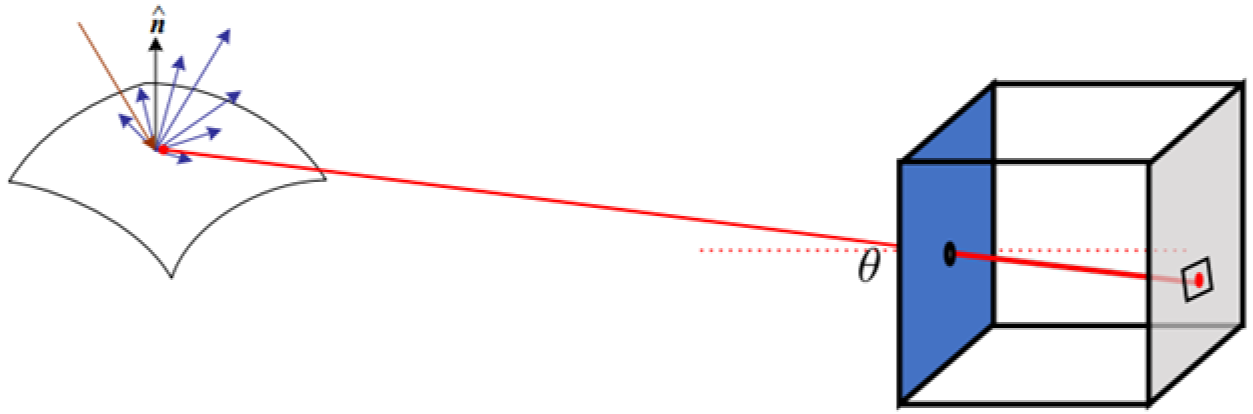
$$\int\int L(\mathbf{x},\omega)\ cos\theta\ dA\ d\Omega$$

(for a very, very narrow band of ***x***, *w* – depending on sensor size and lens)

$$\propto L(\mathbf{x}^*,\omega)$$

radiance for: ***x**** = optical centre, *w* = direction from x* to pixel sensor

Surface-Rendering



Here, we assume that ray travels through air (without any scattering/absorption. Hence, pixel value is radiance of the surface point in that direction

$$\propto L(\mathbf{x}^*, \omega)$$

radiance for: ***x**** = optical centre, *w* = direction from x* to pixel sensor

$$= L(\mathbf{x}^* - \lambda\omega, \omega)$$

Pixel value → Outgoing radiance at a **single** (surface) point

What happens when medium of transmission changes ?

Fog

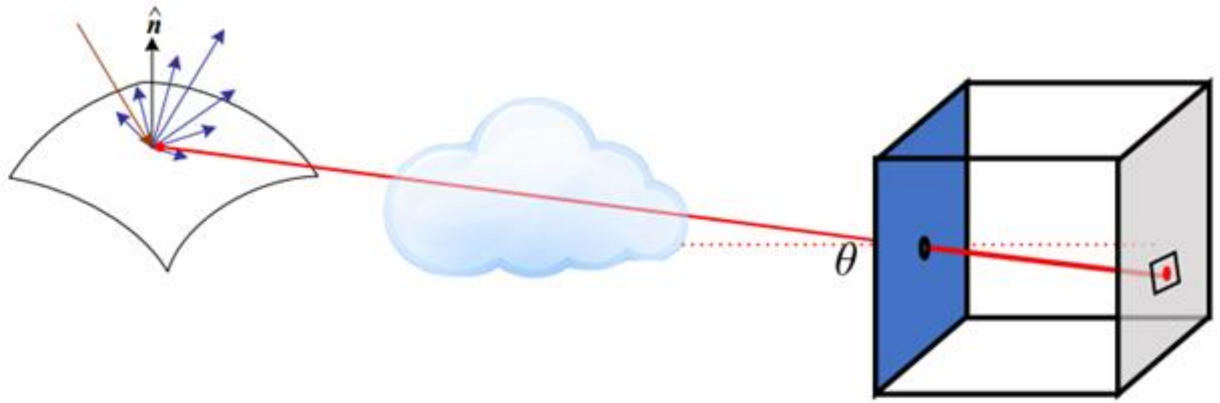
Scattering

Liquids

Exercise :

- List other scenarios where assumption that “pixel value is radiance of the surface point in a given direction”.

Surface-Rendering



$$\propto L(\mathbf{x}^*, \omega)$$

radiance for: ***x**** = pixel sensor centre, *w* = direction from x to optical centre

$$\neq L(\mathbf{x}^* - \lambda\omega, \omega)$$

Pixel value **cannot** be reduced to emitted radiance by a single surface point

How to model L(x,ω) vary along a ray ?

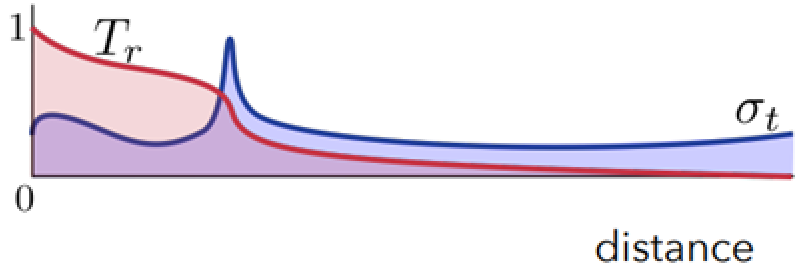
Transmittance

$$T(\mathbf{x}, \mathbf{y})$$

What fraction of radiance at **x** in direction of **y**, reaches **y** ?
(along a straight line under absorption-only model)

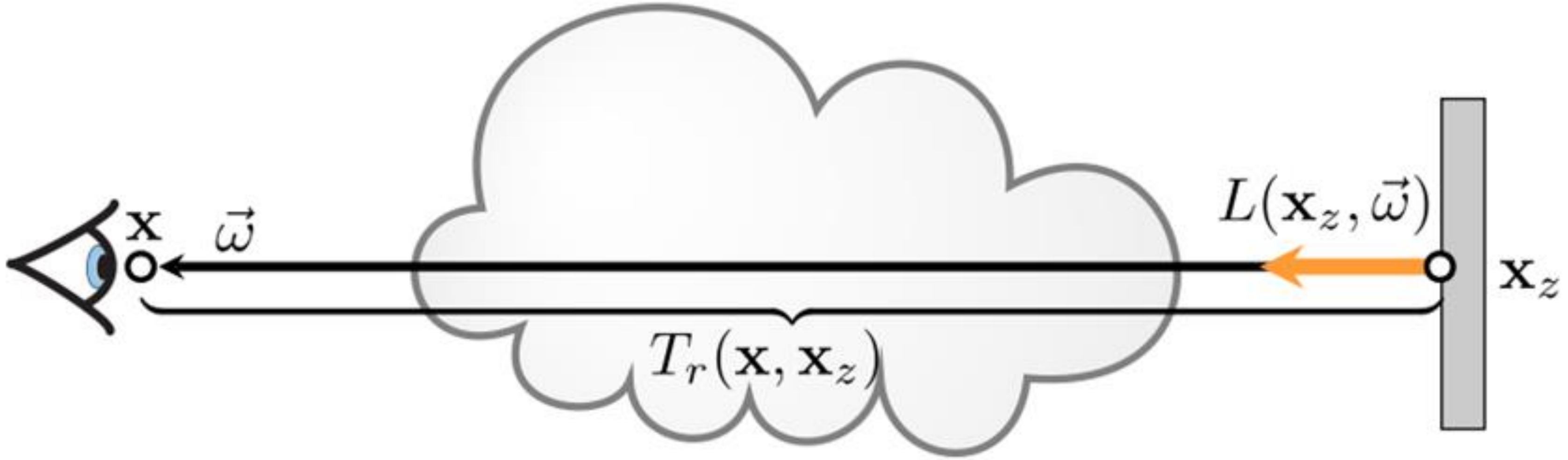
Homogenous Medium: $$e^{-\sigma\|\mathbf{x}-\mathbf{y}\|}$$

Non-Homogenous Medium: $$e^{-\int_{t=0}^{\|\mathbf{x}-\mathbf{y}\|} \sigma(\mathbf{x}+\omega\mathbf{t})}$$



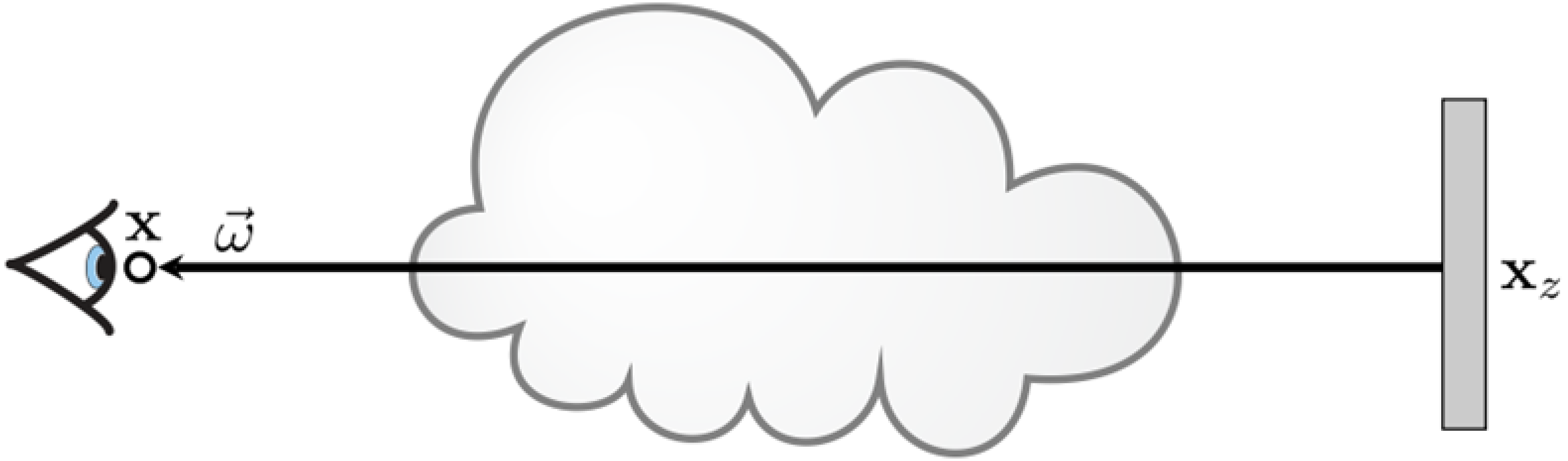
Multiplicativity: $$T(\mathbf{x}, \mathbf{y}) = T(\mathbf{x}, \mathbf{z})T(\mathbf{z}, \mathbf{y})$$

Absorption Only Volumetric Rendering


$\mathbf{x}$
$\vec{\omega}$
$L(\mathbf{x}_z, \vec{\omega})$
$\mathbf{x}_z$
$T_r(\mathbf{x}, \mathbf{x}_z)$

$$L(\mathbf{x}, \omega) = T(\mathbf{x}, \mathbf{x}_z) L(\mathbf{x}_z, \omega)$$

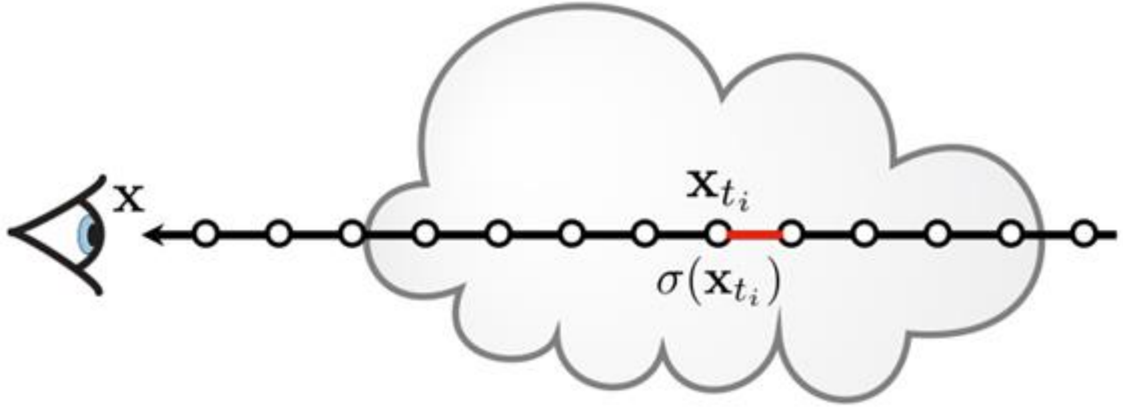
Emission-Absorption Volumetric Rendering



$$L(\mathbf{x}, \omega) = T(\mathbf{x}, \mathbf{x}_z) L(\mathbf{x}_z, \omega) + \int_0^z T(\mathbf{x}, \mathbf{x}_t) \sigma(\mathbf{x}_t) L_e(\mathbf{x}_t, \omega) dt$$

Can we find this model analytically ? or Should we find an approximation ?

Volume Rendering



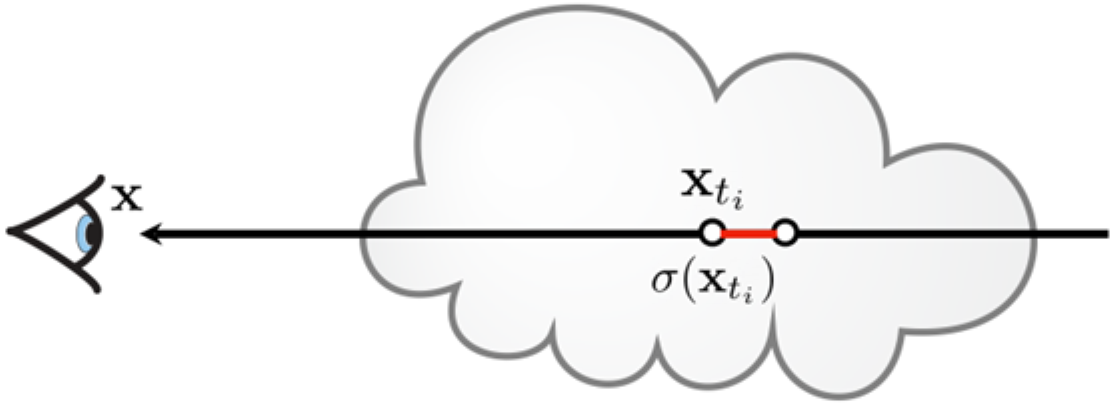
$$L(\mathbf{x}, \omega) = \sum_{i=1}^{N} \text{(contribution from ith segment)}$$

Approximate with a discrete sum

$\mathbf{x}_{t_i}$: ith sample along ray at depth t_i

Δt : distance between successive samples

Volume Rendering

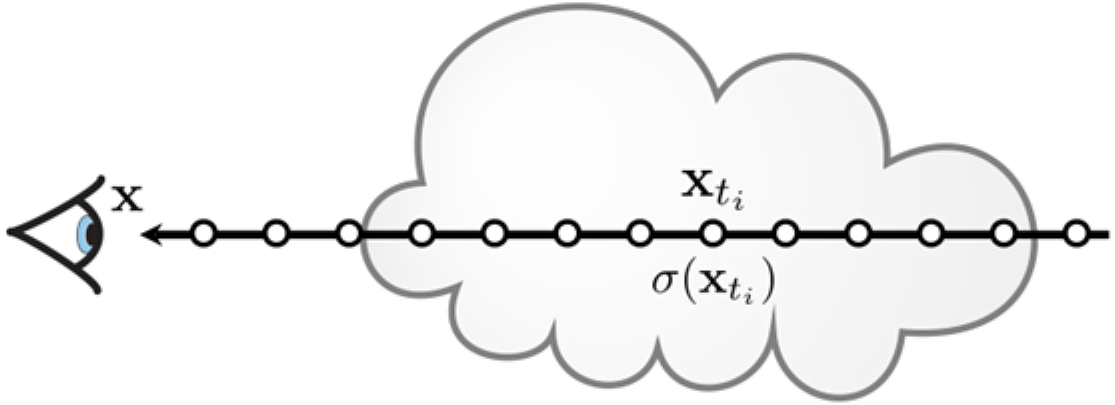


$$L(\mathbf{x},\omega) = \sum_{i=1}^{N} T(\mathbf{x},\mathbf{x}_{t_i}) \underline{(1-e^{-\sigma_{t_i}\Delta t})L_e(\mathbf{x}_{t_i},\omega)}$$

$\mathbf{x}_{t_i}$: ith sample along ray at depth t_i

Δt : distance between successive samples

Volume Rendering



1. Draw uniform samples along a ray (N segments, or N+1 points)
2. Compute transmittance between camera and each sample
3. Aggregate contributions across segments to get overall radiance (color)

Volume Rendering - Summary

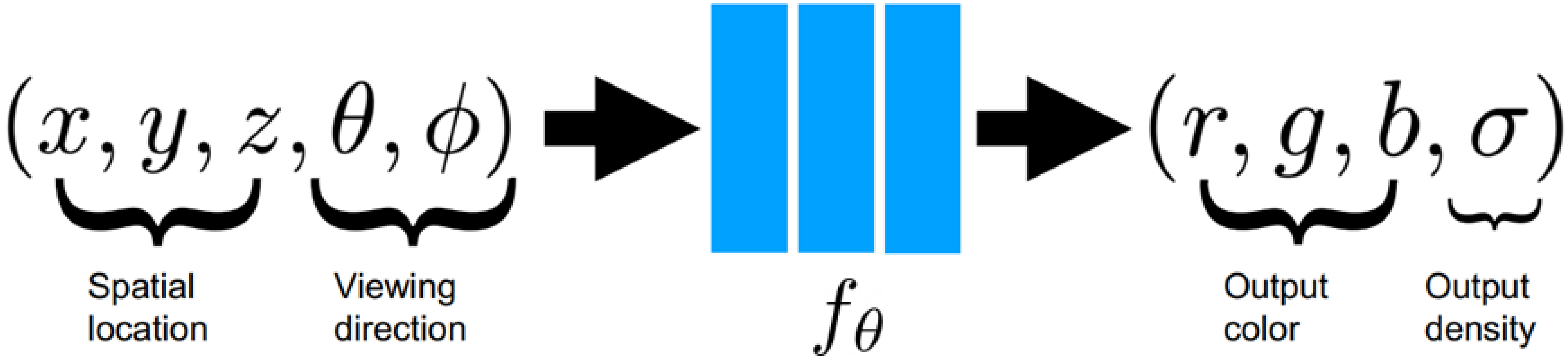
Rendering a ray along medium :

- Computer per-point density
- Computer per-point emitted color given a direction

If we can render a ray, we can synthesize the entire image.

<u>Differentiable</u> w.r.t density, emitted light !!!

Neural Radiance Fields



A scene is represented by NeRF such that :

- Given an input point, the network predicts density
- Given an input point and direction, the network predicts color

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

Volumetric Rendering - Neural Radiance Fields

Rendering model for ray r(t) = o + td:

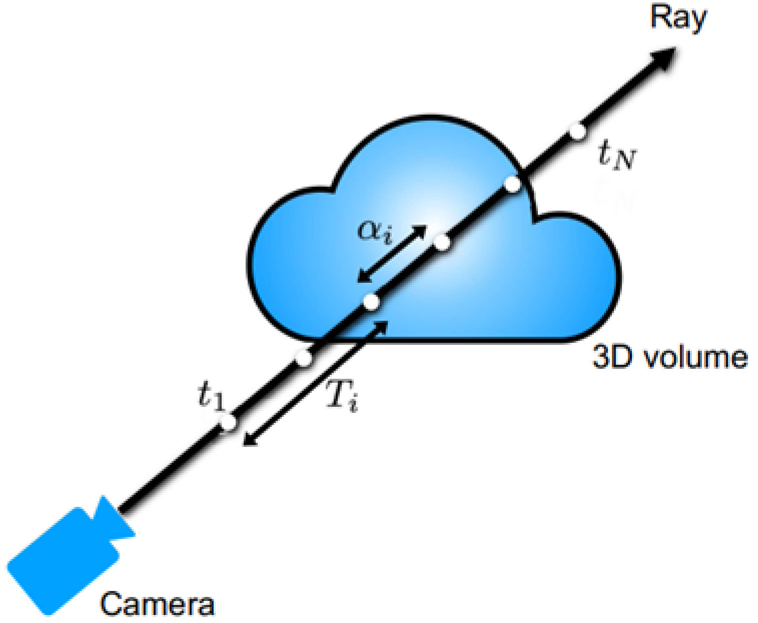
$$C \approx \sum_{i=1}^{N} \underbrace{T_i \alpha_i}_{\text{weights}} \underbrace{c_i}_{\text{colors}}$$

How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

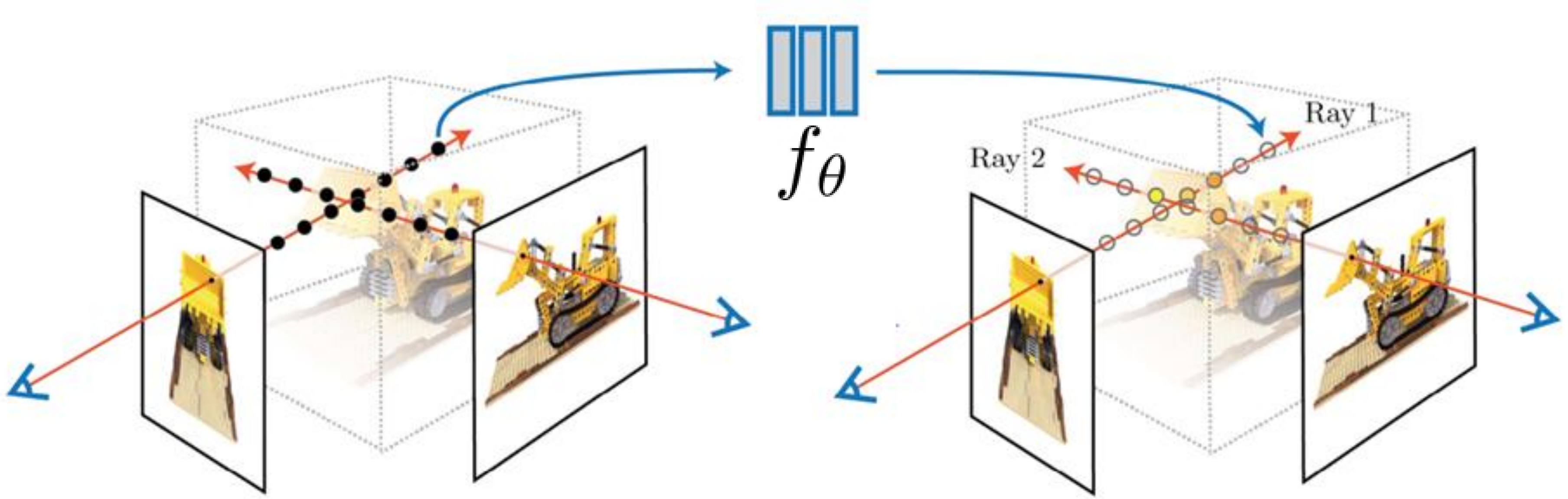
How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i} \leftarrow \text{Density * Distance Between Points}$$

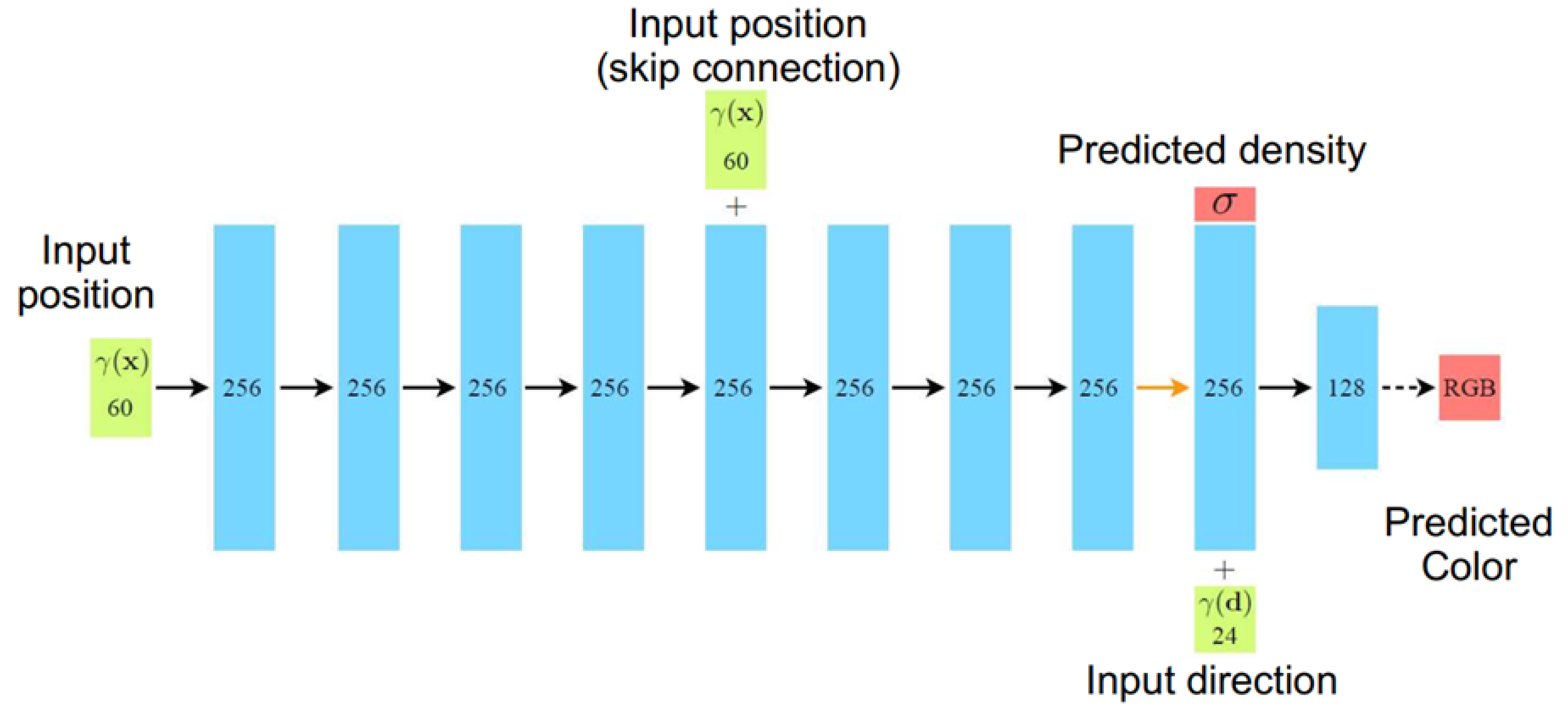


Training - Neural Radiance Fields

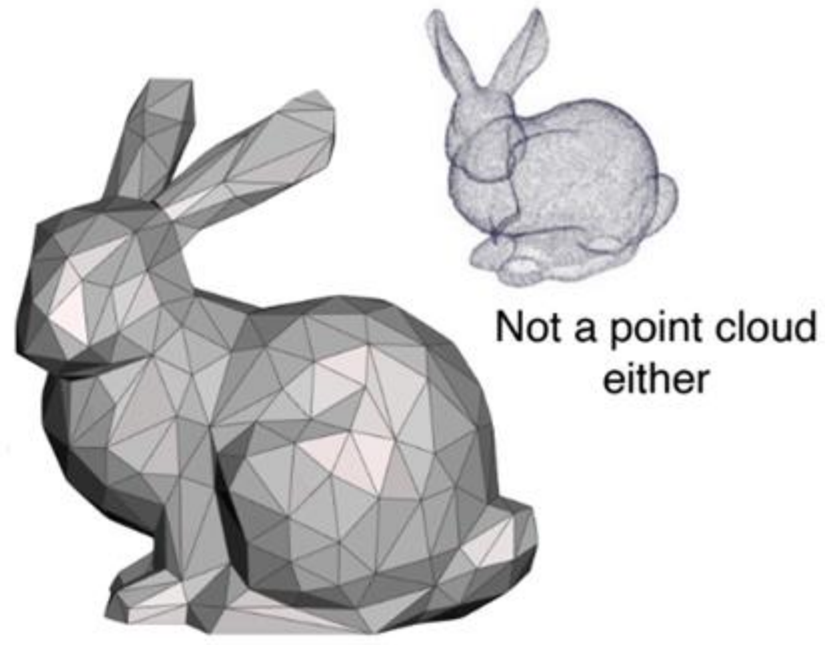
$$\min_{\theta} \sum_{I} \sum_{\mathbf{p}} \| \underbrace{\text{render}(\overset{\text{(pixel, camera) -> ray}}{\mathbf{p}, \pi}; f_\theta)}_{\text{volume rendering}} - I[\mathbf{p}] \|^2$$



Network Structure

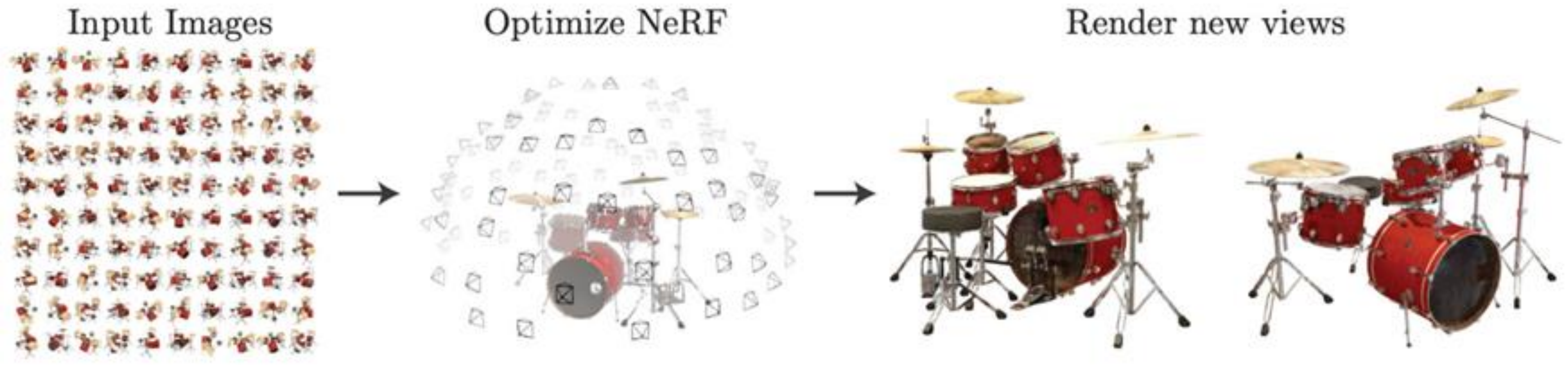


What kind of 3D representation NeRF learns ?



- It is neither a point cloud nor a 3D mesh
- It is a continuous voxel representation of the 3D scene

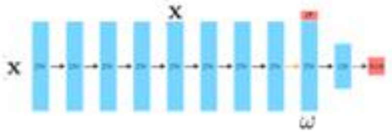
Story So Far ……



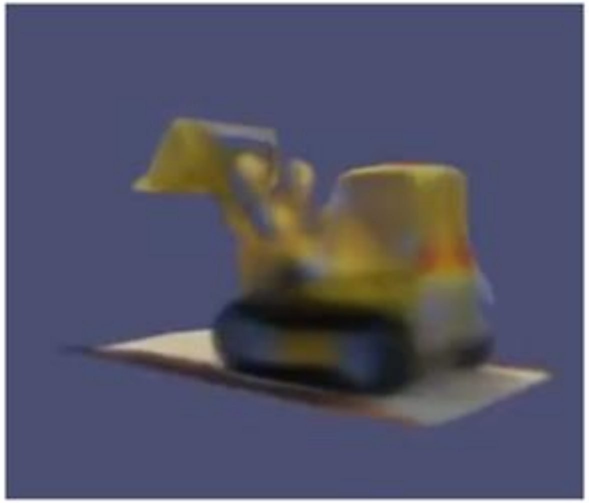
- Acquire multi-view images of a scene
- Run COLMAP to extract camera-poses
- Define a NeRF : $\boldsymbol{f_\theta}$
- Train $\boldsymbol{f_\theta}$ with a volumetric rendering based reconstruction loss

Initial Attempt

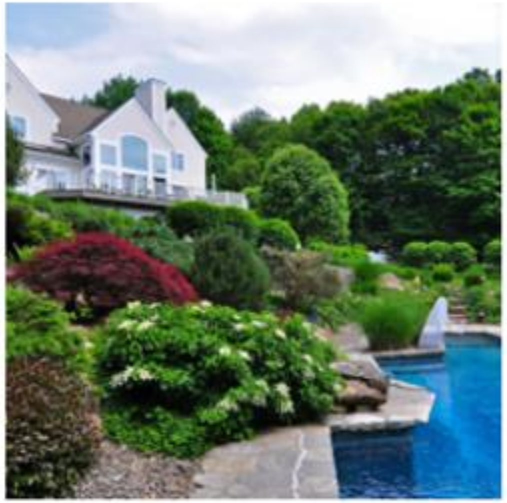
100 training images

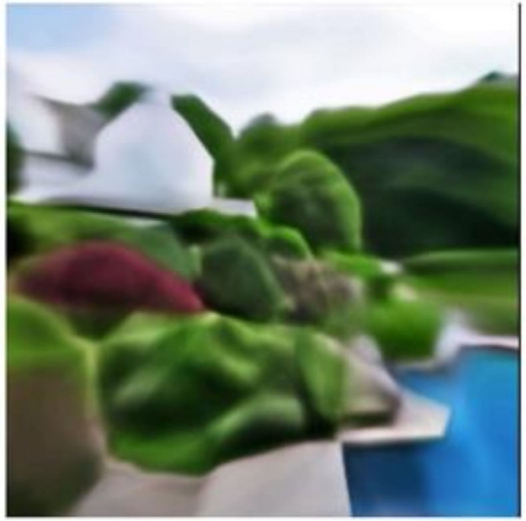


Optimized Neural Net


Novel-view Renderings

NeRF - Positional Encoding


Input Image


Using a ‘standard’ MLP

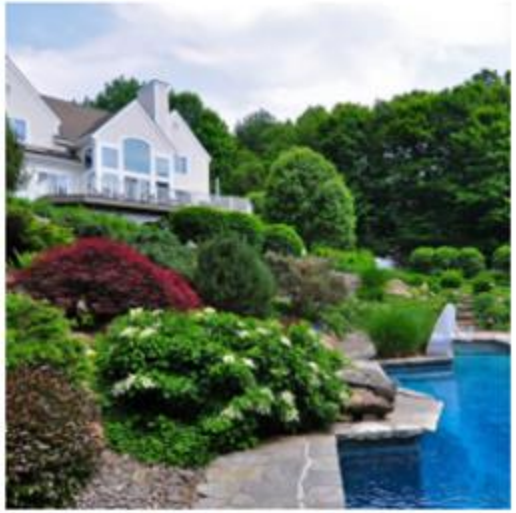
$$\mathbf{v}$$

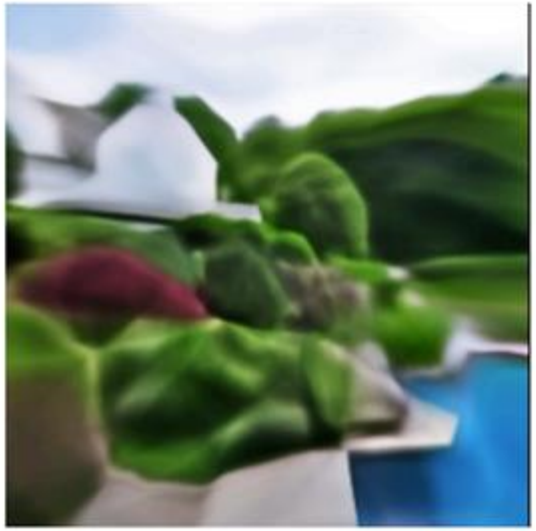
$$\begin{gathered}\sin(\mathbf{v}), \cos(\mathbf{v}) \\ \sin(2\mathbf{v}), \cos(2\mathbf{v}) \\ \sin(4\mathbf{v}), \cos(4\mathbf{v}) \\ \dots \\ \sin(2^{L-1}\mathbf{v}), \cos(2^{L-1}\mathbf{v})\end{gathered}$$

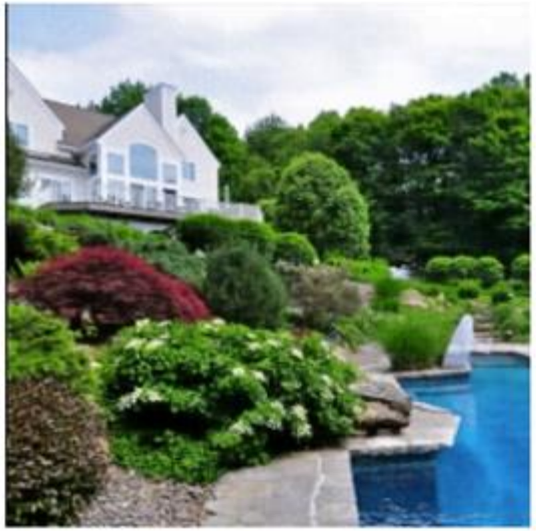
$$\gamma(\mathbf{v})$$

Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. Tancik et. al.

NeRF - Positional Encoding

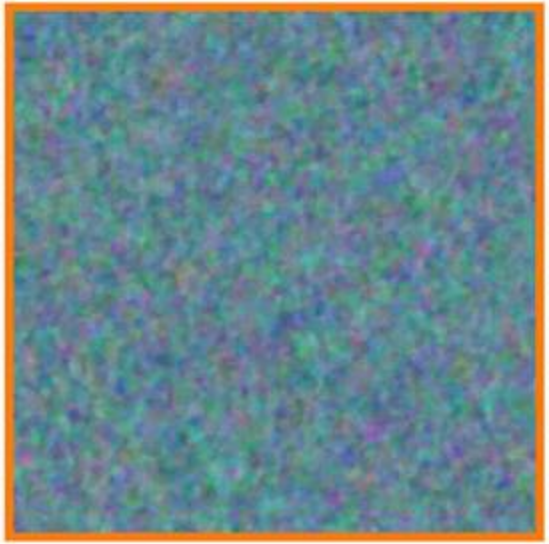

Input Image

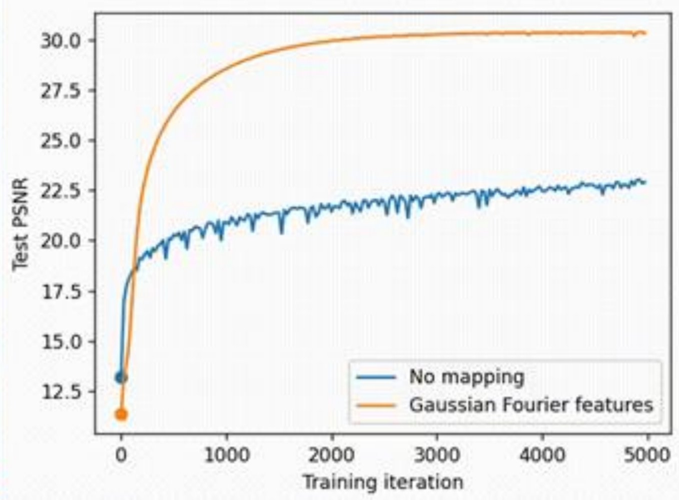

Using a ‘standard’ MLP


Using position encoding

Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. Tancik et. al.

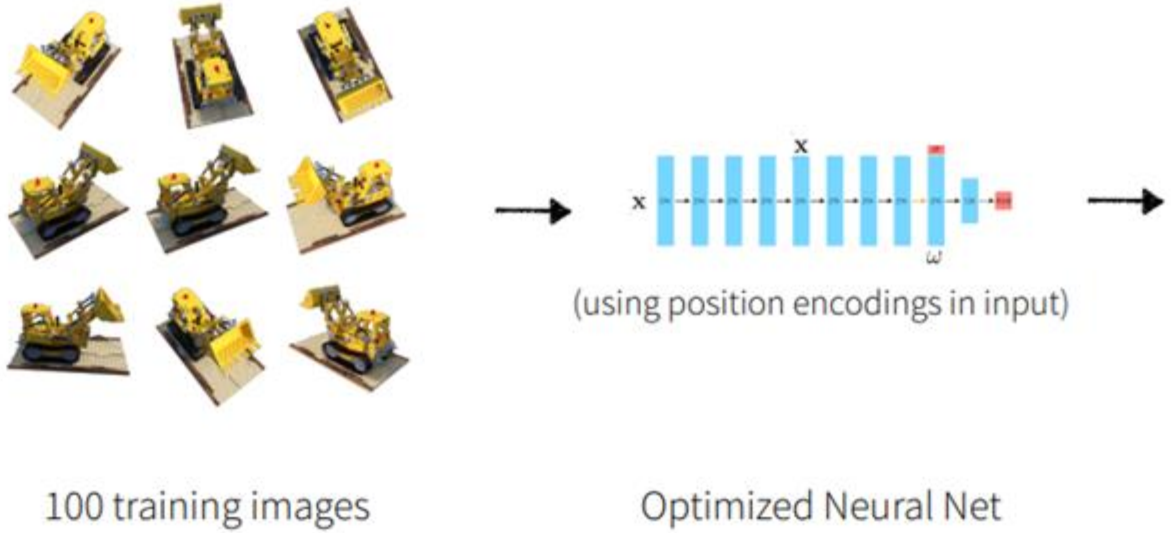
NeRF - Positional Encoding



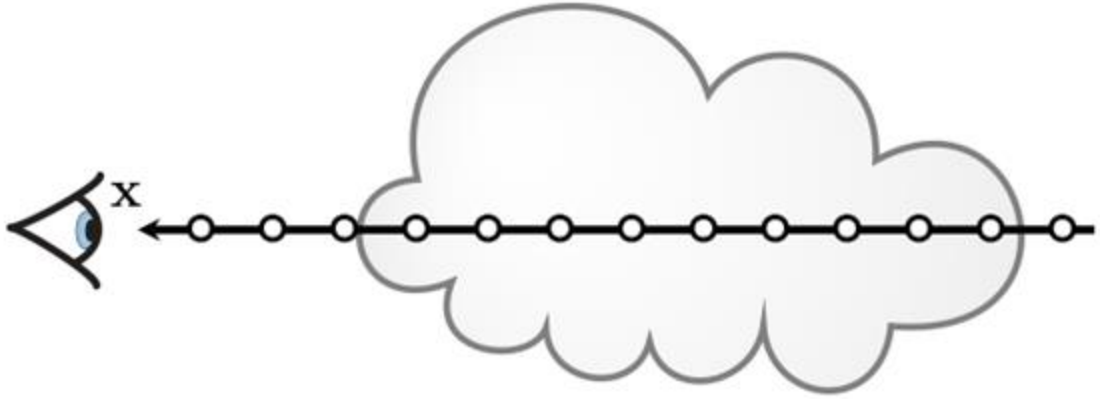


Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. Tancik et. al.

NeRF - Positional Encoding

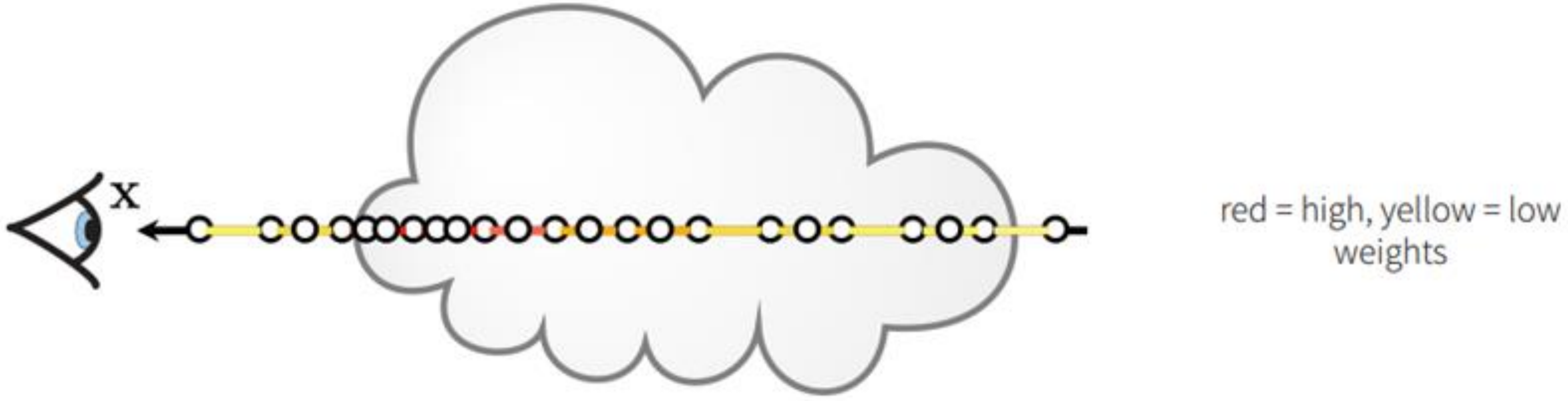


NeRF - Uniform To Hierarchical Sampling



$$L(\mathbf{x},\omega) = \sum_{i=1}^{N} T(\mathbf{x},\mathbf{x}_{t_i})(1-e^{-\sigma_{t_i}\Delta t})L_e(\mathbf{x}_{t_i},\omega)$$

NeRF - Uniform To Hierarchical Sampling

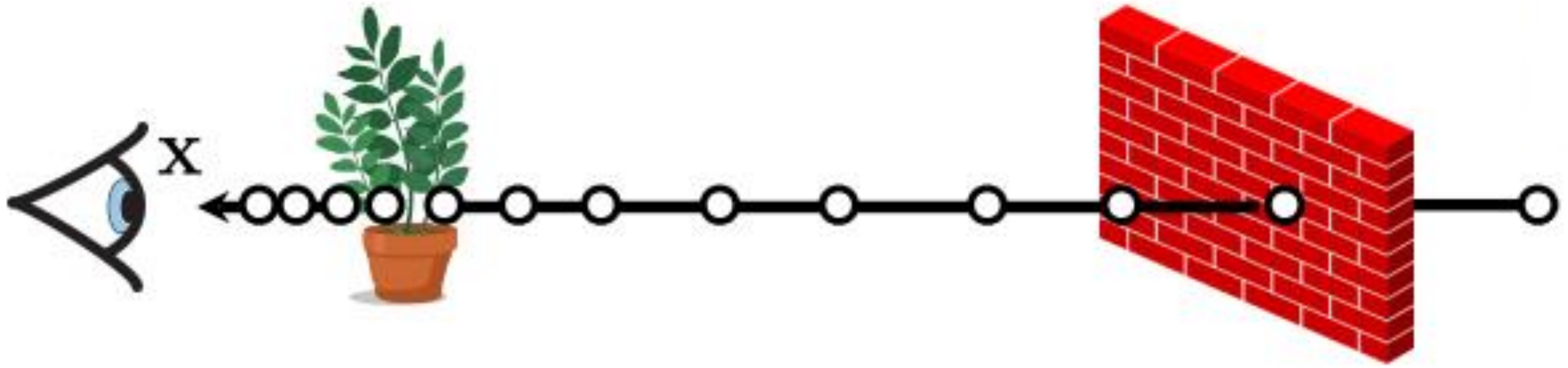


$$L(\mathbf{x}, \omega) = \sum_i w_i \mathbf{c}_i; \quad w_i = T_i(1 - e^{-\sigma_{t_i} \Delta t})$$

weights indicate how much each segment contributes to the pixel value

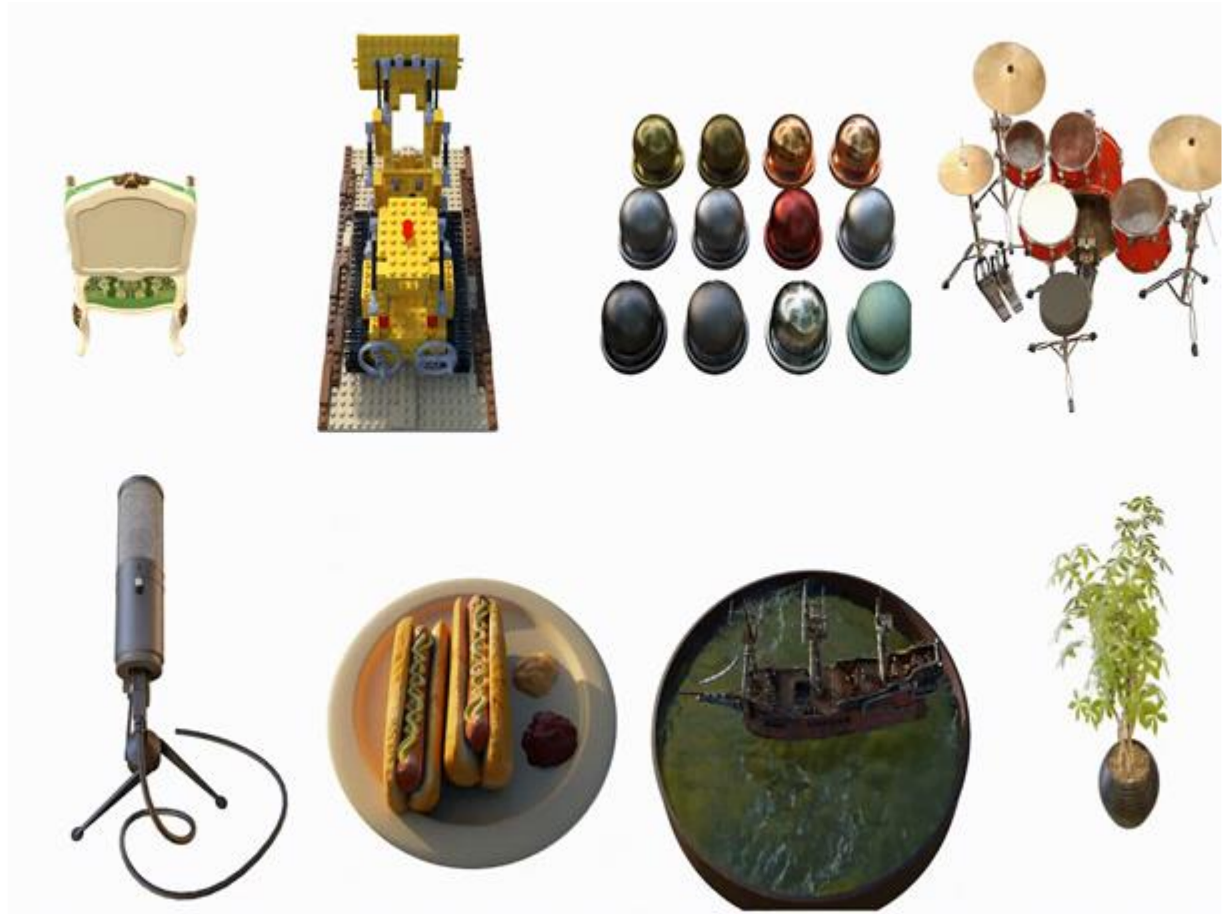
- Coarser Network : Sample points uniformly
- Finer Network : Sample more points depending on weights of the segments

NeRF - Inverse Depth Sampling



- **Problem :** Need to model background for real-scenes. For e.g. sky is at infinity.
- In general, objects of interest are more closer to the camera. Hence sample, uniformly in (1/*d*) space; where *d* is depth

Synthetic Scenes



Real Scenes

Depth Visualization

Applications in Augmented Reality

Mip-NeRF

Mip-NeRF

- NeRF uses a single ray per-pixel and may reproduce renderings that are blurred or aliased
- Instead, we need to render multiple rays as illustrated in the purple cone Fig. (b)
- As we know rendering is expensive, it’s impractical to render multiple rays for each cone
- Mip-NeRF extends NeRF to represent scenes at continuous scale by rendering anti-aliased frustums

Mip-NeRF – Integrated Positional Encoding

Positonal Encoding (PE) maps a single point into a feature vector

IntegratedPositonal Encoding (IPE) considers gaussians instead of infinitesimal points
This allows “region of space” as query to a coordinated network

When wider region is considered, contribution from higher frequencies shrinks down.

Mip-NeRF - Results

Mip-NeRF - Results

Mip-NeRF 360

Limitations of Previous Representations :

- Both NeRF and mip-NeRF needs a bounded domain i.e they only show results on forward-facing scenes or synthetic datasets

Problem in Representing Unbounded Scenes :

- Large scene requires more network capacity, which is expensive
- Observations are sparse and reconstruction becomes ill-posed
- How to model far away objects like distant wall, horizon ?

Solutions:

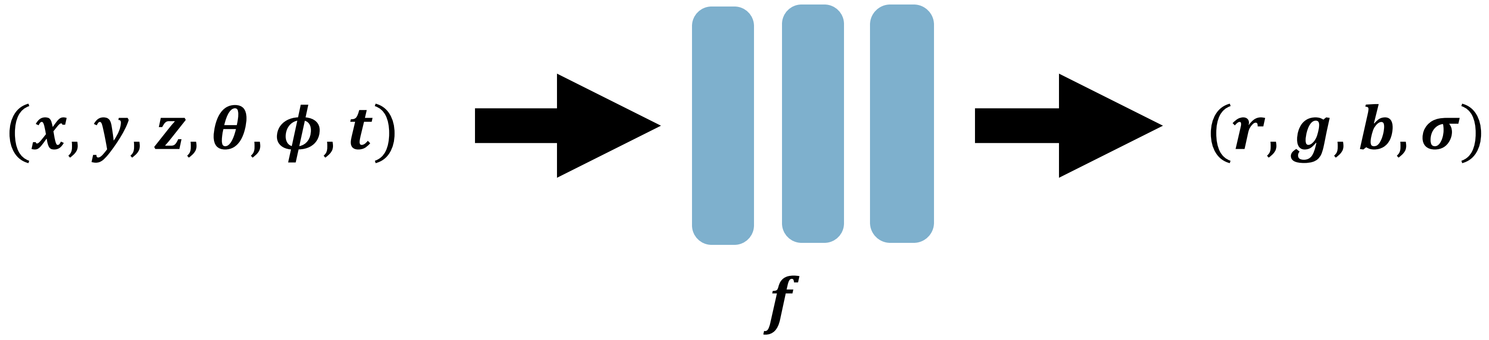
- Apply a Kalman-like warp to mip-NeRF Gaussians
- Online distillation from large MLP to small MLP
- Regularization of density along ray intervals

Mip-NeRF 360

Mip-NeRF casts Gaussians from camera positions. For large scenes, at far away distances we get elongated Gaussians which violates bounded assumption in mip-NeRF

Mip-NeRF 360 contracts the elongated Gaussians in the bounded space of mip-NeRF

Dynamic Scenes – A Simple Approach



- Does-not generalize well for strictly monocular views
- Works reasonably well for multi-view dynamic scenes but fails to model disocclusions, shadows etc.

D-NeRF: Neural Radiance Fields for Dynamic Scenes

- Maps an observed/deformed scene to canonical space using a deformable network
- Fits a radiance field for the canonical space
- Decouples motion and space by using two separate MLP networks

Nerfies: Deformable Neural Radiance Fields

- Trace camera rays in the observed frame and transform samples along the canonical space using deformation field
- Query the template NeRF for these transformed points
- Instead of position-encoded time, they use a learnable deformation field
- Appearance code is used to handle illumination variations

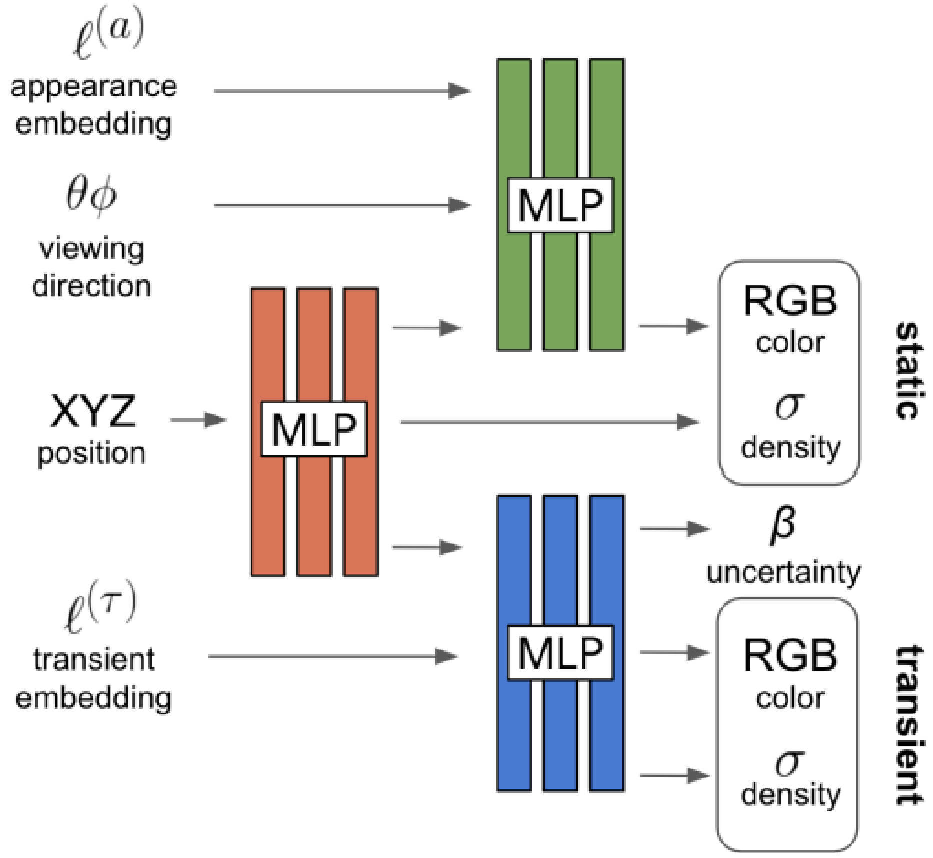
Hyper-NeRF

- Trace camera rays in the observed frame and warps these sampled points to the canonical space using deformation field
- Slice a surface from canonical hyper-space using ambient slocing network
- Concatenate them and query the template NeRF

Appearance Changes

- Exposure Differences
- Lighting Changes (Day, Night, …)
- Passing by clouds
- External Lighting

NeRF in the Wild



- Use learnable appearance embeddings to model exposure variations
- Use learnable transient embeddings to model transient color variations.

References

- **https://www.matthewtancik.com/nerf**
- https://learning3d.github.io/
- https://sites.google.com/berkeley.edu/nerf-tutorial/home
- https://jonbarron.info/mipnerf/
- https://jonbarron.info/mipnerf360/
- https://www.albertpumarola.com/research/D-NeRF/index.html
- https://nerfies.github.io/
- https://nerf-w.github.io/
- https://dreamfusion3d.github.io/