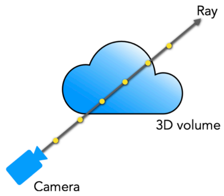
Volume Differentiable Rendering & NeRFs

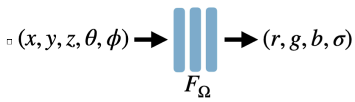
Based on slides from [Gkioulekas, 2025], [Takikawa et al, 2023] and [Tulsiani, 2024]

Recap - Volume Rendering Pipeline

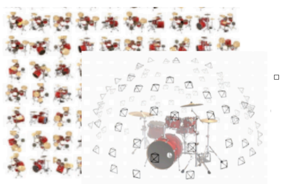
- Differentiable Volumetric Rendering Function



- Neural Volumetric 3D Scene Model



- Reconstruction via Analysis-bySynthesis



Outline

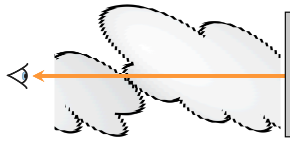
- Radiative Transfer Equation
- Volume Rendering Equation
- Differentiable Rendering
- 3D Scene Models
- NeRFs

Radiative Transfer Equation

Slides from [Gkioulekas, 2025]

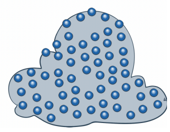
Participating Media

Typically, we do not model particles of a medium explicitly (wouldn't fit in memory, completely impractical to ray trace)



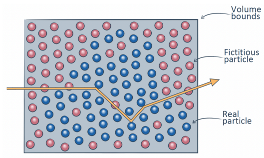
The properties are described statistically using various coefficients and densities

- Conceptually similar idea as microfacet models

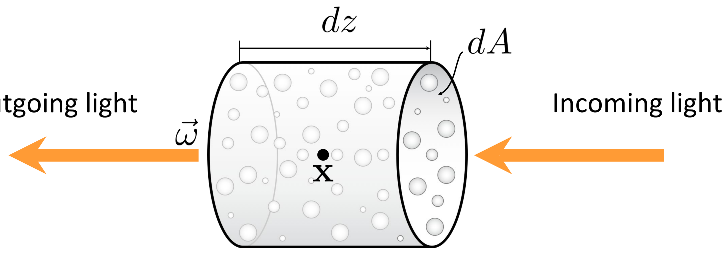


Heterogeneous (spatially varying coefficients):

- Procedurally, e.g., using a noise function
- Simulation + volume discretization, e.g., a voxel grid



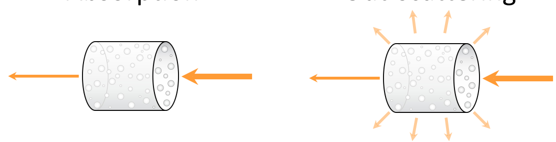
Differential Beam Segment



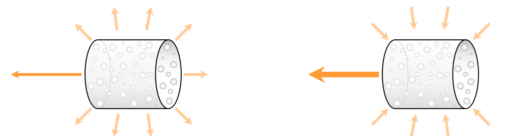
(Gkioulekas)

Radiative Transfer Equation (RTE)

Absorption Out-scattering



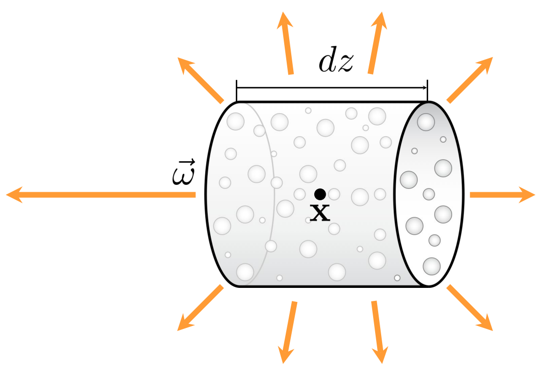
$$dL(\mathbf{x},\vec{\omega}) = \boxed{-\,\sigma_a(\mathbf{x})L(\mathbf{x},\vec{\omega})dz \quad -\,\sigma_s(\mathbf{x})L(\mathbf{x},\vec{\omega})dz} \quad \underline{\text{Losses}}$$
$$\boxed{+\,\sigma_a(\mathbf{x})L_e(\mathbf{x},\vec{\omega})dz \;+\,\sigma_s(\mathbf{x})L_s(\mathbf{x},\vec{\omega})dz} \quad \underline{\text{Gains}}$$



Emission In-scattering

(Gkioulekas)

Emission



$$dL(\mathbf{x},\vec{\omega}) = \sigma_a(\mathbf{x})L_e(\mathbf{x},\vec{\omega})dz$$

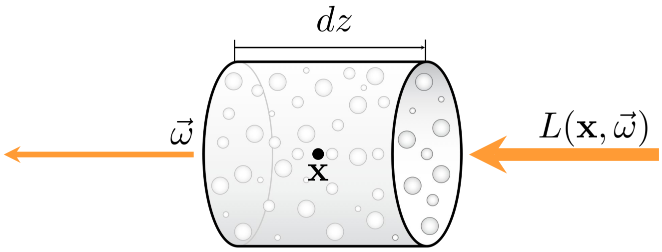
*Sometimes modeled without the absorption coefficient just by specifying a "source" term

$\sigma_a(\mathbf{x})$: absorption coefficient $[m^{-1}]$

$L_e(\mathbf{x},\vec{\omega})$: emitted radiance

(Gkioulekas)

Absorption

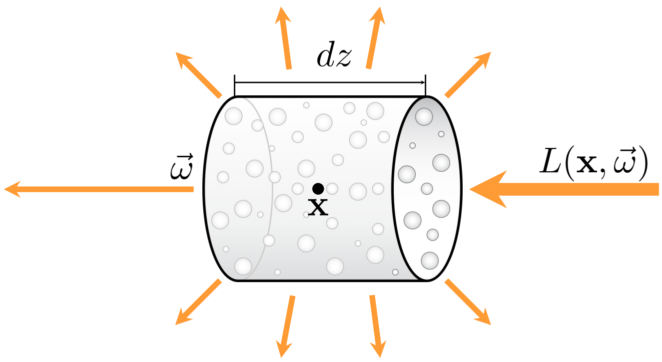


$$dL(\mathbf{x}, \vec{\omega}) = -\sigma_a(\mathbf{x}) L(\mathbf{x}, \vec{\omega}) dz$$

$\sigma_a(\mathbf{x})$: absorption coefficient $[m^{-1}]$

(Gkioulekas)

Out-scattering

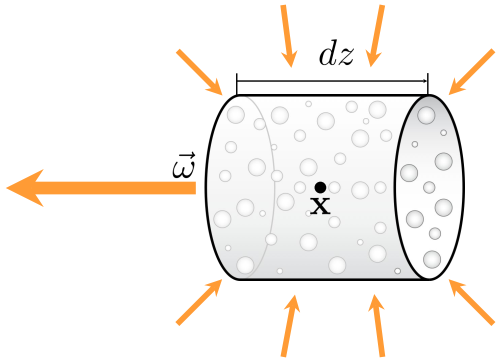


$$dL(\mathbf{x}, \vec{\omega}) = -\sigma_s(\mathbf{x}) L(\mathbf{x}, \vec{\omega}) dz$$

$\sigma_s(\mathbf{x})$: scattering coefficient $[m^{-1}]$

(Gkioulekas)

In-scattering



$$dL(\mathbf{x}, \vec{\omega}) = \sigma_s(\mathbf{x}) L_s(\mathbf{x}, \vec{\omega}) dz$$

$\sigma_s(\mathbf{x})$: scattering coefficient $[m^{-1}]$

$L_s(\mathbf{x}, \vec{\omega})$: in-scattered radiance

(Gkioulekas)

Complexity Progression - Scattering

homogeneous vs. heterogeneous

- none
- fake ambient
- single
- multiple

(Gkioulekas)

Volume Rendering Equation

Slides from [Gkioulekas, 2025]

Transmittance

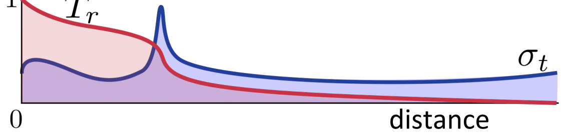
Homogeneous volume:

$$T_r(\mathbf{x}, \mathbf{y}) = e^{-\boxed{\sigma_t \|\mathbf{x}-\mathbf{y}\|}}$$

Heterogeneous volume (spatially varying σ_t):

$$T_r(\mathbf{x}, \mathbf{y}) = e^{-\boxed{\int_0^{\|\mathbf{x}-\mathbf{y}\|} \sigma_t(t)\,dt}}$$

Optical thickness



(Gkioulekas)

Volume Rendering Equation

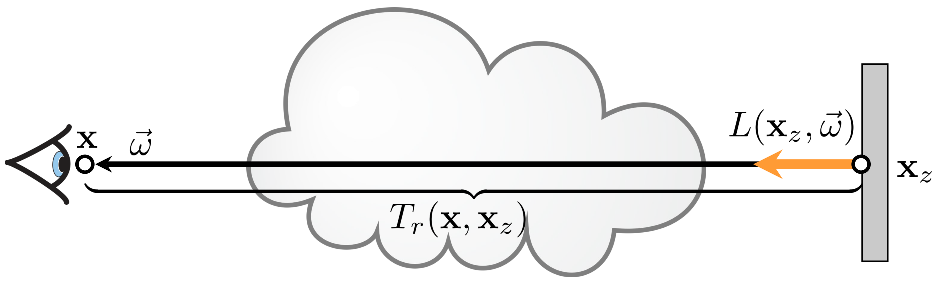
$$L(\mathbf{x},\vec{\omega}) = T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega})$$
$$+ \int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_a(\mathbf{x}_t)L_e(\mathbf{x}_t,\vec{\omega})dt$$
$$+ \int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_s(\mathbf{x}_t)\int_{S^2} f_p(\mathbf{x}_t,\vec{\omega}',\vec{\omega})L_i(\mathbf{x}_t,\vec{\omega}')d\vec{\omega}'dt$$

(Gkioulekas)

Volume Rendering Equation

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

Reduced (background) surface radiance

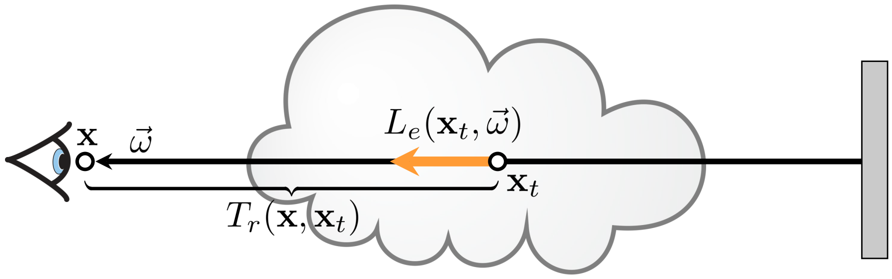


(Gkioulekas)

Volume Rendering Equation

$$L(\mathbf{x},\vec{\omega}) = T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega}) + \boxed{\int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_a(\mathbf{x}_t)L_e(\mathbf{x}_t,\vec{\omega})dt}$$

Accumulated emitted radiance

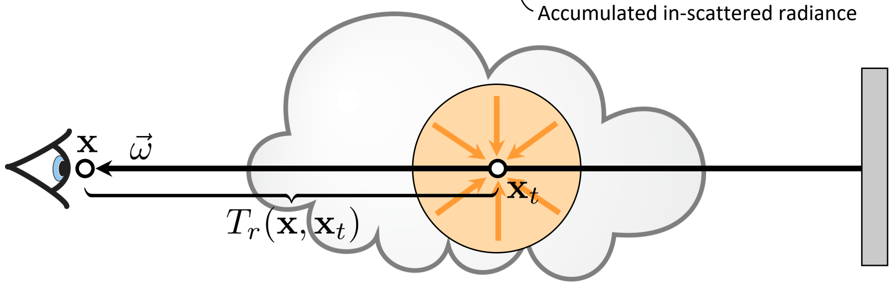


(Gkioulekas)

Volume Rendering Equation

$$L(\mathbf{x},\vec{\omega}) = T_r(\mathbf{x},\mathbf{x}_z)L(\mathbf{x}_z,\vec{\omega}) + \int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_a(\mathbf{x}_t)L_e(\mathbf{x}_t,\vec{\omega})dt + \boxed{\int_0^z T_r(\mathbf{x},\mathbf{x}_t)\sigma_s(\mathbf{x}_t)\int_{S^2} f_p(\mathbf{x}_t,\vec{\omega}',\vec{\omega})L_i(\mathbf{x}_t,\vec{\omega}')d\vec{\omega}'dt}$$

Accumulated in-scattered radiance



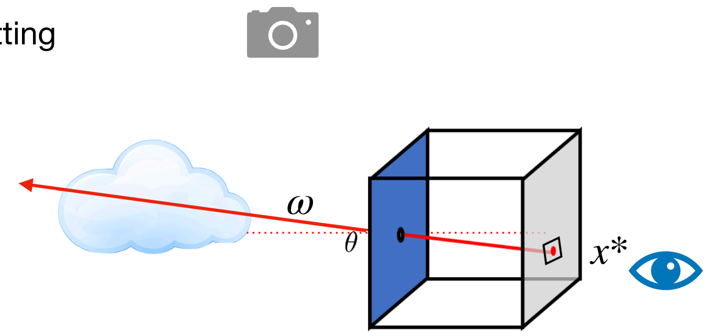
(Gkioulekas)

Differentiable Rendering

Slides from [Tulsiani, 2024]

Differentiable Volume Rendering

- Volume Rendering Setting



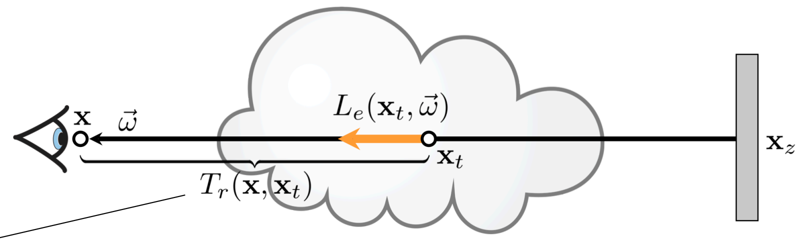
$$\propto L(\mathbf{x}^*, \omega)$$

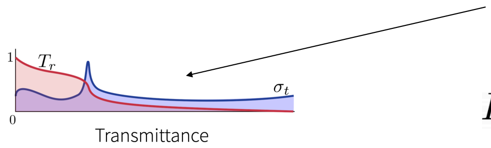
radiance for: ***x**** = pixel sensor centre, *w* = direction from x to optical centre

(Tulsiani)

Mathematical Model

- Emission-Absorption Volume Rendering





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

Background Radiance

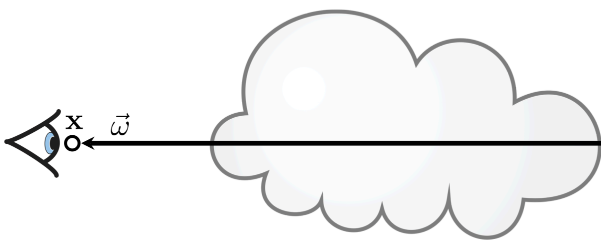
$$+ \int_0^z T(\mathbf{x}, \mathbf{x}_t) \sigma(\mathbf{x}_t) L_e(\mathbf{x}_t, \omega) dt$$

Accumulated Emitted Radiance from inside

(Tulsiani)

Mathematical Model (simplified)

- Emitted Radiance from Inside Volume

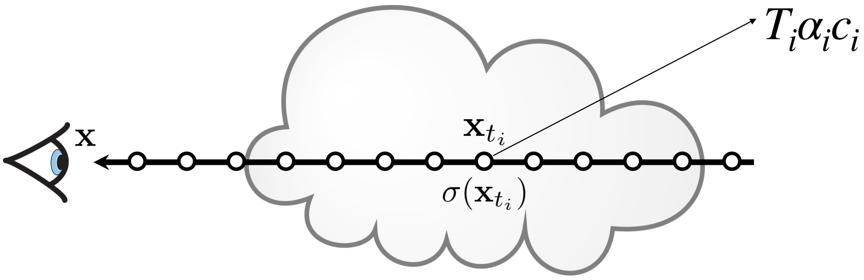


$$L(\mathbf{x}, \omega) = \int_0^z T(\mathbf{x}, \mathbf{x}_t) \sigma(\mathbf{x}_t) L_e(\mathbf{x}_t, \omega) dt$$

Only the Accumulated Emitted Radiance Term!

(Tulsiani)

Computational Volume Rendering: Ray Marching



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

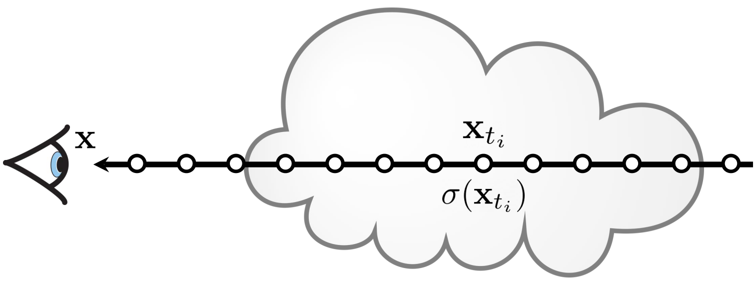
Multiplicativity

$$T(\mathbf{x},\mathbf{x}_{t_i}) = T(\mathbf{x},\mathbf{x}_{t_{i-1}})e^{-\sigma_{t_{i-1}}\Delta t}$$

Assume constant coefficient between samples

(Tulsiani)

Computational Volume Rendering: Ray Marching



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)

(Tulsiani)

Rendering Model

- Computation for a Ray

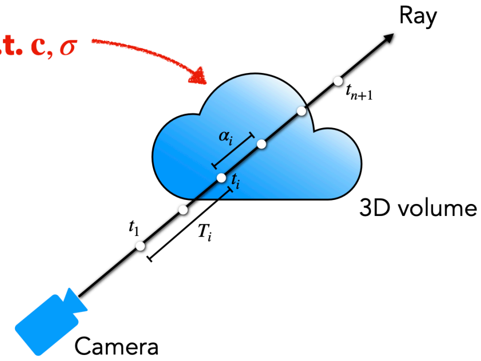
Rendering model for ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$:

$$\boxed{\mathbf{c} \approx \sum_{i=1}^{n} T_i \alpha_i \mathbf{c}_i}$$

differentiable w.r.t. $\mathbf{c}, \sigma$

colors

weights



How much light is blocked earlier along ray:

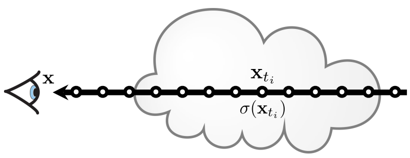
$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

How much light is contributed by ray segment i:

$$\alpha_i = 1 - \exp(-\sigma_i \delta_i)$$

(Tulsiani)

Computational Volume Rendering: A summary



$$\sigma_{t_i} \equiv \sigma(\mathbf{x}_{t_i})$$

$$L_e(\mathbf{x}_{t_i}, \omega) \longrightarrow L(\mathbf{x}, \omega)$$

If we can compute:

a) (per-point) density

b) (per-point, direction) emitted light,

we can render **any** ray through the medium

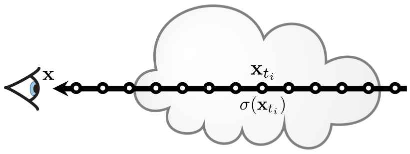
Equivalently, we can render an image from any camera viewpoint (using H*W rays)

Note: **Differentiable** process w.r.t. the **density**, **emitted light**

and also camera parameters if density, emission are differentiable functions of position, direction

(Tulsiani)

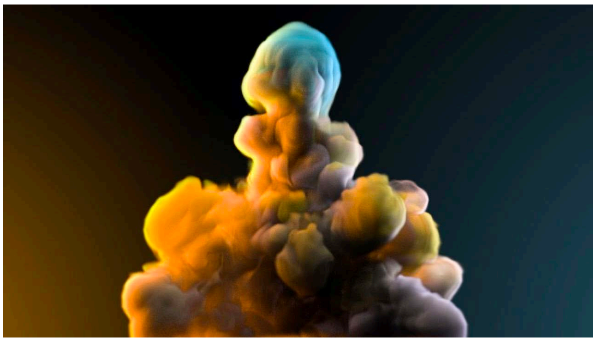
Volumes: Rendering and Representation



$\sigma_{t_i} \equiv \sigma(\mathbf{x}_{t_i})$

$L_e(\mathbf{x}_{t_i}, \omega)$ $\longrightarrow$ $L(\mathbf{x}, \omega)$

Rendering Algorithm



How to represent volumes?

(such that we can compute pointwise density and emitted light)

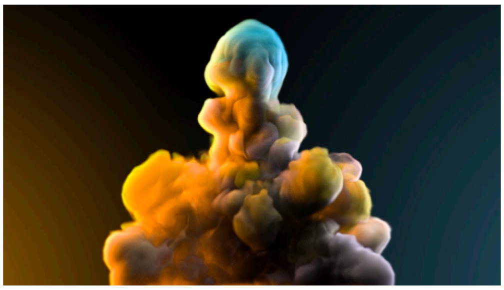
(Tulsiani)

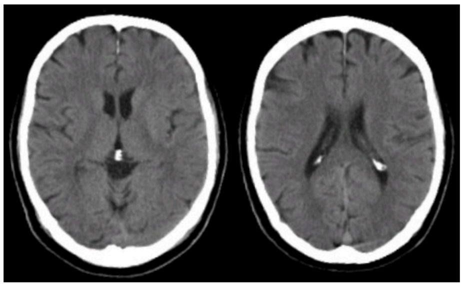
Modeling the Scene

Slides from[Takikawa et al, 2023]]

3D Scene Model

- Density Fields

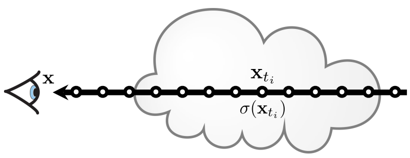




$V[x, y, z] \in \mathbf{R}^{+}$ $\qquad$ $f(\mathbf{p}) \in \mathbb{R}^{+}$

(Tulsiani)

Volumes: Rendering and Representation

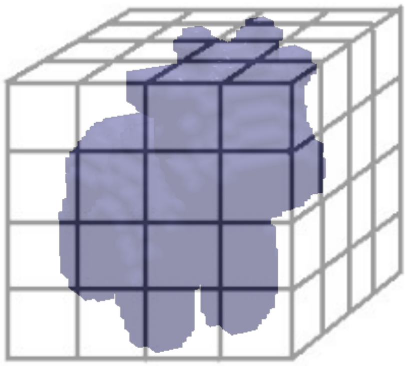


$\sigma_{t_i} \equiv \sigma(\mathbf{x}_{t_i})$

$$\longrightarrow \quad L(\mathbf{x}, \omega)$$

$L_e(\mathbf{x}_{t_i}, \omega)$

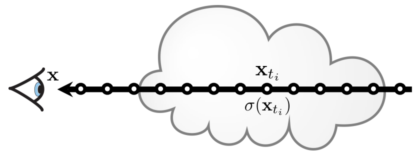
Rendering Algorithm



Option 1: A grid

(Tulsiani)

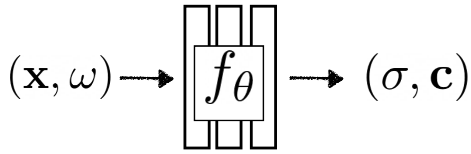
Volumes: Rendering and Representation



$\sigma_{t_i} \equiv \sigma(\mathbf{x}_{t_i})$

$L_e(\mathbf{x}_{t_i}, \omega)$ $\longrightarrow$ $L(\mathbf{x}, \omega)$

Rendering Algorithm



$\sigma \in \mathbb{R}^+ \quad \mathbf{c} \in [0, 1]^3$

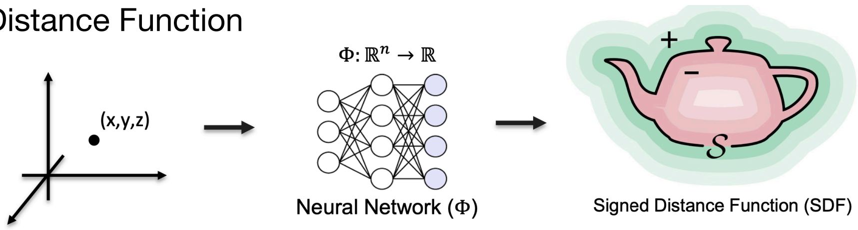
Ensure with designing MLP that density only depends on **x**

Option 2: An MLP

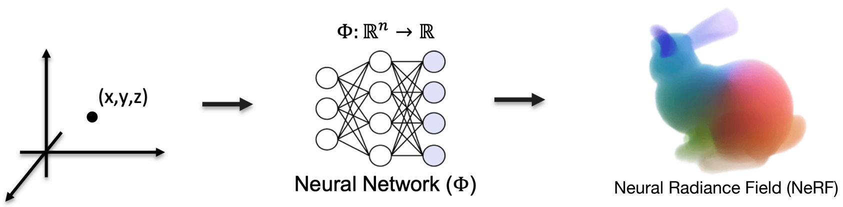
(Tulsiani)

Neural 3D Scene Models

- Signed Distance Function



- Radiance Field



(Takikawa)

NeRF

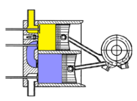
Slides from[Takikawa et al, 2023]]

Neural Radiance Fields

- Characteristics

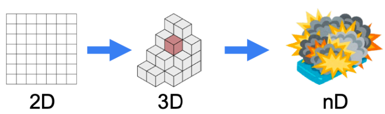
Strengths:

Compactness



Self-regularizing $\mathrm{argmin}_x \|y - F(x)\| + \cancel{\lambda P(x)}.$

Domain agnostic



Weaknesses:

Computationally expensive

Not easily editable

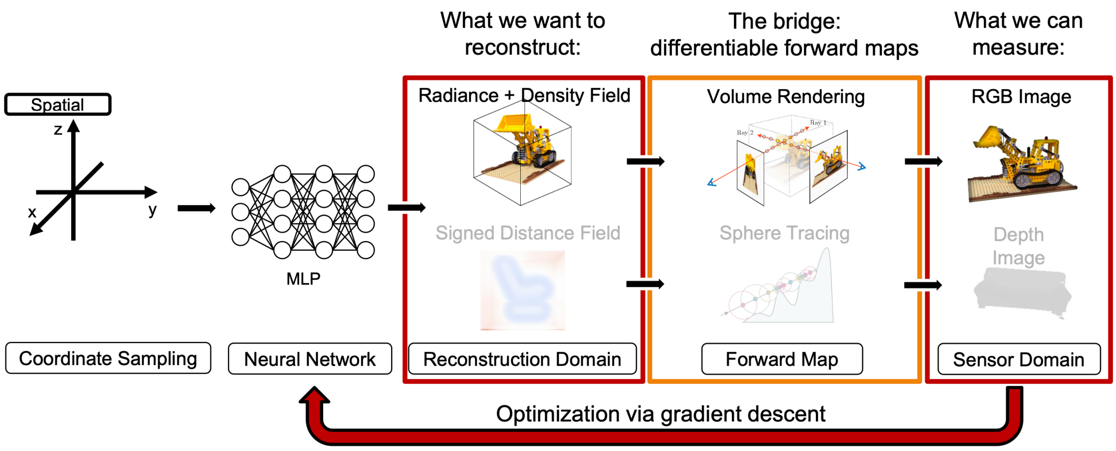
Hard to model semantics and discrete data

Lack of theoretical understanding

(Takikawa)

Analysis-by-Synthesis Reconstruction

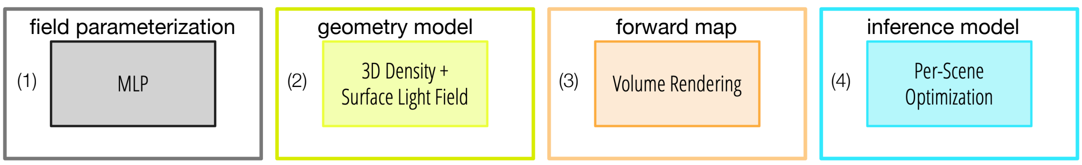
- General Framework



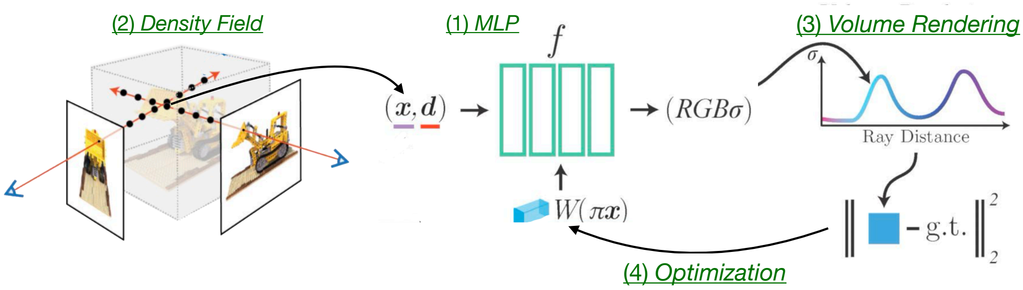
(Takikawa)

Reconstruction Method

- Elements



- Algorithm



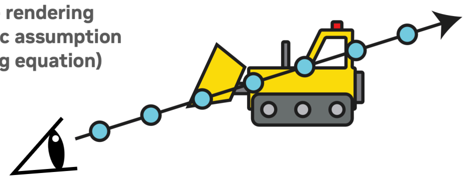
(Takikawa)

Loss Function

- Optimization

NeRF Innovation

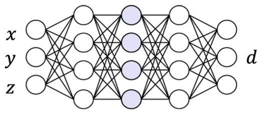
NeRF uses volume rendering
(i.e. removes the Dirac assumption
in Kajiya's rendering equation)

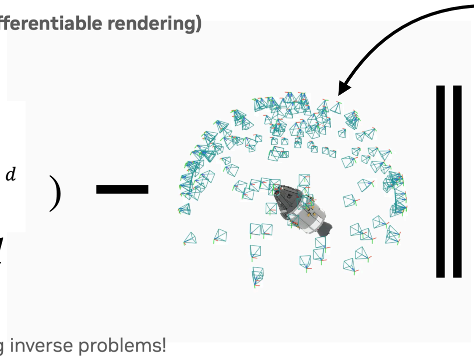


Forward Map (e.g. differentiable rendering)

All Images!

$$\min_{\theta} \left\| F(\quad f_\theta(x, y, z) = d \quad) - \quad \right\|$$

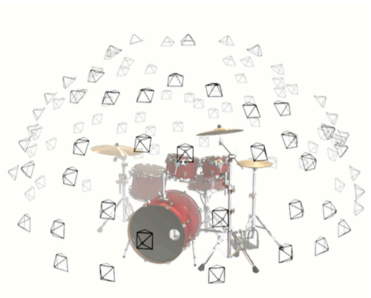




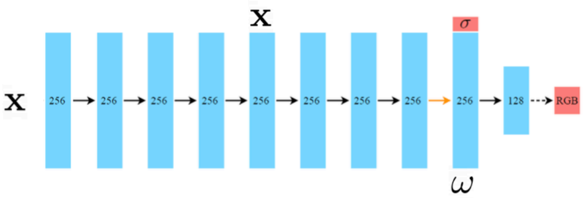
Baking via solving inverse problems!

(Takikawa)

Learning Neural Radiance Fields



1) Acquire multiple images of a scene with associated camera viewpoints



2) Design a neural network $(\sigma, \mathbf{c}) = f_\theta(\mathbf{x}, \omega)$

$$\min_{\theta} \sum_{I} \sum_{\mathbf{p}} \|\text{render}(\mathbf{p}, \pi; f_\theta) - I[\mathbf{p}]\|^2$$

3) Train with a view-synthesis loss using volume rendering

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.

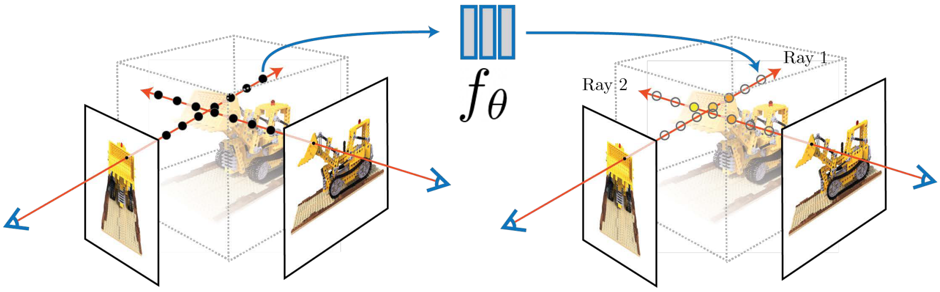
(Tulsiani)

Learning Neural Radiance Fields

(pixel, camera) -> ray

$$\min_{\theta} \sum_{I} \sum_{\mathbf{p}} \|\text{render}(\mathbf{p}, \pi; f_\theta) - I[\mathbf{p}]\|^2$$

volume rendering

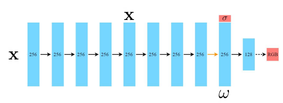


NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.

(Tulsiani)

Learning Neural Radiance Fields

Details: Spectral Projection (Fourier Feature Mapping)



(using $\boxed{\text{position encodings}}$ in input)

100 training images

Optimized Neural Net

Novel-view Renderings

A great example that 'execution matters'

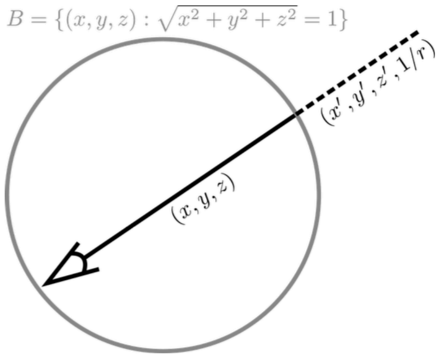
NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.

(Tulsiani)

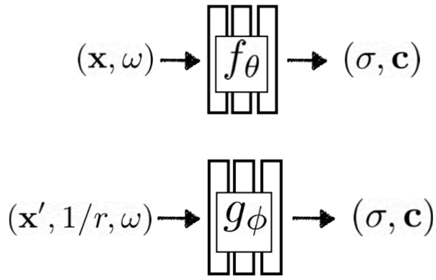
Foreground and Background Radiance

NERF++: ANALYZING AND IMPROVING NEURAL RADIANCE FIELDS

Kai Zhang Cornell Tech
Gernot Riegler Intel Labs
Noah Snavely Cornell Tech
Vladlen Koltun Intel Labs



Unbounded Scenes

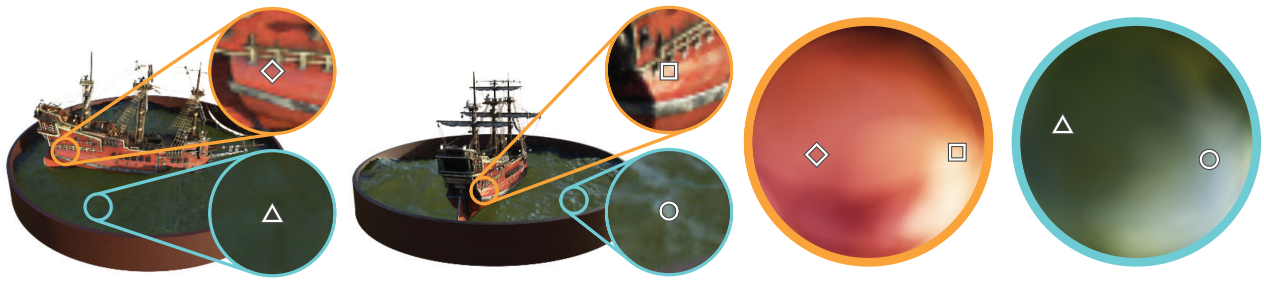


A ‘background’ NeRF uses normalized sphere coordinate and 1/r as input

Uniform sampling for ray segments inside sphere, and 1/r based sampling outside

(Tulsiani)

View-dependent Effects



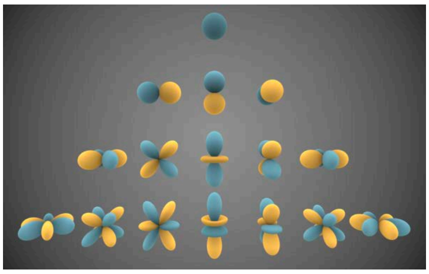
Same 3D point — different color based on viewing direction

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.

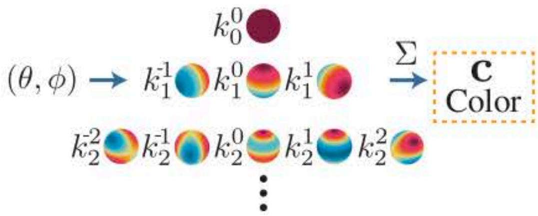
(Tulsiani)

Do we need Neural Radiance?

From NeRF to Grids & S.H.



Spherical Harmonics: A basis for scalar functions on a sphere



View-dependent color can be inferred via basis coefficients

Plenoxels: Radiance Fields without Neural Networks. Yu et. al.

(Tulsiani)