Neural Implicit Representations for 3D Reconstruction

PhD director: Marchand Eric

Supervisor: Boukhayma Adnane

PhD Student: Ouasfi Amine

Inverse Graphics

Images → Camera Poses

Assume Given

SLAM / BundleAdjustment
(OrbSLAM, COLMAP)

Reconstruct

3D Scene

Radiance

- Scene is a cloud of tiny colored particles

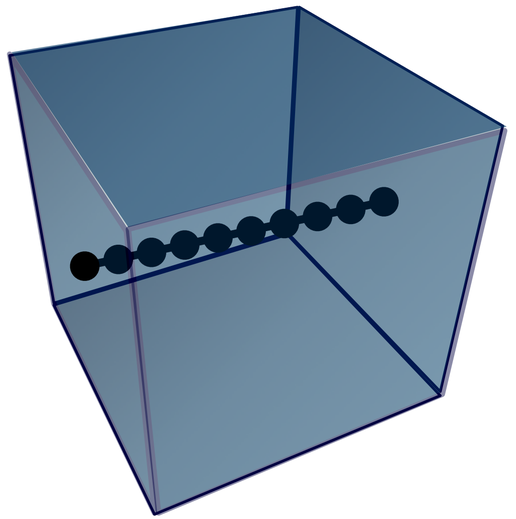
- Radiance:

Direction

$$L(x, w, \lambda) : \mathbb{R}^3 \times \mathbb{S}^2 \times \mathbb{R} \rightarrow \mathbb{R}^+$$

Position Wavelength

- Radiance along an unblocked ray is constant (energy conservation)



Tagliasacchi and Mildenhall “Volume Rendering Digest (for NeRF)” arXiv 2022

Radiance

- Scene is a cloud of tiny colored particles

The rate at which light energy is emitted from a surface in a particular direction

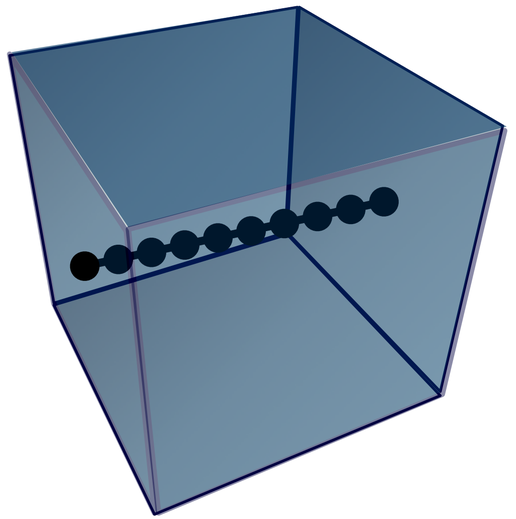
- Radiance:

Direction

$$L(x, w, \lambda) : \mathbb{R}^3 \times \mathbb{S}^2 \times \mathbb{R} \rightarrow \mathbb{R}^+$$

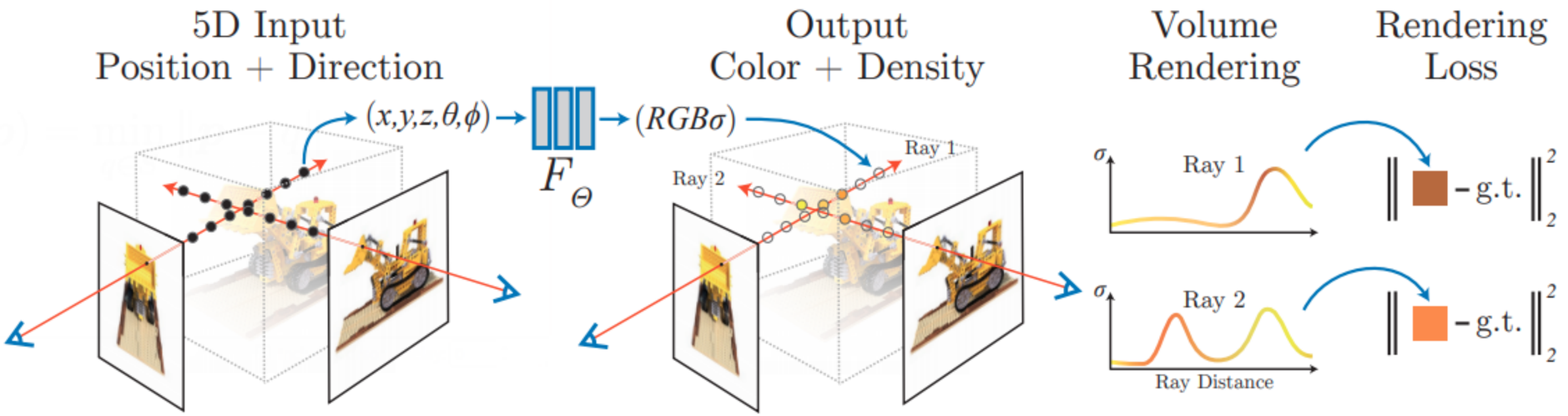
Position Wavelength

- Radiance along an unblocked ray is constant (energy conservation)

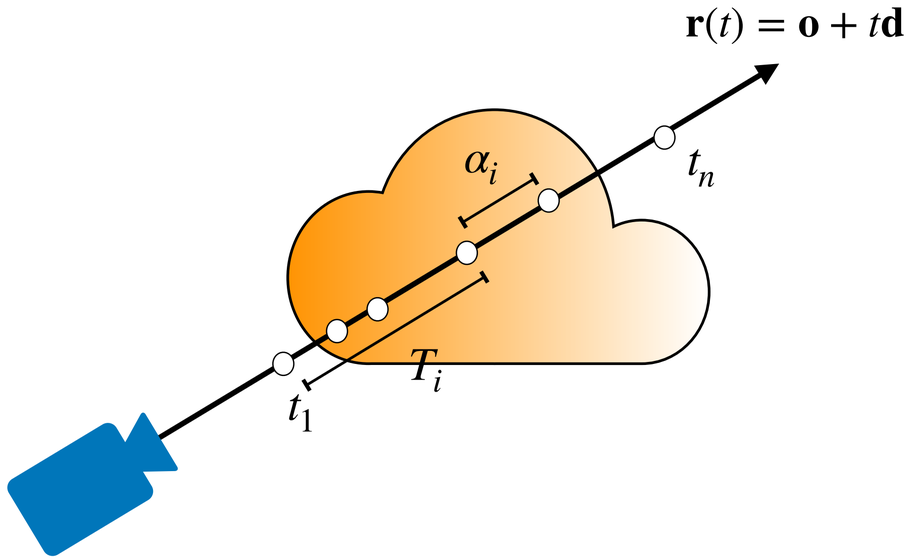


Tagliasacchi and Mildenhall “Volume Rendering Digest (for NeRF)” arXiv 2022

NERF in a nutshell

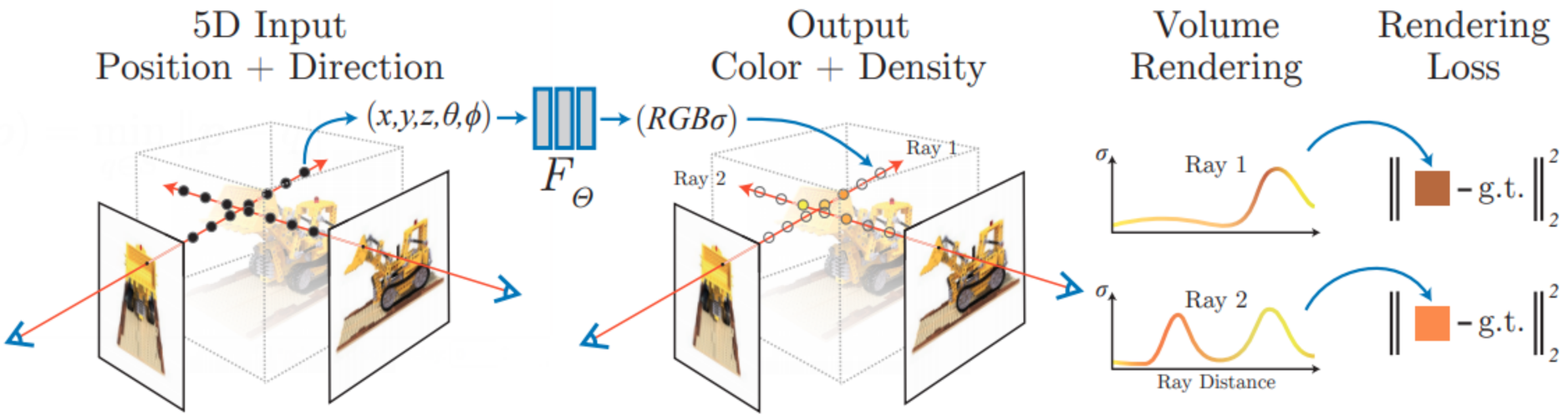


Volumetric rendering with ray tracing:



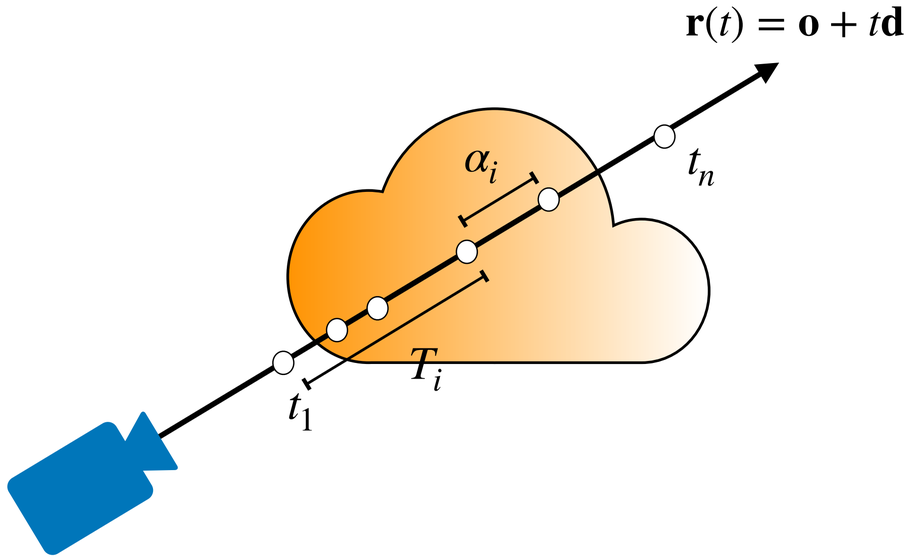
Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

NERF in a nutshell



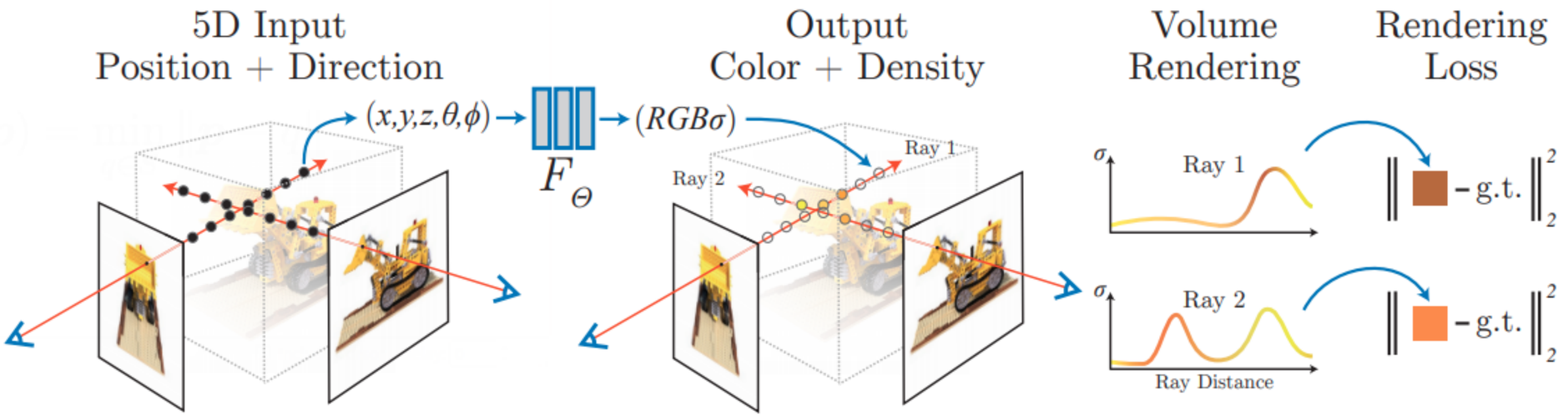
Volumetric rendering with ray tracing:

$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

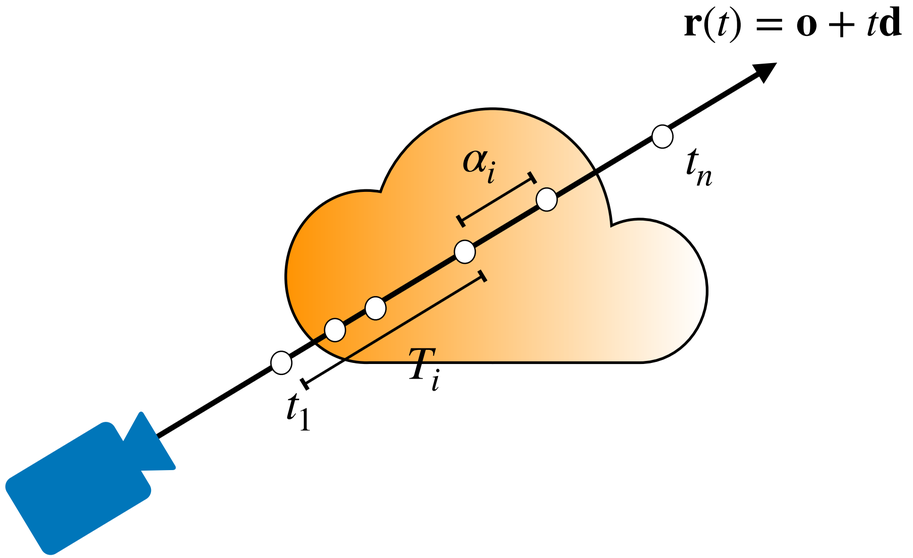
NERF in a nutshell



Volumetric rendering with ray tracing:

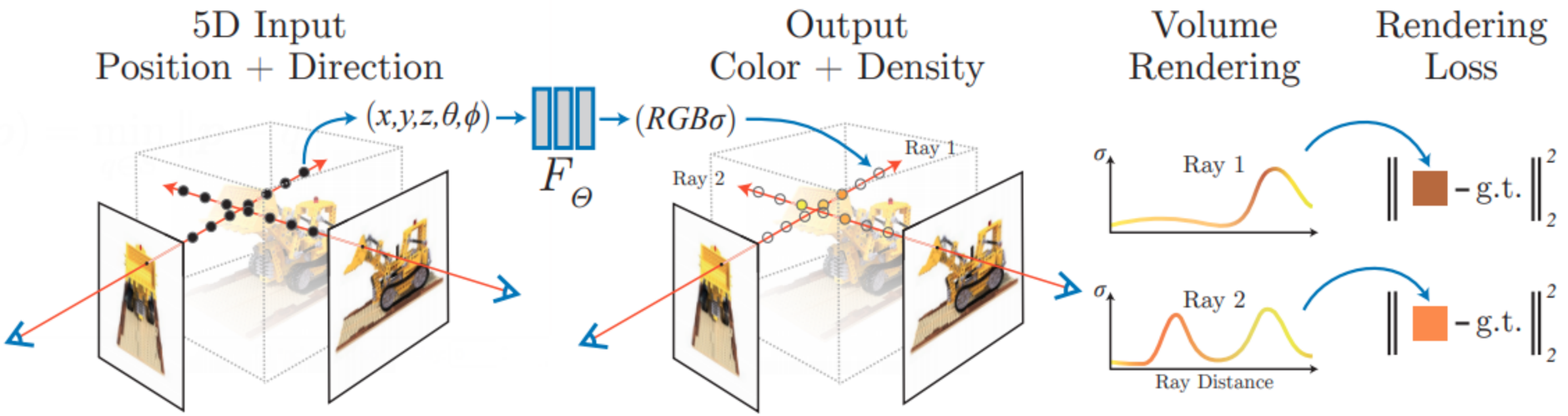
$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

$$\alpha_i = (1 - e^{-\sigma_i(t_{i+1}-t_i)})$$



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

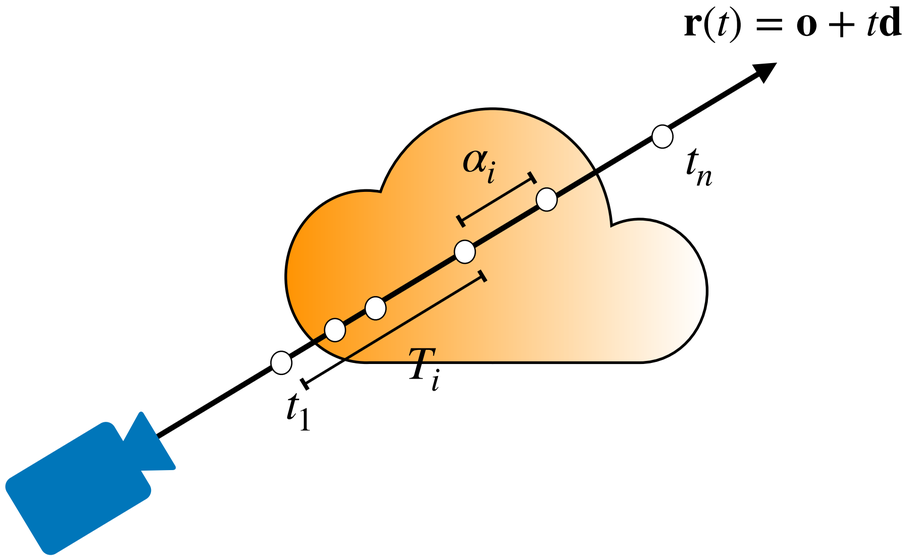
NERF in a nutshell



Volumetric rendering with ray tracing:

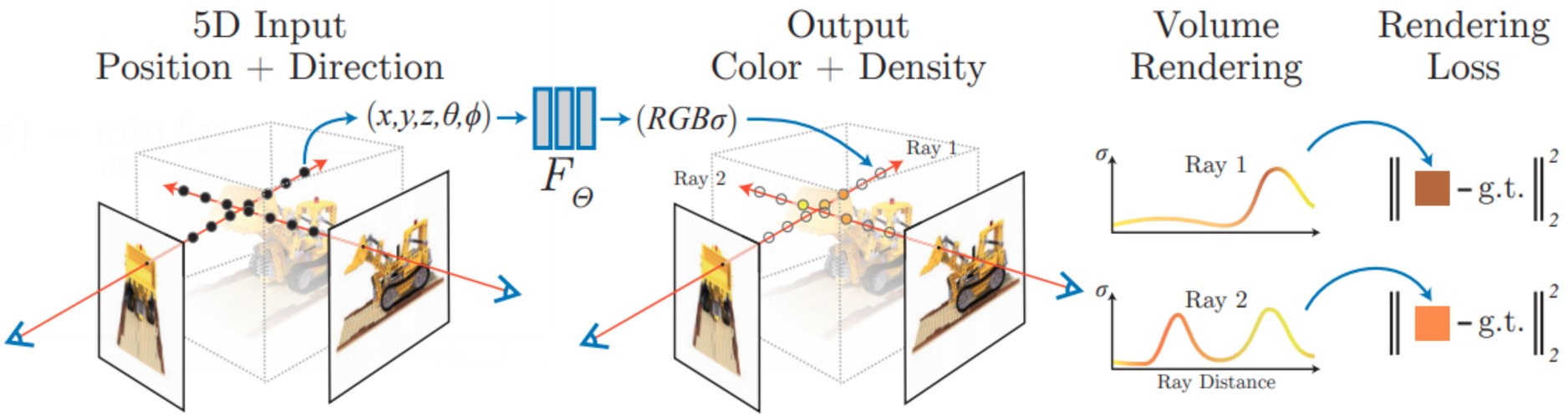
$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

$$\alpha_i = (1 - e^{-\sigma_i(t_{i+1}-t_i)})$$



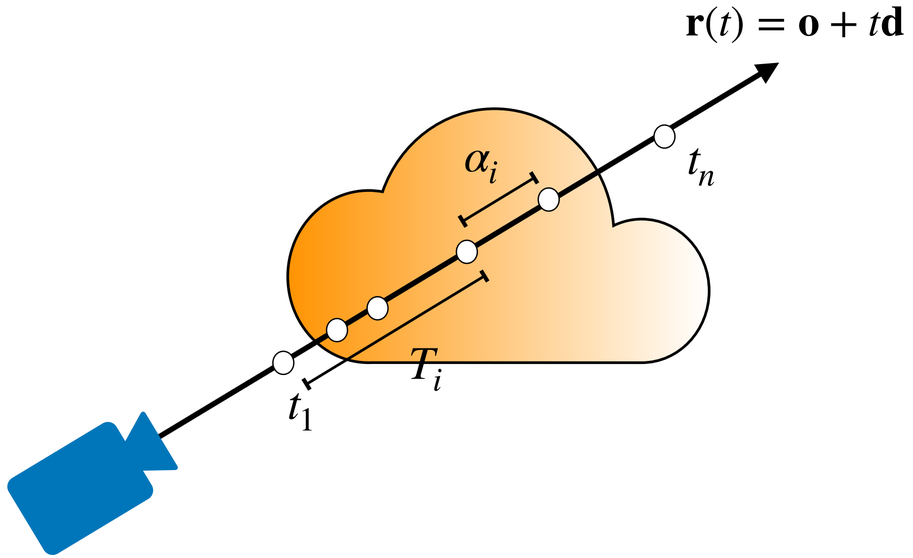
Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

NERF in a nutshell



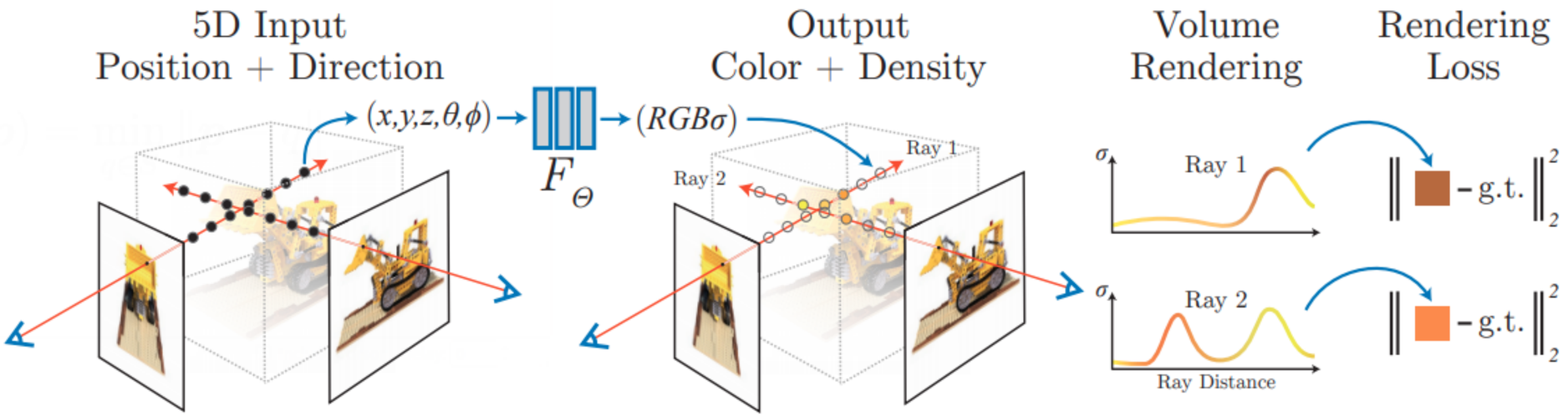
Volumetric rendering with ray tracing:

$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

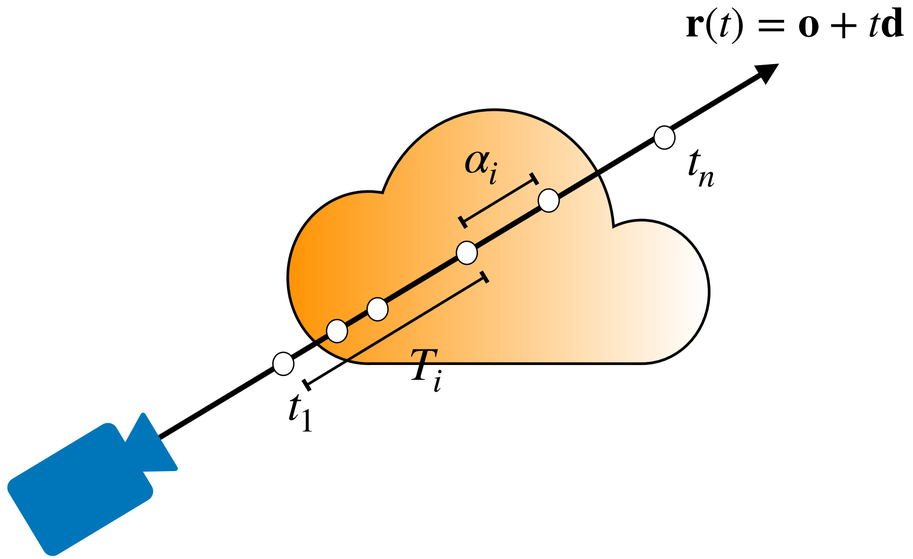
NERF in a nutshell



Volumetric rendering with ray tracing:

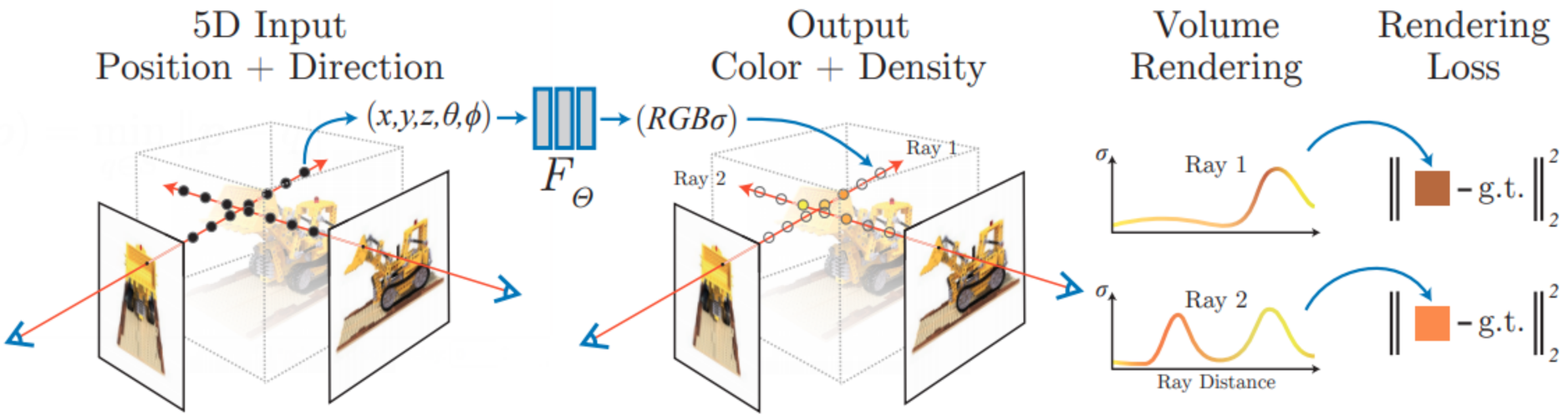
$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

$$\mathscr{L}(\theta) = \mathbb{E}_{\mathbf{C}\sim\mathbf{C}_i} \mathbb{E}_{\mathbf{r}\sim\mathbf{C}} \| \mathbf{C}(\mathbf{r},\theta) - \mathbf{C}_i(\mathbf{r}) \|_2^2$$



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

NERF in a nutshell

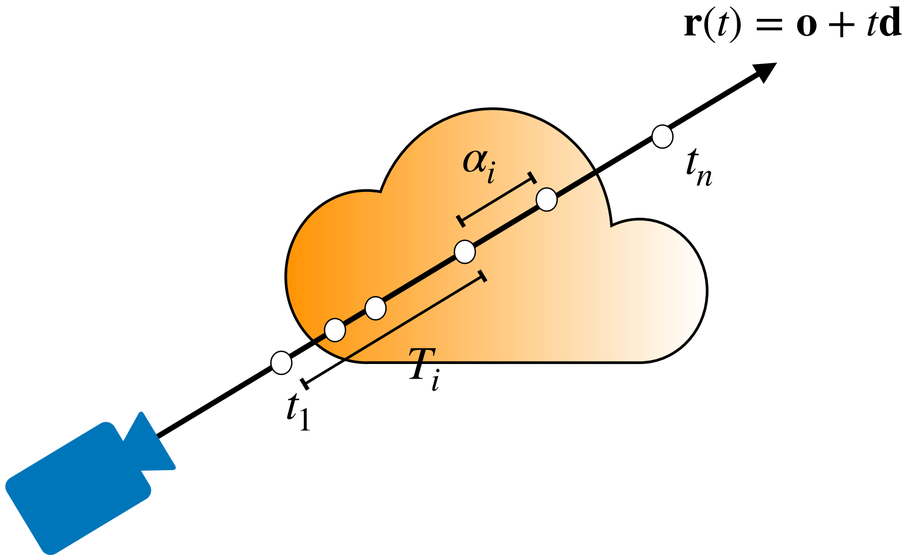


Volumetric rendering with ray tracing:

$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

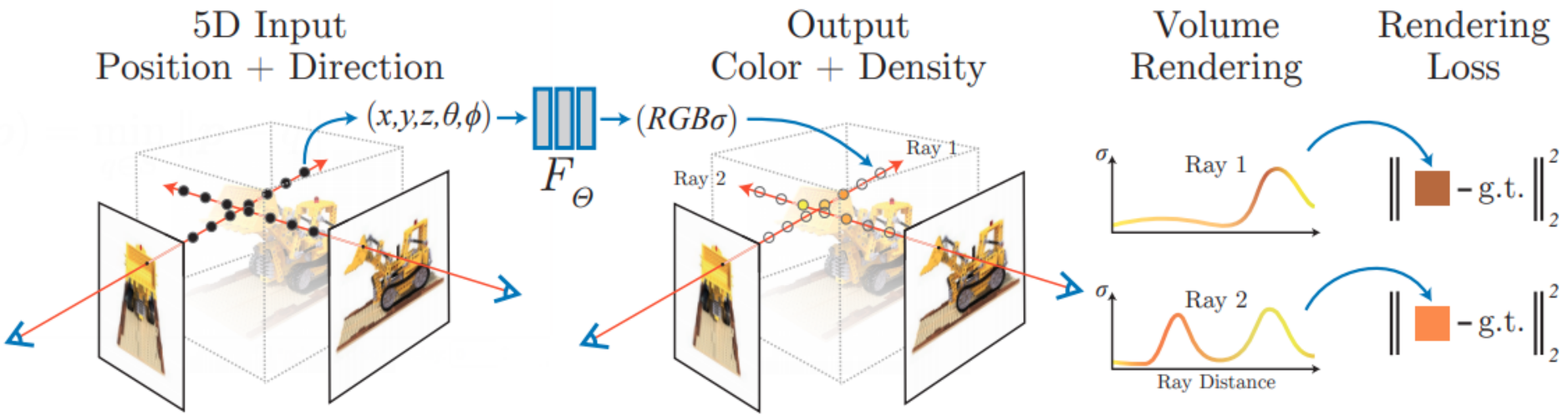
$$\mathscr{L}(\theta) = \mathbb{E}_{\mathbf{C}\sim\mathbf{C}_i} \mathbb{E}_{\mathbf{r}\sim\mathbf{C}} \left|\left| \mathbf{C}(\mathbf{r},\theta) - \mathbf{C}_i(\mathbf{r}) \right|\right|_2^2$$

Select image



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

NERF in a nutshell

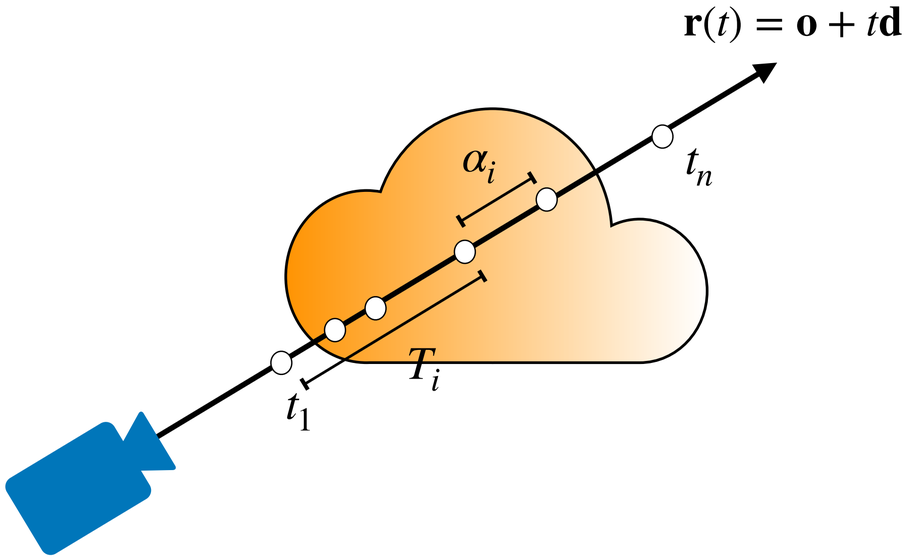


Volumetric rendering with ray tracing:

$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

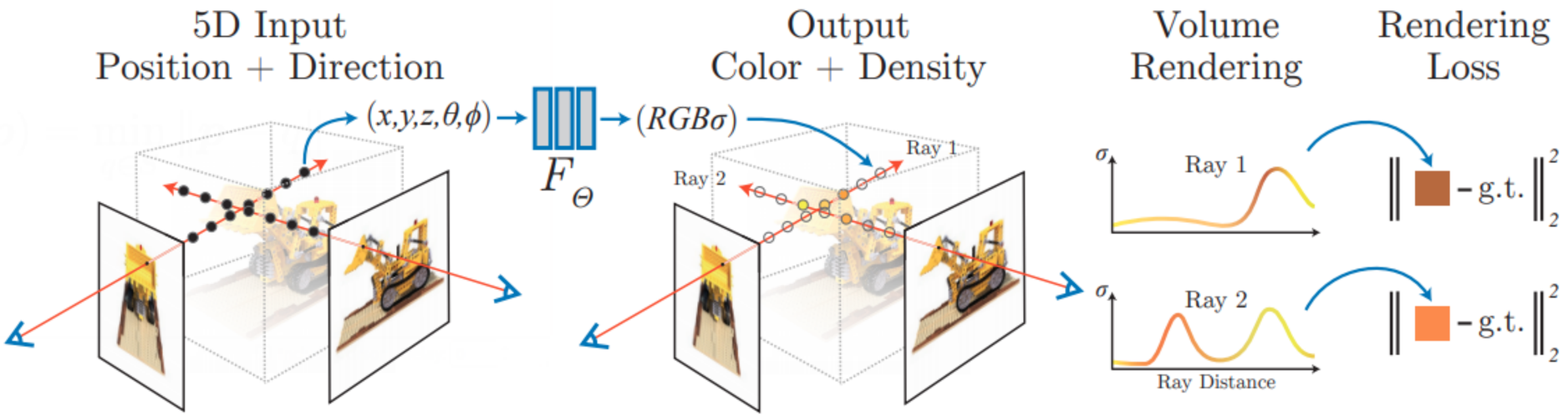
$$\mathscr{L}(\theta) = \mathbb{E}_{\mathbf{C}\sim\mathbf{C}_i} \mathbb{E}_{\mathbf{r}\sim\mathbf{C}} \left|\left| \mathbf{C}(\mathbf{r},\theta) - \mathbf{C}_i(\mathbf{r}) \right|\right|_2^2$$

Select image — Select pixel/ray



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

NERF in a nutshell

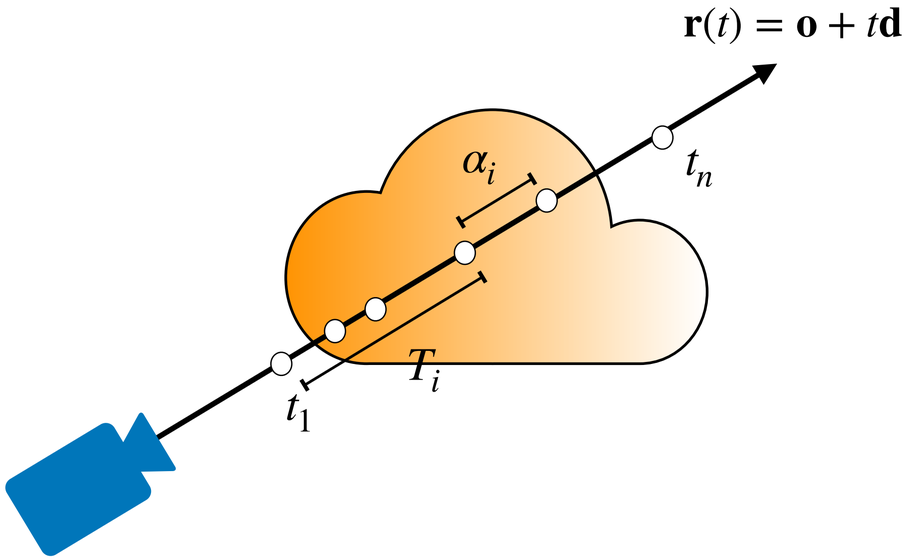


Volumetric rendering with ray tracing:

$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

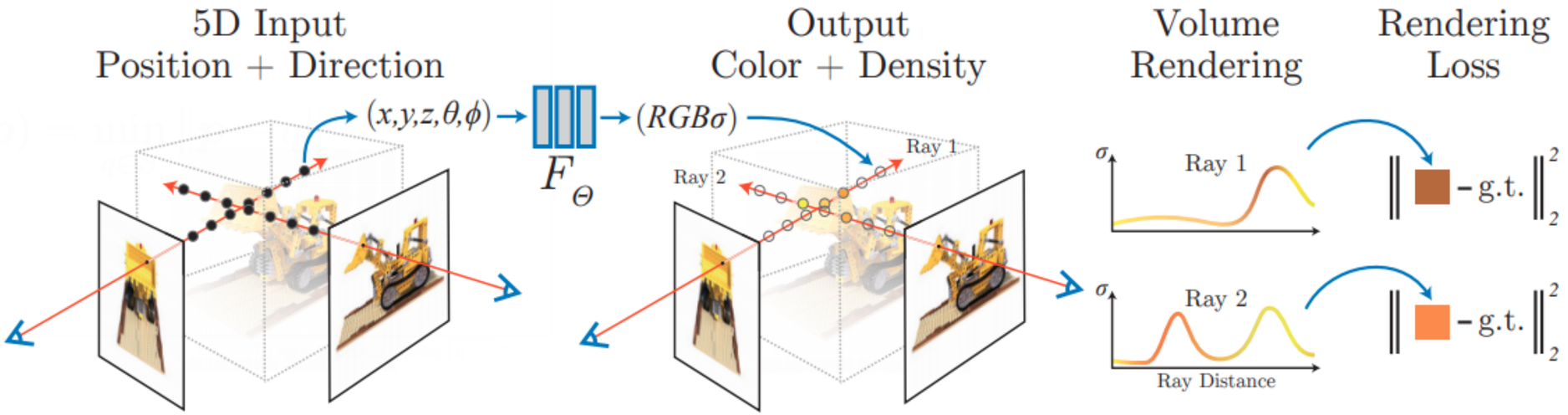
$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{C}\sim\mathbf{C}_i} \mathbb{E}_{\mathbf{r}\sim\mathbf{C}} \left|\left| \mathbf{C}(\mathbf{r},\theta) - \mathbf{C}_i(\mathbf{r}) \right|\right|_2^2$$

Select image — Select pixel/ray — predicted color



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

NERF in a nutshell

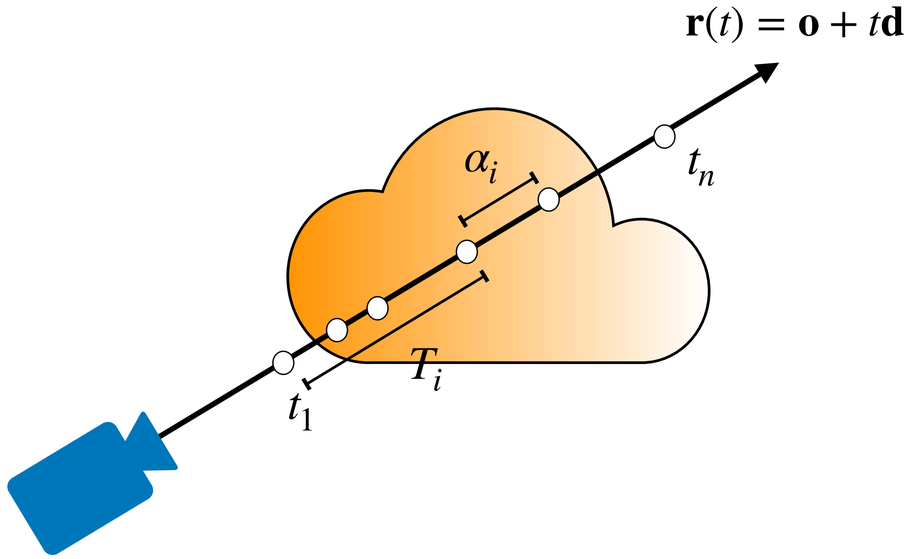


Volumetric rendering with ray tracing:

$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

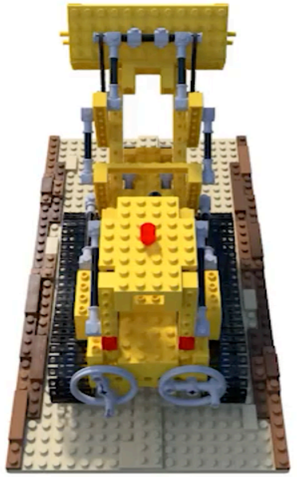
$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{C}\sim\mathbf{C}_i} \mathbb{E}_{\mathbf{r}\sim\mathbf{C}} \left|\left| \mathbf{C}(\mathbf{r},\theta) - \mathbf{C}_i(\mathbf{r}) \right|\right|_2^2$$

Select image — Select pixel/ray — predicted color — ground truth



Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

Results

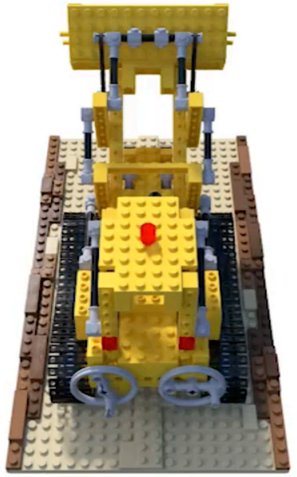







Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

Results





Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

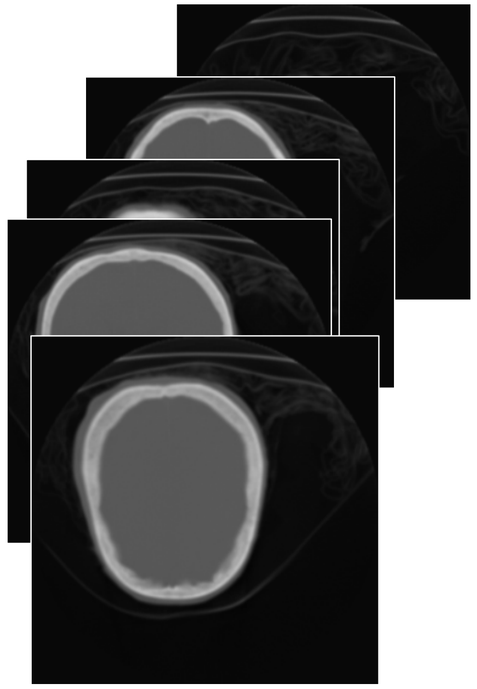
Volume rendering

Engel et al. Real-Time Volume Graphics. EUROGRAPHICS 2006 Tutorial

Volume rendering

Volume Data

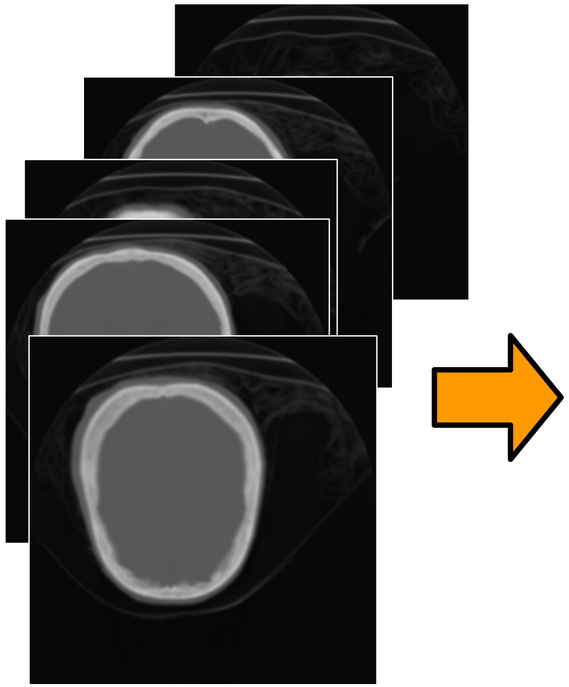
3D Rendering



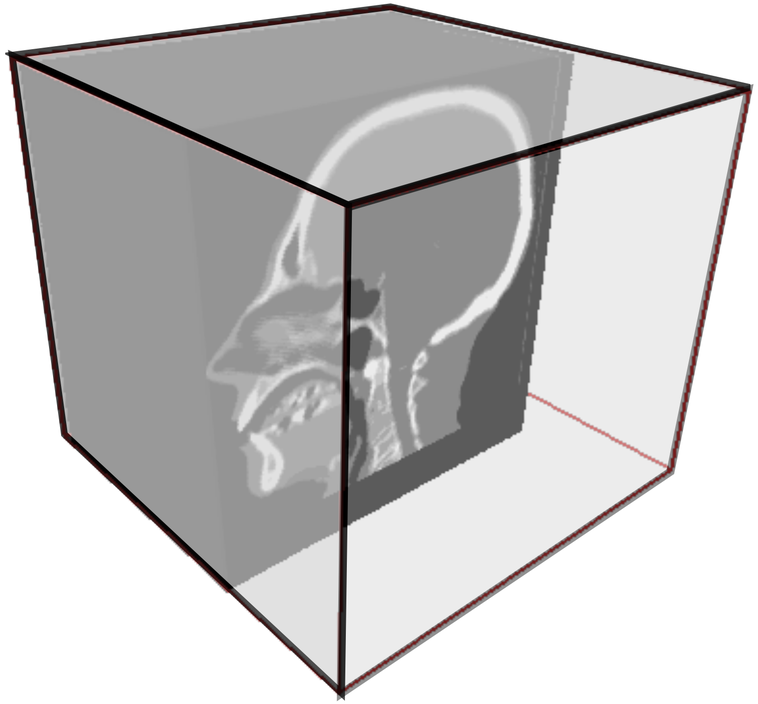
Engel et al. Real-Time Volume Graphics. EUROGRAPHICS 2006 Tutorial

Volume rendering

Volume Data



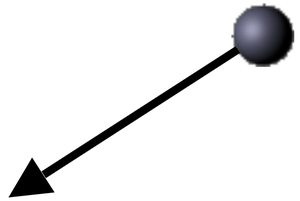
3D Rendering



Reconstruction

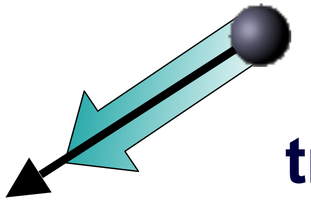
Engel et al. Real-Time Volume Graphics. EUROGRAPHICS 2006 Tutorial

Physical Model of Radiative Transfer



Physical Model of Radiative Transfer

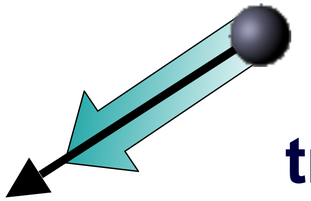
Increase



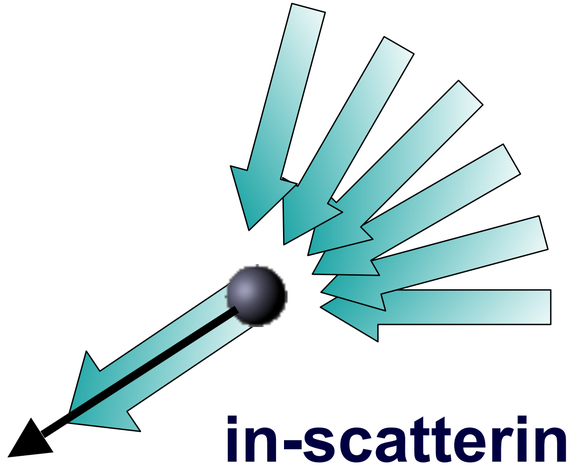
true emission

Physical Model of Radiative Transfer

Increase

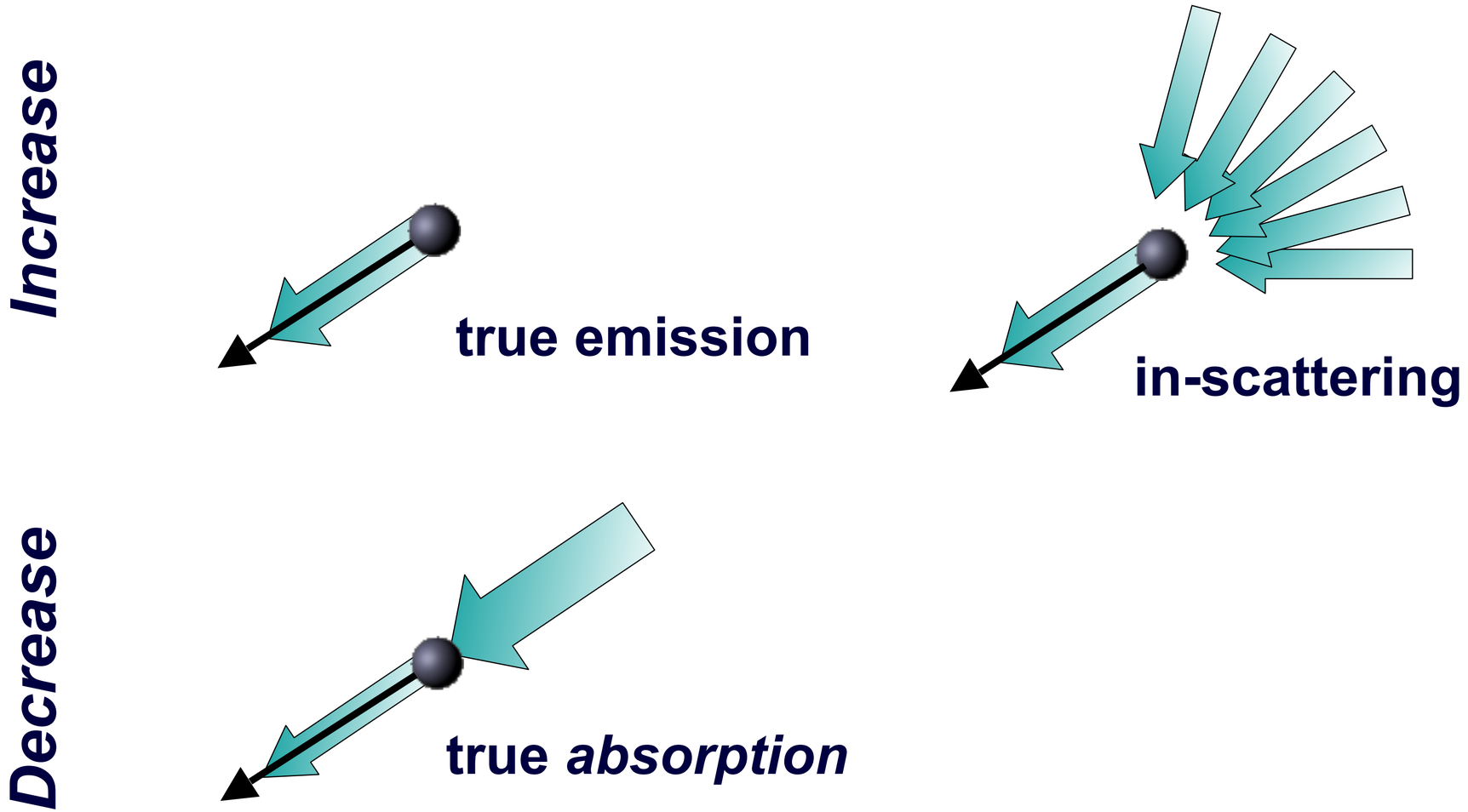


true emission



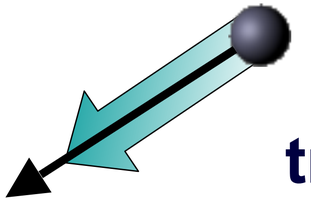
in-scattering

Physical Model of Radiative Transfer

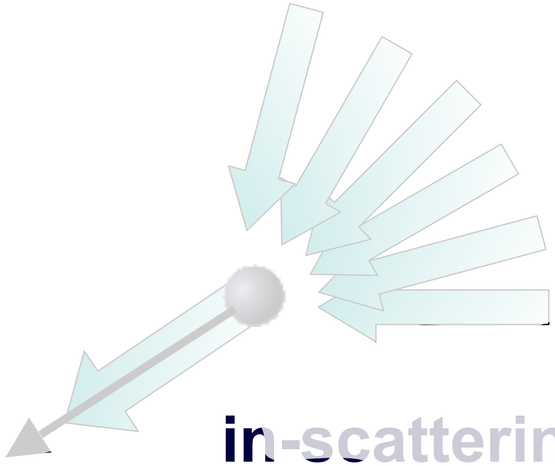


Physical Model of Radiative Transfer

Increase

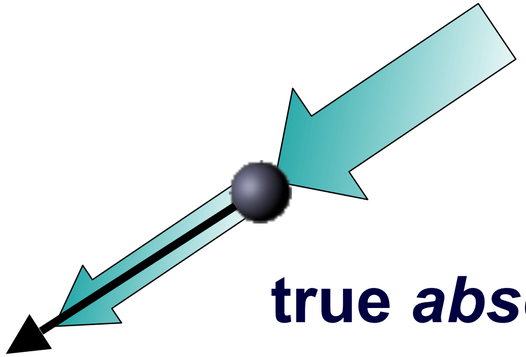


true emission

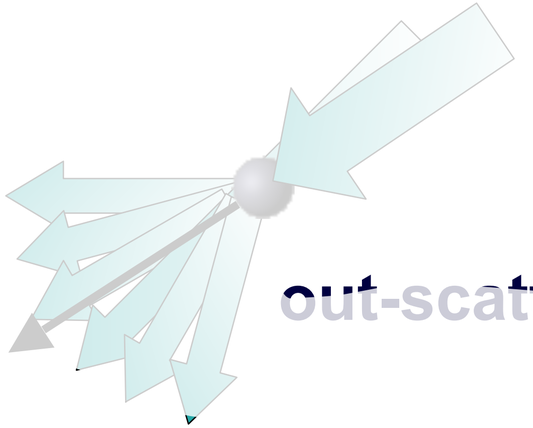


in-scattering

Decrease



true *absorption*

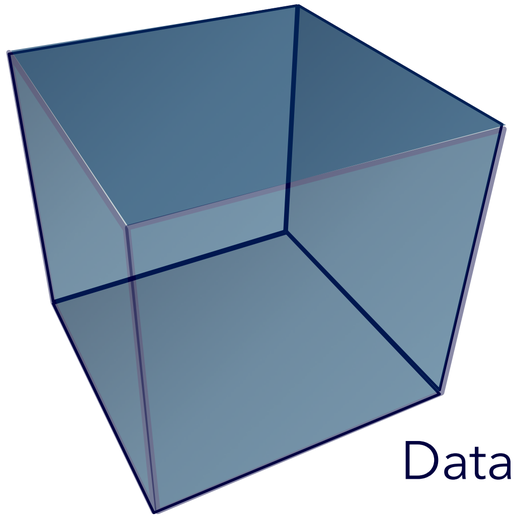


out-scattering

Ray Casting

Engel et al. Real-Time Volume Graphics. EUROGRAPHICS 2006 Tutorial

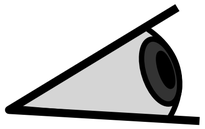
Ray Casting

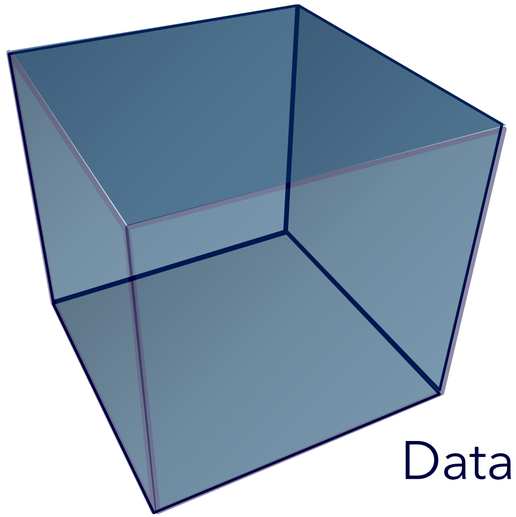


Data Set

Ray Casting

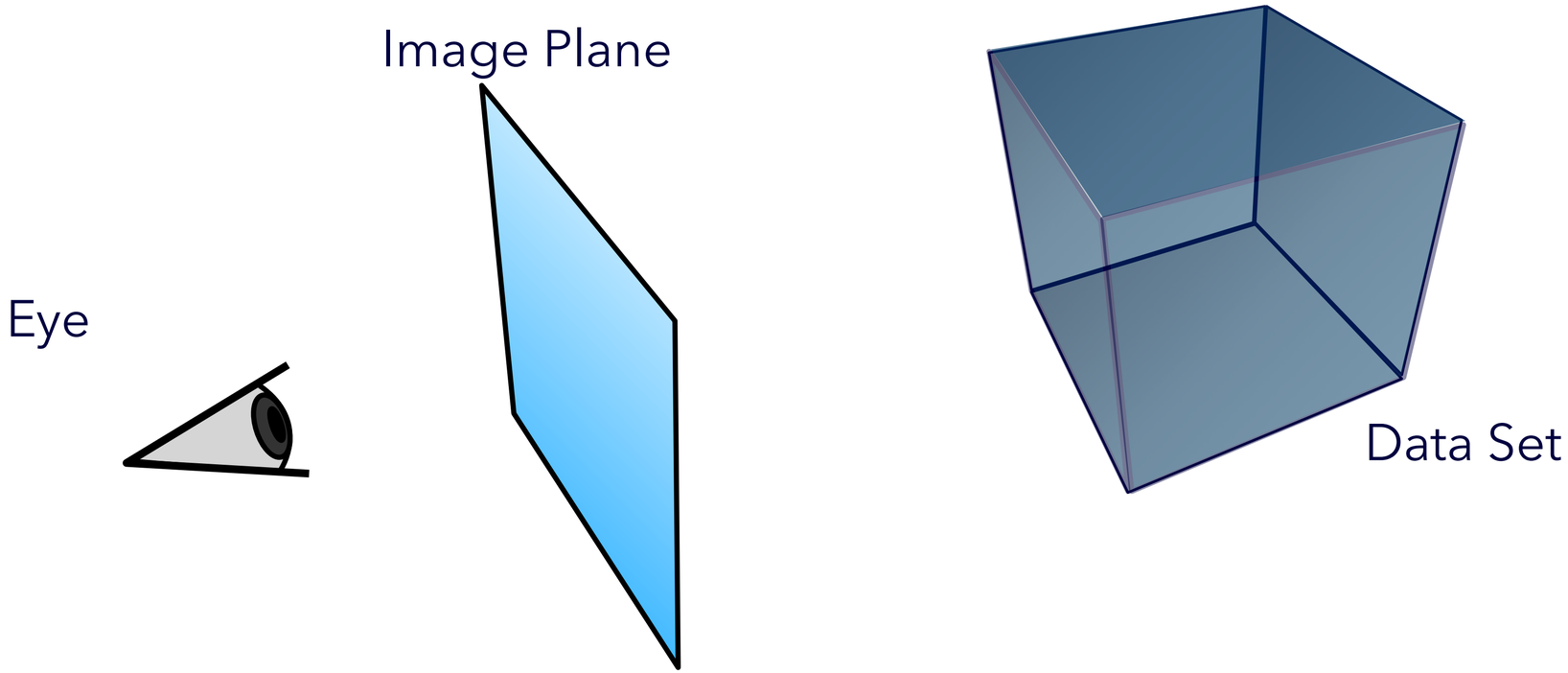
Eye





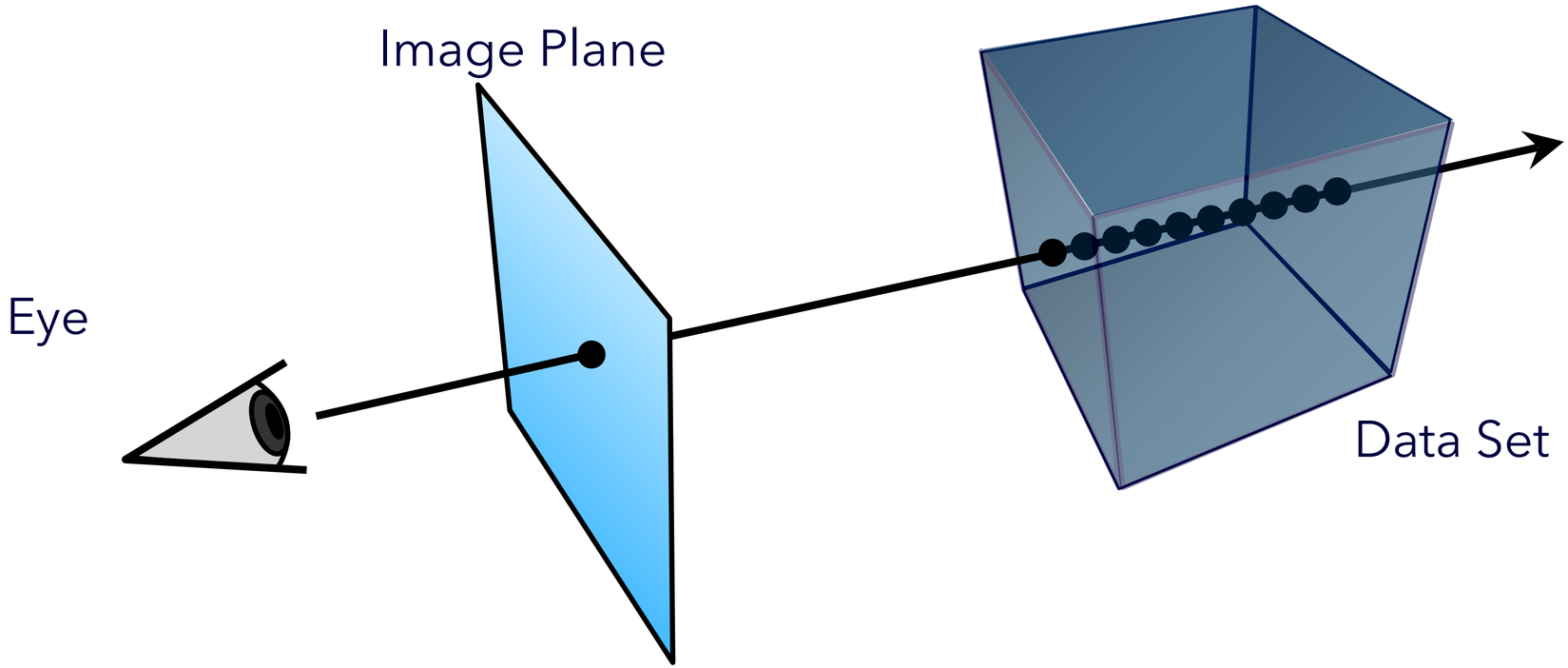
Data Set

Ray Casting



Engel et al. Real-Time Volume Graphics. EUROGRAPHICS 2006 Tutorial

Ray Casting



Engel et al. Real-Time Volume Graphics. EUROGRAPHICS 2006 Tutorial

Ray integration

Physical model: emission and absorption, no scattering

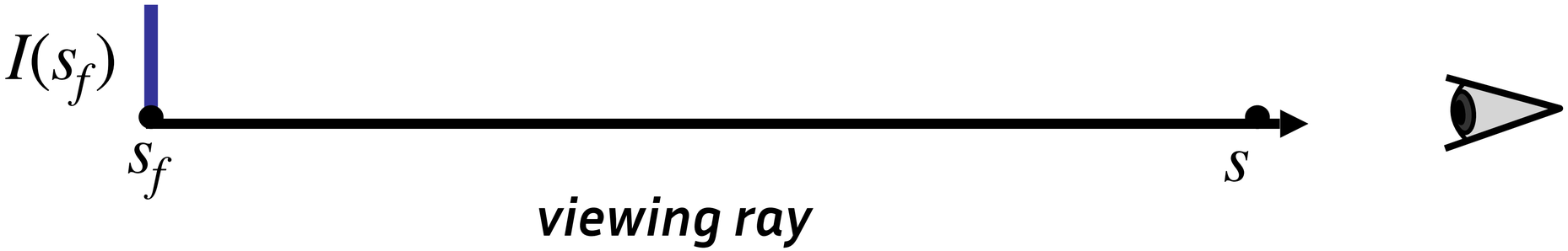
$I(s_f)$

s_f

s

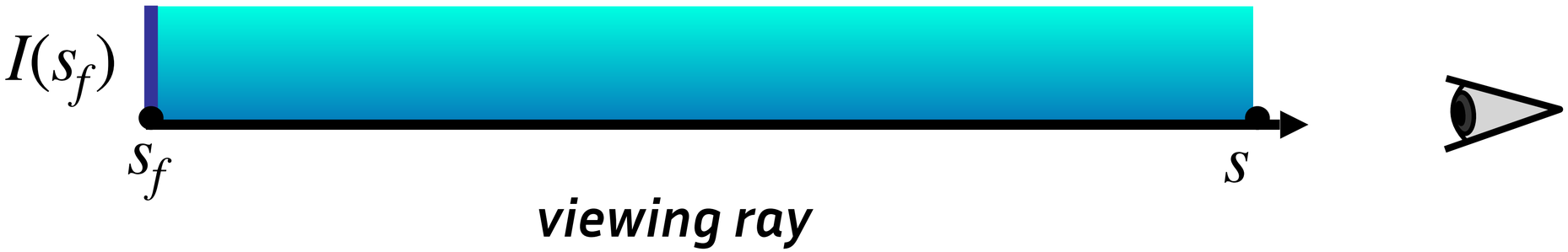
Ray integration

Physical model: emission and absorption, no scattering



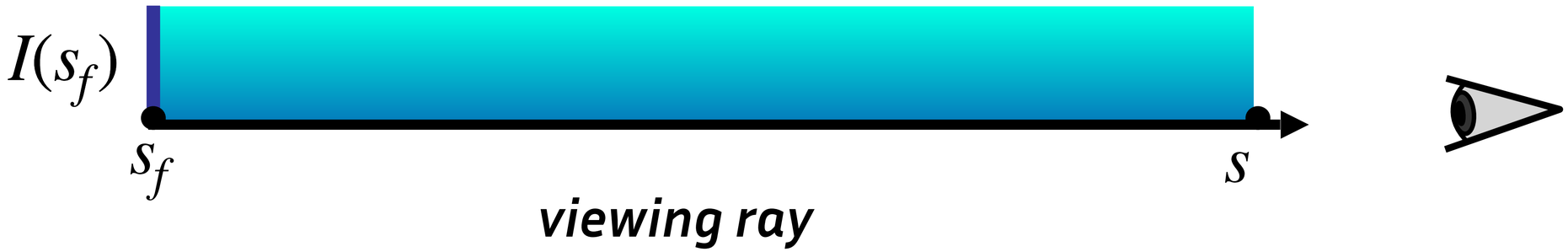
Ray integration

Physical model: emission and absorption, no scattering



Ray integration

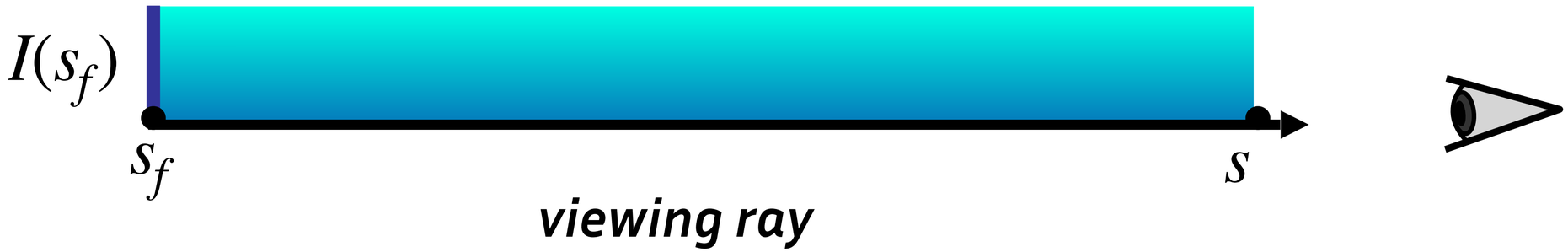
Physical model: emission and absorption, no scattering



$$C(s) =$$

Ray integration

Physical model: emission and absorption, no scattering



$$C(s) = I(s_f)$$

Ray integration

Physical model: emission and absorption, no scattering

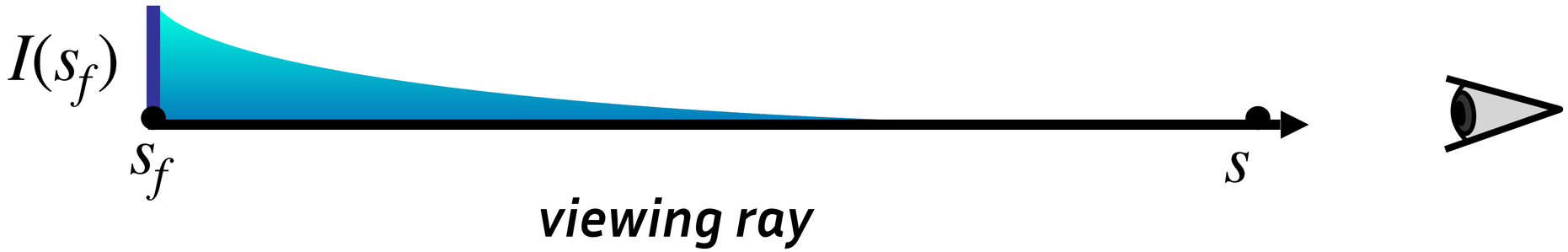
Initial intensity at s_f

$$C(s) = I(s_f)$$

Without absorption all the initial radiant energy would reach the point ***s***

Ray integration

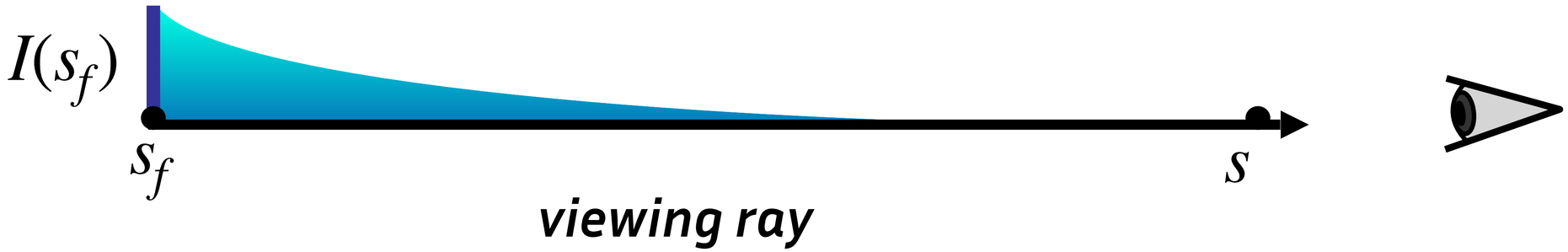
Physical model: emission and absorption, no scattering



$$C(s) = I(s_f)$$

Ray integration

Physical model: emission and absorption, no scattering



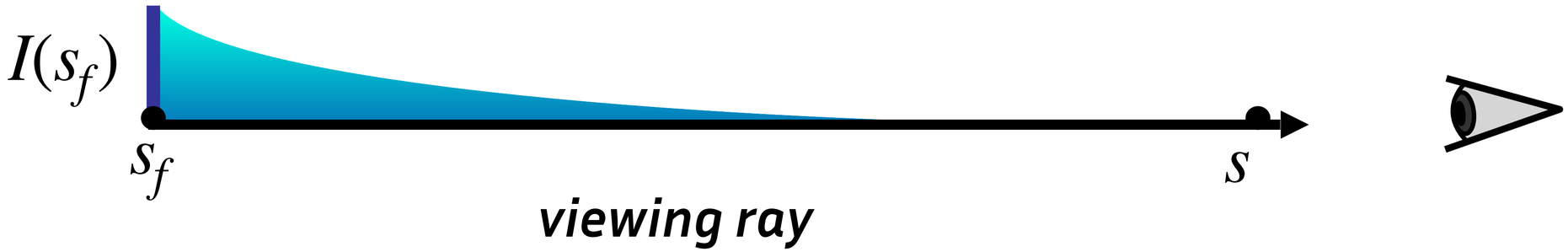
$$C(s) = I(s_f)\ e^{-\tau(s_f,s)}$$

Extinction τ
Absorption σ

$$\tau(s_1, s_2) = \int_{s_1}^{s_2} \sigma(s)ds$$

Ray integration

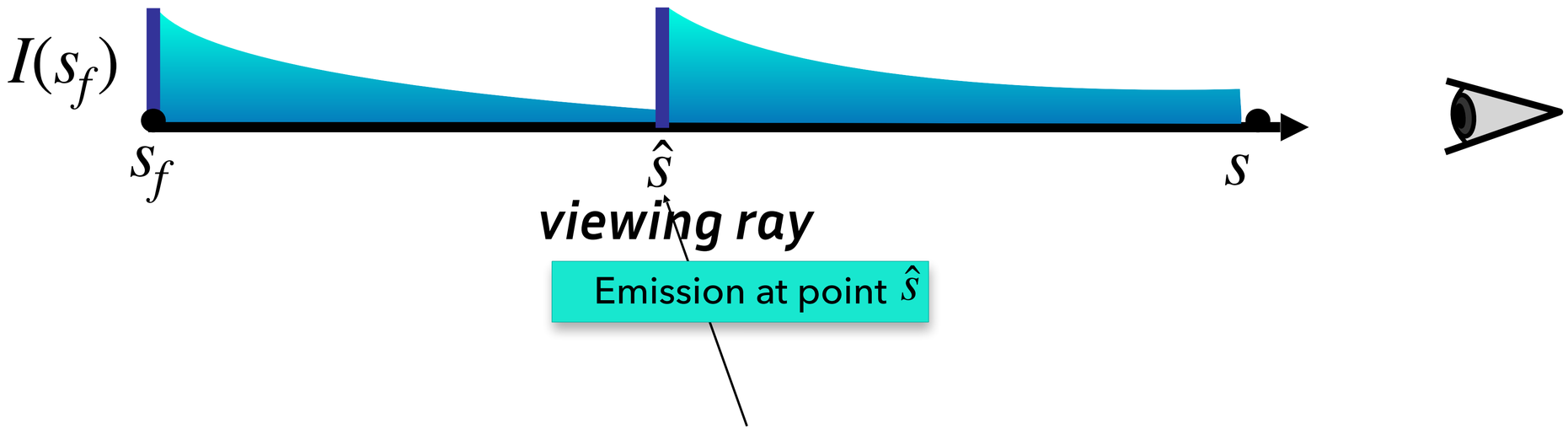
Physical model: emission and absorption, no scattering



$$C(s) = I(s_f) \; e^{-\tau(s_f,s)}$$

Ray integration

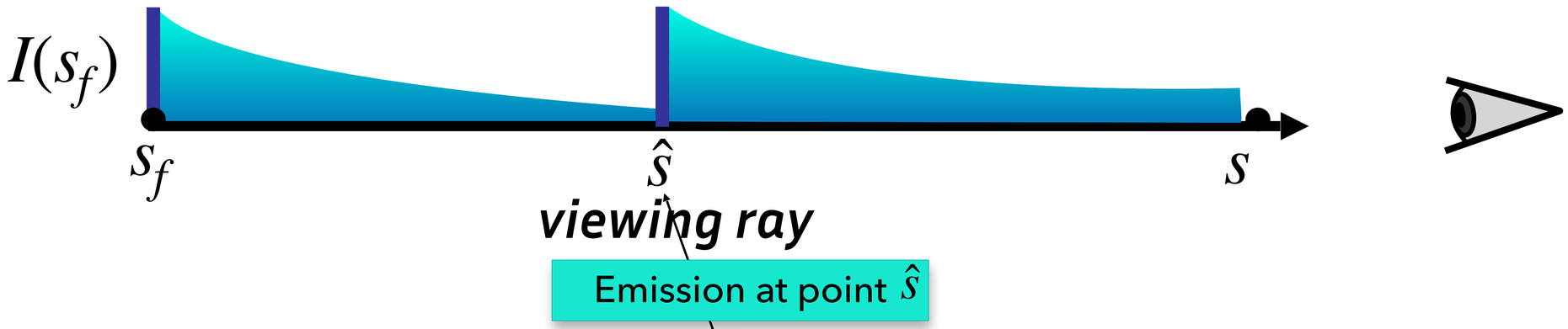
Physical model: emission and absorption, no scattering



$$C(s) = I(s_f) \; e^{-\tau(s_f,s)}$$

Ray integration

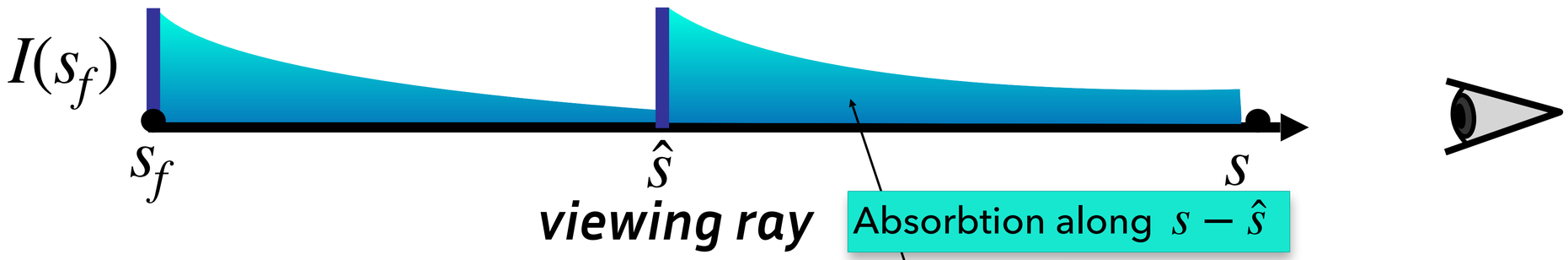
Physical model: emission and absorption, no scattering



$$C(s) = I(s_f)\ e^{-\tau(s_f,s)} + C(\hat{s})\sigma(\hat{s})$$

Ray integration

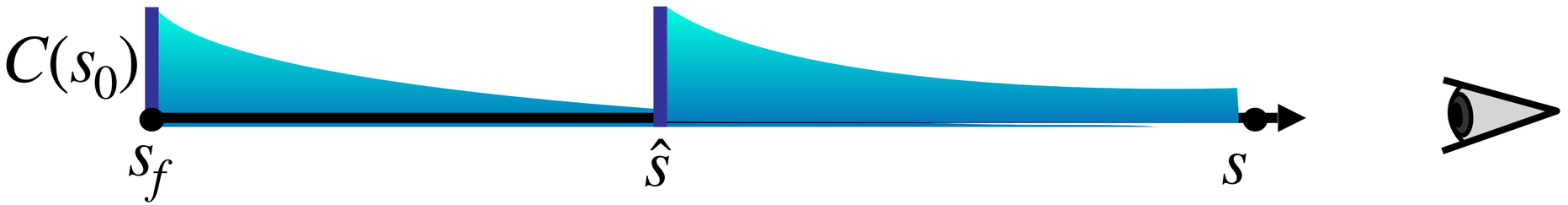
Physical model: emission and absorption, no scattering



$$C(s) = I(s_f)\ e^{-\tau(s_f,s)} + C(\hat{s})\sigma(\hat{s})\ e^{-\tau(s,\hat{s})}$$

Ray integration

Physical model: emission and absorption, no scattering



viewing ray

Extinction τ
Absorption σ

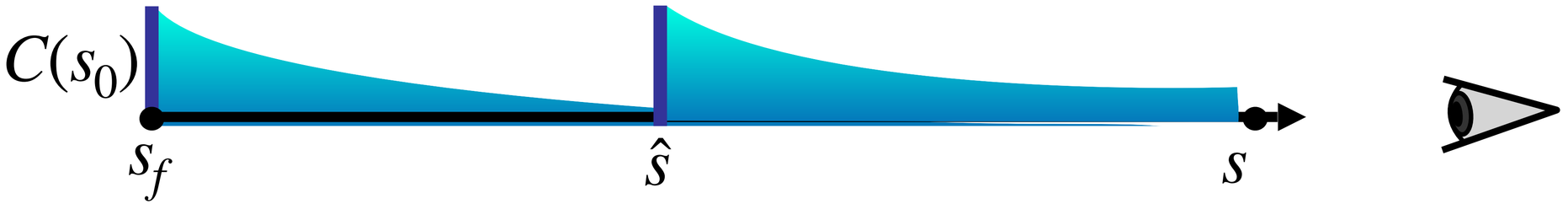
$$\tau(s_1, s_2) = \int_{s_1}^{s_2} \sigma(s) ds$$

All *the points along the ray emit additional radiant energy*

$$C(s) = I(s_f)\ e^{-\tau(s_f,s)} + \int_{s}^{s_f} C(\hat{s})\sigma(\hat{s})\ e^{-\tau(s,\hat{s})} d\hat{s}$$

Ray integration

Physical model: emission and absorption, no scattering



viewing ray

Extinction τ
Absorption σ

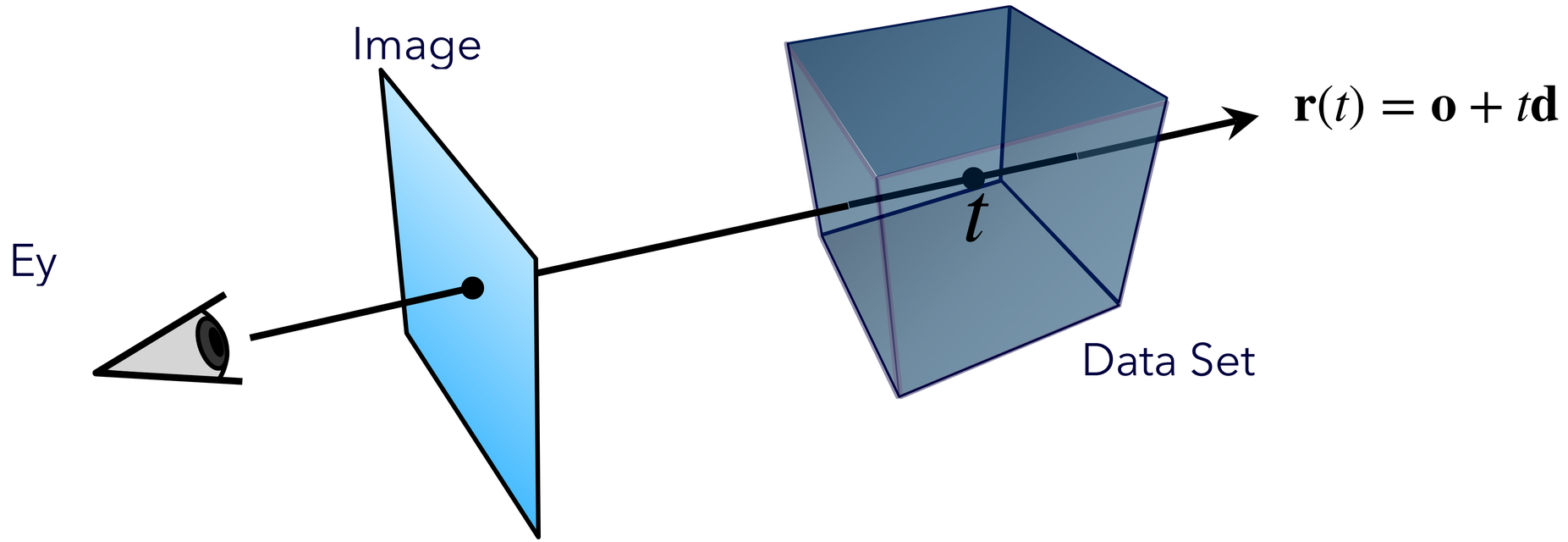
$$\tau(s_1, s_2) = \int_{s_1}^{s_2} \sigma(s) ds$$

All *the points along the ray emit additional radiant energy*

$$C(s) = I(s_f)\ e^{-\tau(s_f,s)} + \int_{s}^{s_f} C(\hat{s})\sigma(\hat{s})\ e^{-\tau(s,\hat{s})} d\hat{s}$$

Volumetric Density

- The ray stops at t if the radiant energy after t is absorbed before reaching the viewer/camera
- Probability that ray stops in a small interval around t is $\sigma(t)dt$

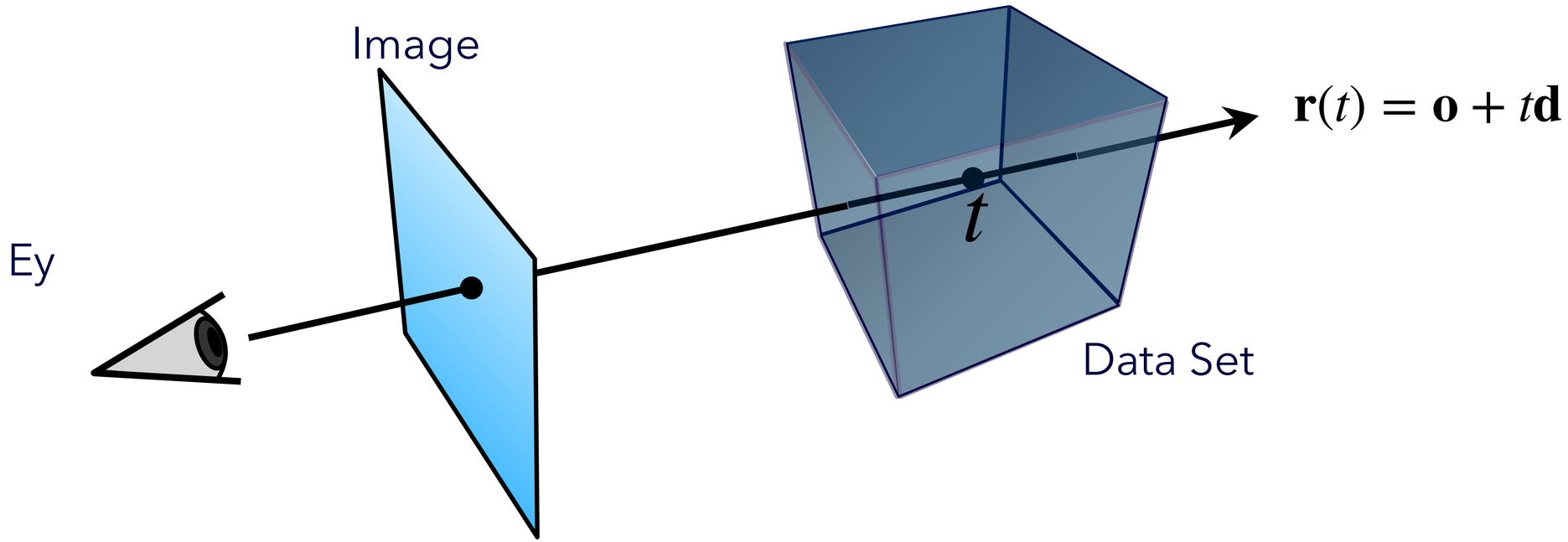


Tagliasacchi and Mildenhall “Volume Rendering Digest (for NeRF)” arXiv 2022

Transmittance

Should we retrieve the color $\mathbf{c}(t)$? ...only if it's not absorbed

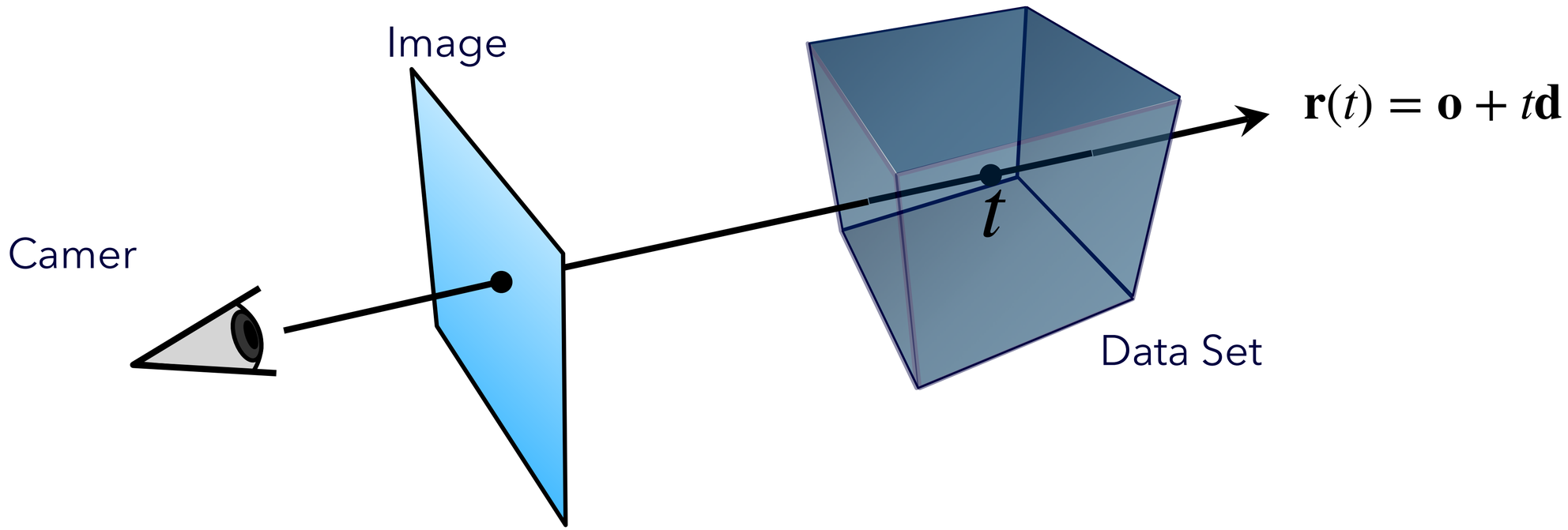
Transmittance $T(t)$ is the probability that the ray doesn't stop in the **[0,t)** range



Tagliasacchi and Mildenhall "Volume Rendering Digest (for NeRF)" arXiv 2022

Transmittance

$$T(t + dt) = T(t)(1 - \sigma(t)dt)$$

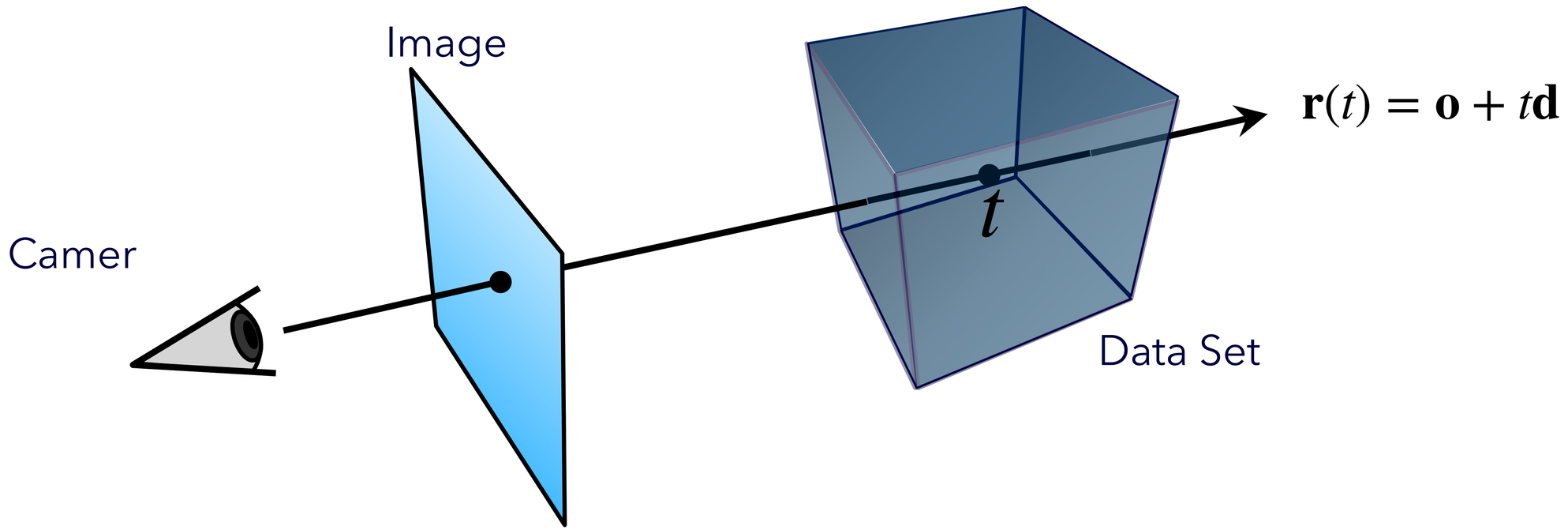


Tagliasacchi and Mildenhall “Volume Rendering Digest (for NeRF)” arXiv 2022

Transmittance

$$T(t + dt) = T(t)(1 - \sigma(t)dt)$$

The ray doesn't stop in the **[0, t+dt)** range



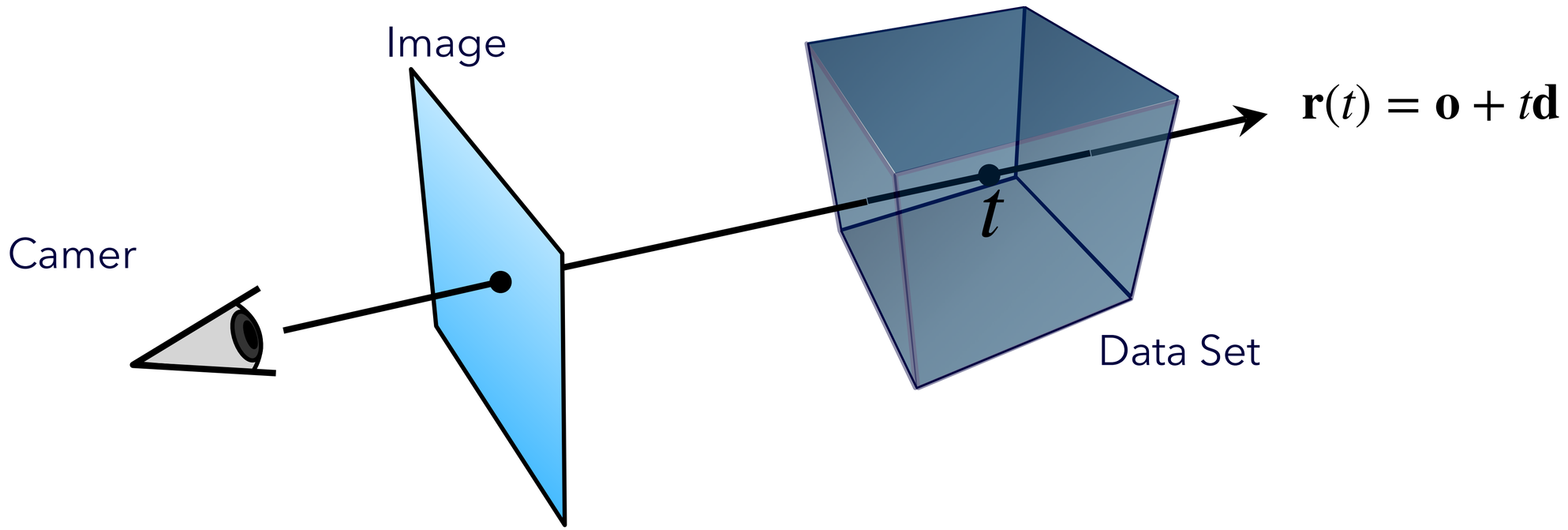
Tagliasacchi and Mildenhall "Volume Rendering Digest (for NeRF)" arXiv 2022

Transmittance

$$T(t + dt) = T(t)(1 - \sigma(t)dt)$$

The ray doesn't stop in the **[0, t+dt)** range

The ray doesn't stop in the **[0, t)** range



Tagliasacchi and Mildenhall "Volume Rendering Digest (for NeRF)" arXiv 2022

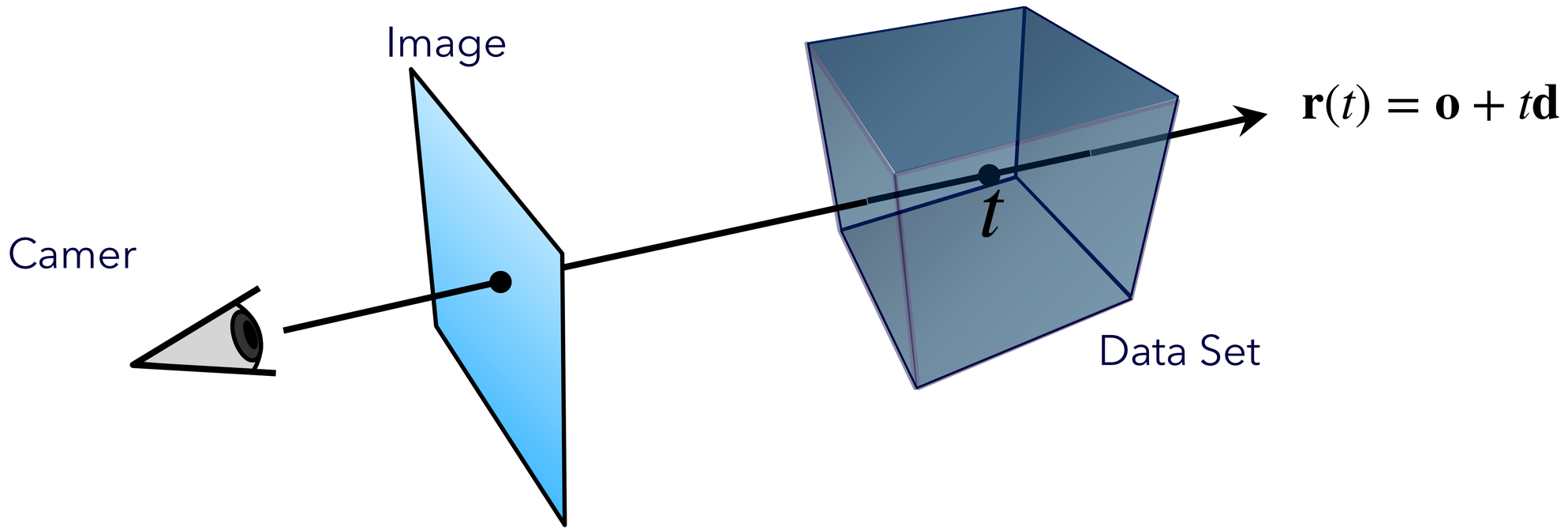
Transmittance

$$T(t + dt) = T(t)(1 - \sigma(t)dt)$$

The ray doesn't stop in the **[0, t+dt)** range

The ray doesn't stop in the **[0, t)** range

The ray doesn't stop in a small interval around **t**



Tagliasacchi and Mildenhall "Volume Rendering Digest (for NeRF)" arXiv 2022

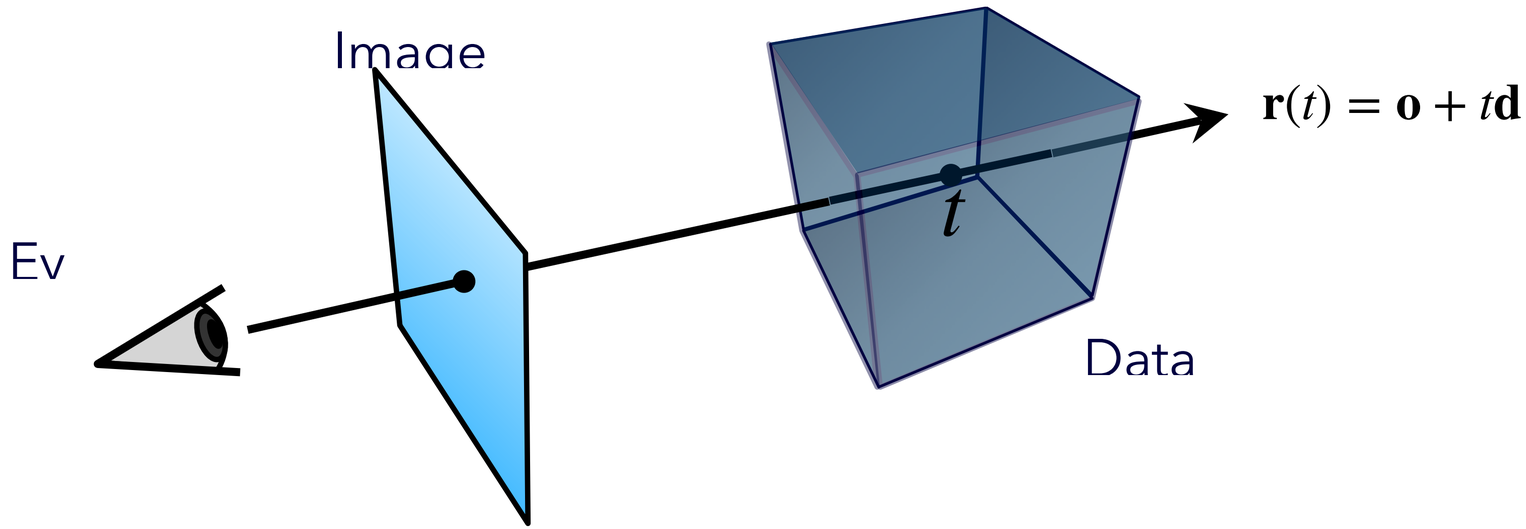
How is Transmittance related to density ?

$$T(t + dt) = T(t)(1 - \sigma(t)dt)$$

Integration

The ray doesn't stop between **a** and **b**

$$T(a \rightarrow b) = e^{-\int_a^b \sigma(t)dt}$$



Tagliasacchi and Mildenhall "Volume Rendering Digest (for NeRF)" arXiv 2022

The probability that the ray stops at t

- $1 - T(t)$ can be seen as cumulative distribution function (CDF)

$$T(t) = exp(-\int_0^t \sigma(t)dt)$$

- $(1 - T(t))' = T(t)\sigma(t)$ is a probability density function

https://en.wikipedia.org/wiki/Cumulative_distribution_function **and** Tagliasacchi and Mildenhall “Volume Rendering Digest (for NeRF)” arXiv 2022

Volumetric Rendering

- $p(t) = (1 - T(t))' = T(t)\sigma(t)$ is a probability density function
- Volumetric rendering can be seen as the expected emitted color $c(t)$ under $t \sim p(t)$

$$C(r) = \int_{t_n}^{t_f} \mathbf{c}(\mathbf{r}(t), \mathbf{d})\sigma(t)e^{-\tau(\hat{s},s)}dt = \int_{t_n}^{t_f} \mathbf{c}(\mathbf{r}(t), \mathbf{d})\sigma(t)T(t)dt$$

Volumetric Rendering

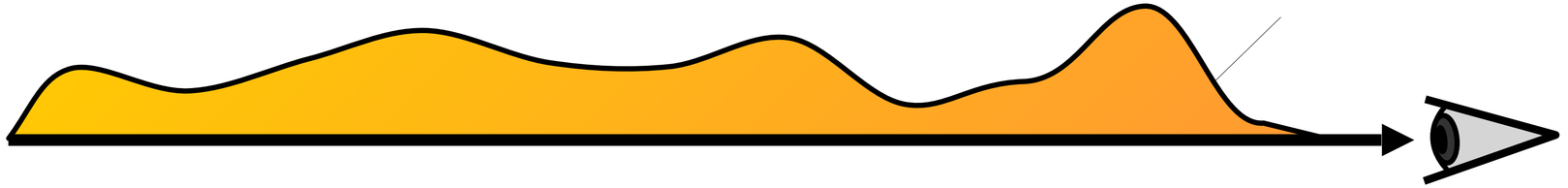
- $p(t) = (1 - T(t))' = T(t)\sigma(t)$ is a probability density function
- Volumetric rendering can be seen as the expected emitted color $c(t)$ under $t \sim p(t)$

$$C(r) = \int_{t_n}^{t_f} \mathbf{c}(\mathbf{r}(t), \mathbf{d})\sigma(t)e^{-\tau(\hat{s},s)}dt = \int_{t_n}^{t_f} \mathbf{c}(\mathbf{r}(t), \mathbf{d})\sigma(t)T(t)dt$$

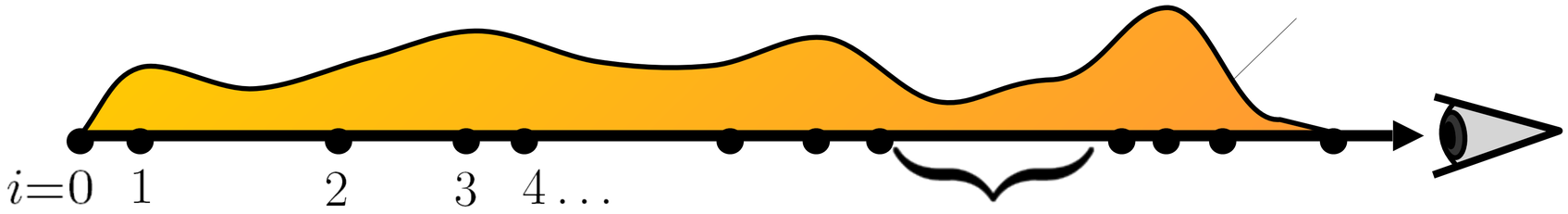
$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})]$$

Numerical Approximation

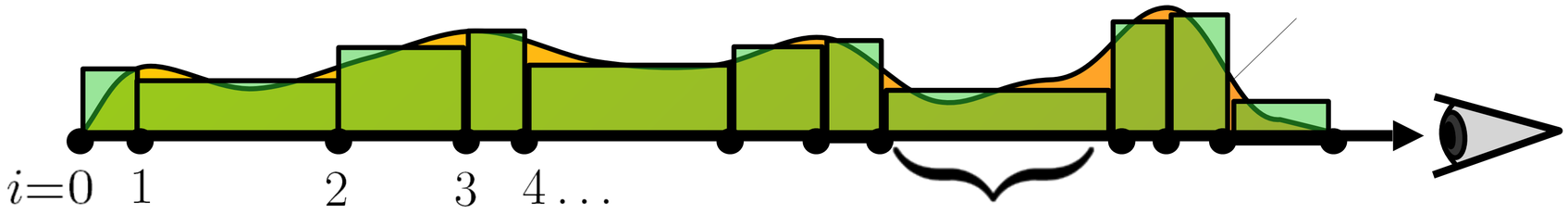
Numerical Approximation



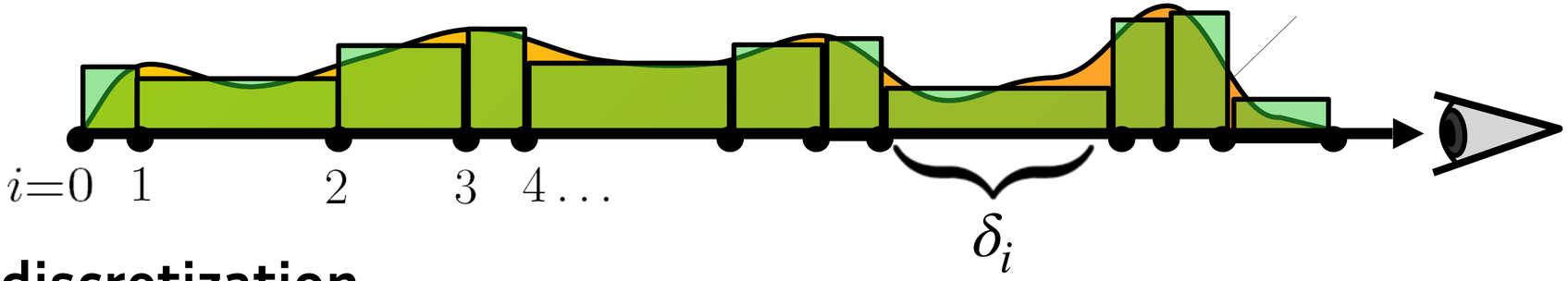
Numerical Approximation



Numerical Approximation



Numerical Approximation



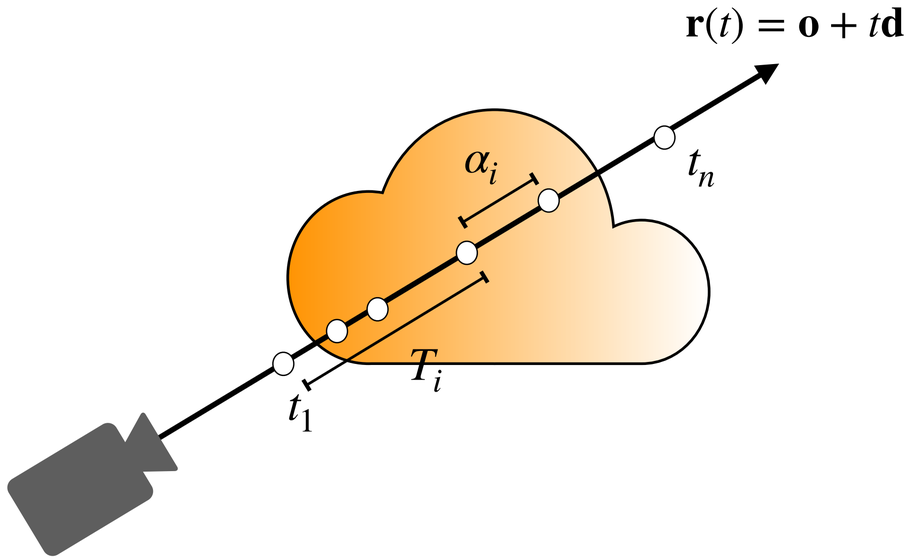
- **Color discretization**

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i, t_{i+1}]}(t)$$

Where $\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

- **Density discretization**

$$\sigma(t) = \sum_{i=0}^{n} \sigma_i \mathbb{I}_{[t_i, t_{i+1}]}(t)$$



Numerical Approximation

Numerical Approximation

- **Approximated color**

Numerical Approximation

- **Approximated color**

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i, t_{i+1}]}(t)$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1})$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) \ = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) \; = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

The ray did not stop in the **[0, t_i)** range

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

↑

The ray did not stop in the **[0, t_i)** range

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

The ray did not stop in the **[0, t_i)** range

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

The ray did not stop in the **[0, t_i)** range

The ray stopped between t_i and t_{i+1}

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

The ray did not stop in the **[0, t_i)** range

The ray stopped between t_i and t_{i+1}

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

The ray did not stop in the **[0, t_i)** range

The ray stopped between t_i and t_{i+1}

$$T(t_i \rightarrow t_{i+1}) = e^{-\int_{t_i}^{t_{i+1}} \sigma(t)dt}$$

$$\sigma(t) = \sum_{i=0}^{n} \sigma_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

The ray did not stop in the **[0, t_i)** range

The ray stopped between t_i and t_{i+1}

$$T(t_i \rightarrow t_{i+1}) = e^{-\int_{t_i}^{t_{i+1}} \sigma(t)dt} = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$\sigma(t) = \sum_{i=0}^{n} \sigma_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

Numerical Approximation

- **Approximated color**

$$C(r) = \mathbb{E}_{t\sim p(t)}[\mathbf{c}(\mathbf{r}(t), \mathbf{d})] = \sum_{i=0}^{n} c_i \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$\mathbb{I}_A(t) = 1$ if $t \in A$ and 0 otherwise

$$\mathbf{c}(t) = \sum_{i=0}^{n} c_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = \mathbb{P}(t_i < \mathbf{t} < t_{i+1}) = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$

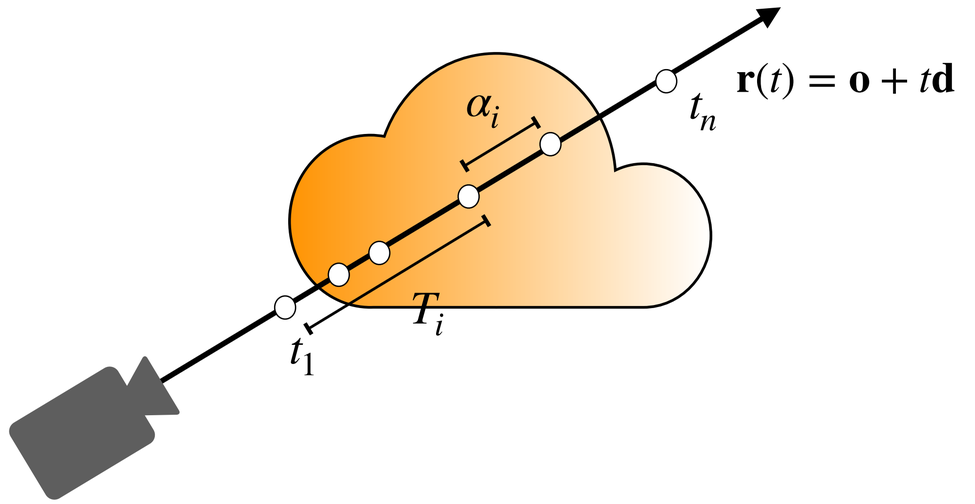
The ray did not stop in the **[0, t_i)** range

The ray stopped between t_i and t_{i+1}

$$T(t_i \rightarrow t_{i+1}) = e^{-\int_{t_i}^{t_{i+1}} \sigma(t)dt} = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$\sigma(t) = \sum_{i=0}^{n} \sigma_i \mathbb{I}_{[t_i,t_{i+1}]}(t)$$

Numerical Approximation

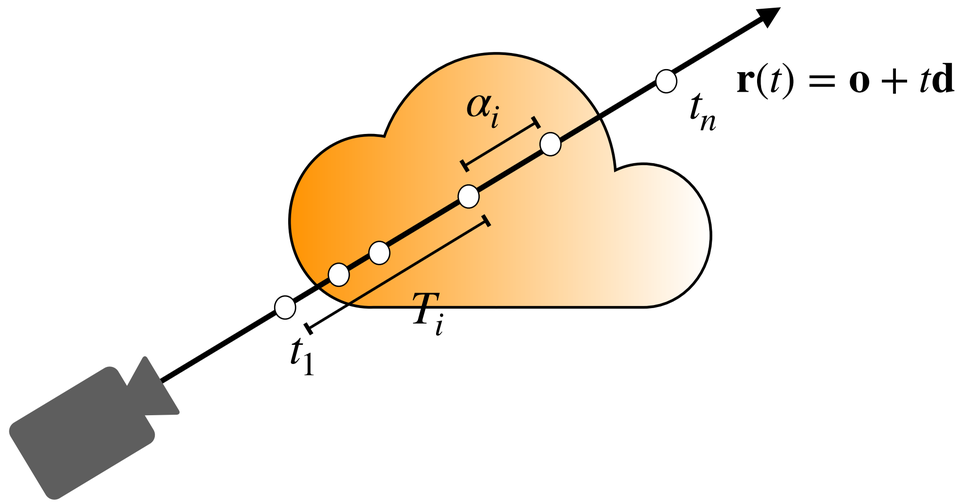


Numerical Approximation

$$T(t_i \rightarrow t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$



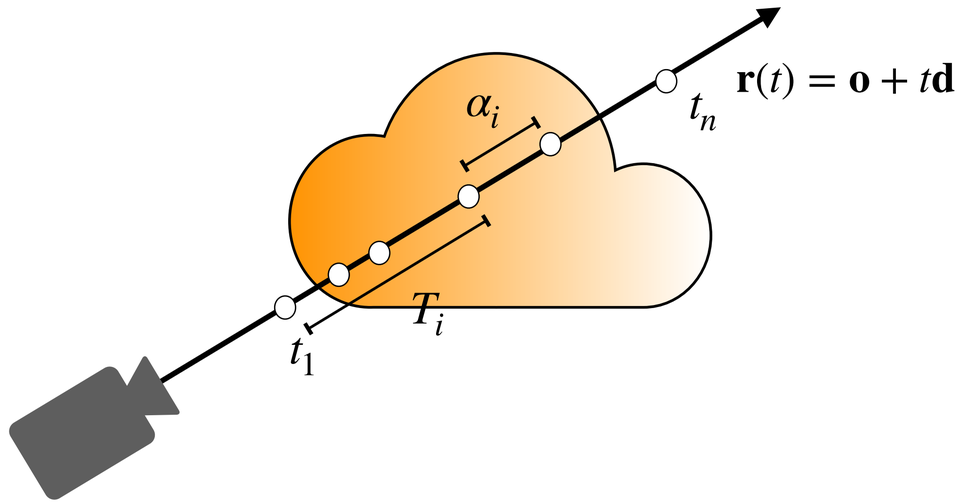
Numerical Approximation

$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



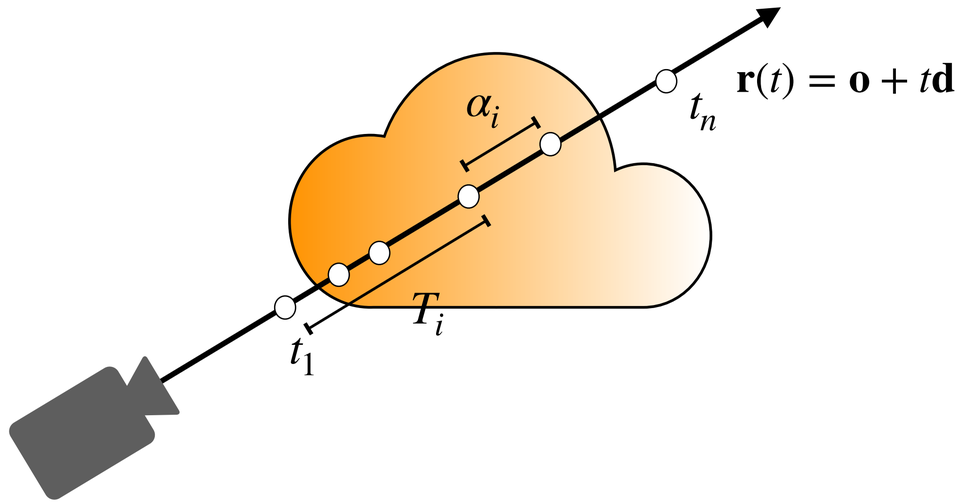
Numerical Approximation

$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



Numerical Approximation

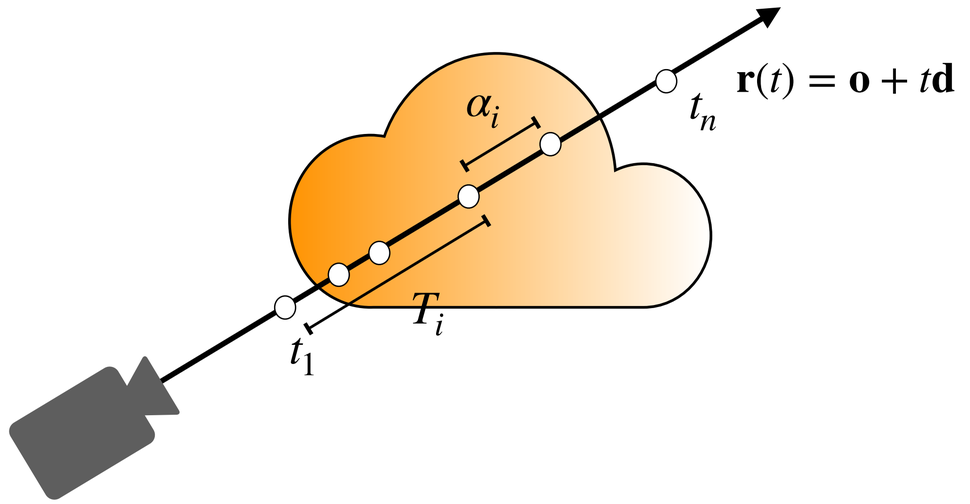
$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$= \sum_{i=0}^{n} T(t_i) \cdot (1 - e^{-\sigma_i(t_{i+1}-t_i)}) \cdot c_i$$

$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



Numerical Approximation

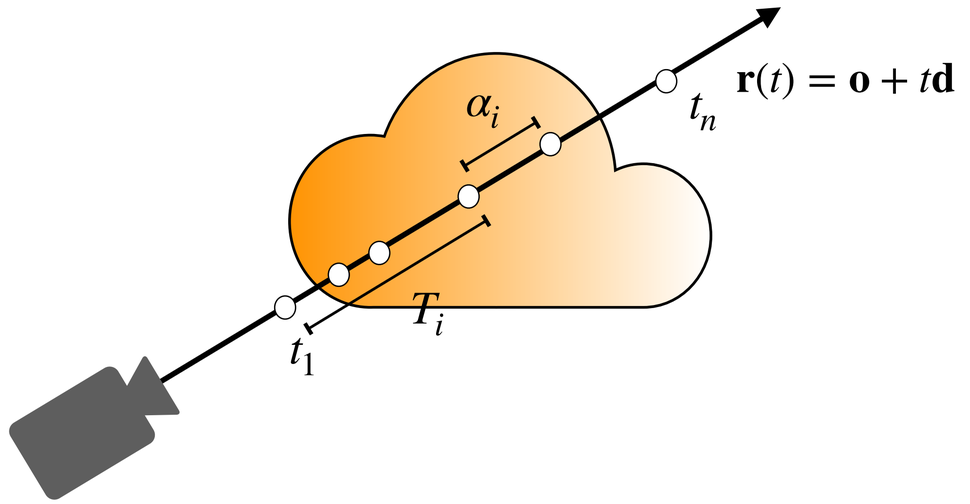
$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$= \sum_{i=0}^{n} T(t_i) \cdot (1 - e^{-\sigma_i(t_{i+1}-t_i)}) \cdot c_i$$

$$T(t_i \rightarrow t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \rightarrow t_{i+1}))$$



Numerical Approximation

$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

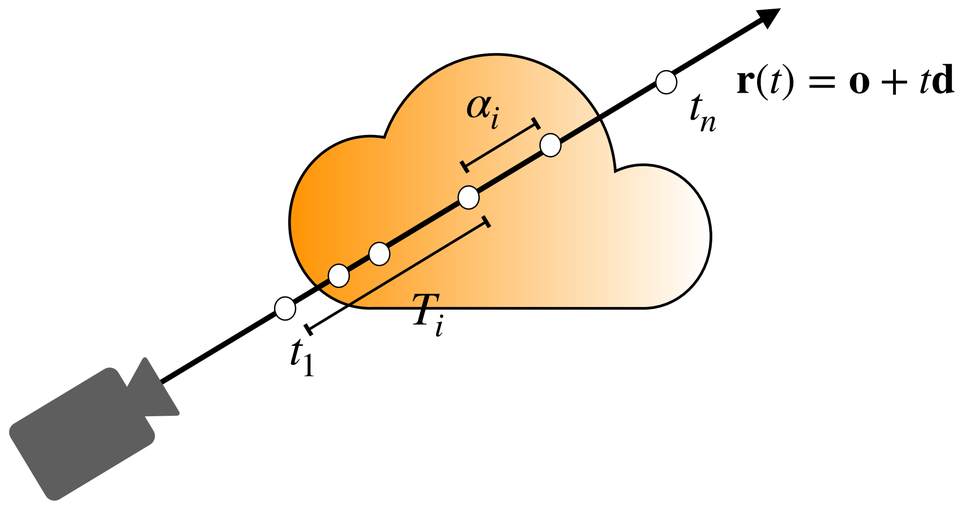
$$= \sum_{i=0}^{n} T(t_i) \cdot (1 - e^{-\sigma_i(t_{i+1}-t_i)}) \cdot c_i$$

$$\alpha_i = (1 - e^{-\sigma_i(t_{i+1}-t_i)})$$

$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



Numerical Approximation

$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$= \sum_{i=0}^{n} T(t_i) \cdot (1 - e^{-\sigma_i(t_{i+1}-t_i)}) \cdot c_i$$

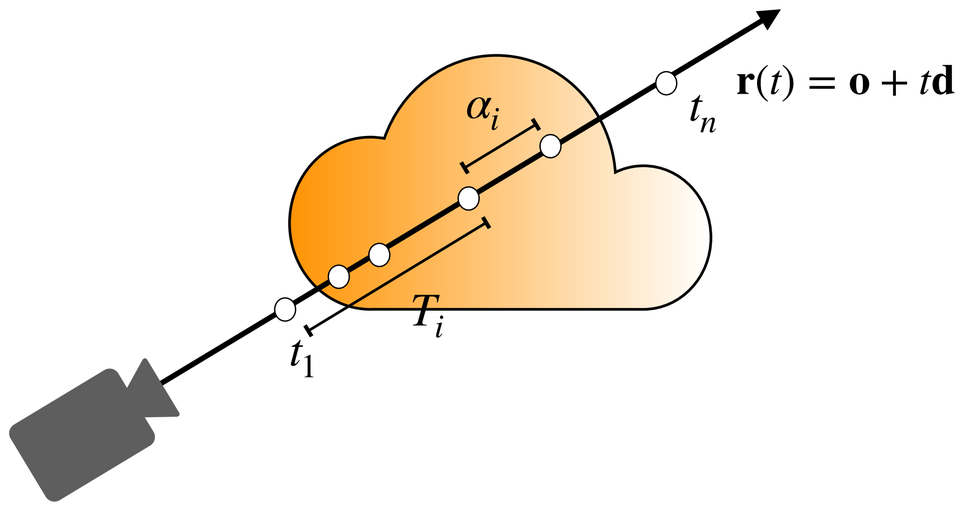
$$= \sum_{i=0}^{n} \prod_{k=0}^{i} (1 - \alpha_k) \cdot \alpha_i \cdot c_i$$

$$\alpha_i = (1 - e^{-\sigma_i(t_{i+1}-t_i)})$$

$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



Numerical Approximation

$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$= \sum_{i=0}^{n} T(t_i) \cdot (1 - e^{-\sigma_i(t_{i+1}-t_i)}) \cdot c_i$$

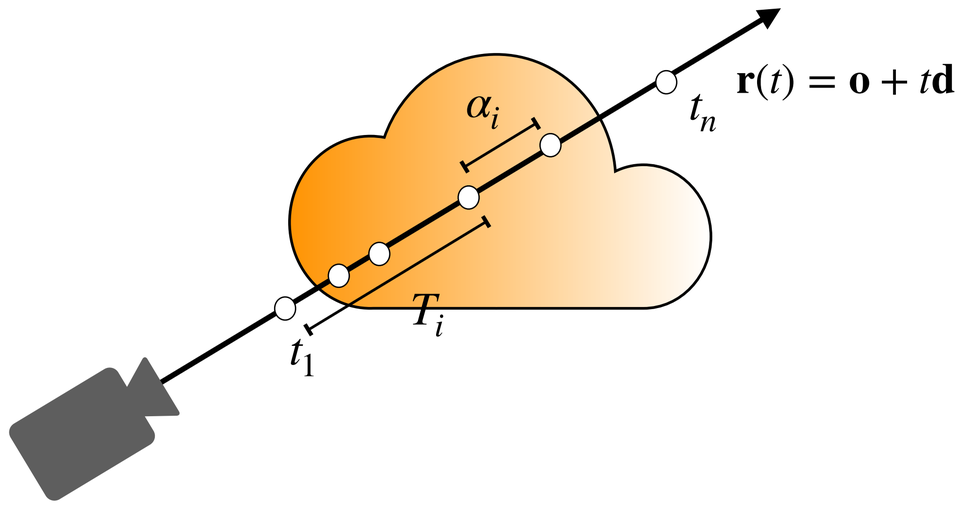
$$= \sum_{i=0}^{n} \prod_{k=0}^{i} (1 - \alpha_k) \cdot \alpha_i \cdot c_i$$

$$\alpha_i = (1 - e^{-\sigma_i(t_{i+1}-t_i)})$$

$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



Numerical Approximation

$$C(r) = \sum_{i=0}^{n} c_i \cdot \mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)]$$

$$= \sum_{i=0}^{n} T(t_i) \cdot (1 - e^{-\sigma_i(t_{i+1}-t_i)}) \cdot c_i$$

$$= \sum_{i=0}^{n} \prod_{k=0}^{i} (1 - \alpha_k) \cdot \alpha_i \cdot c_i$$

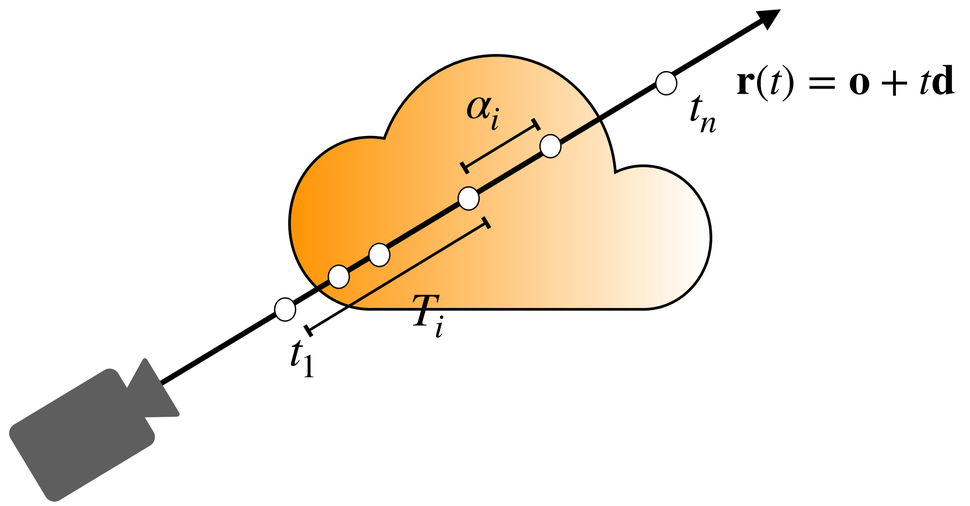
$$\alpha_i = (1 - e^{-\sigma_i(t_{i+1}-t_i)})$$

Alpha compositing !

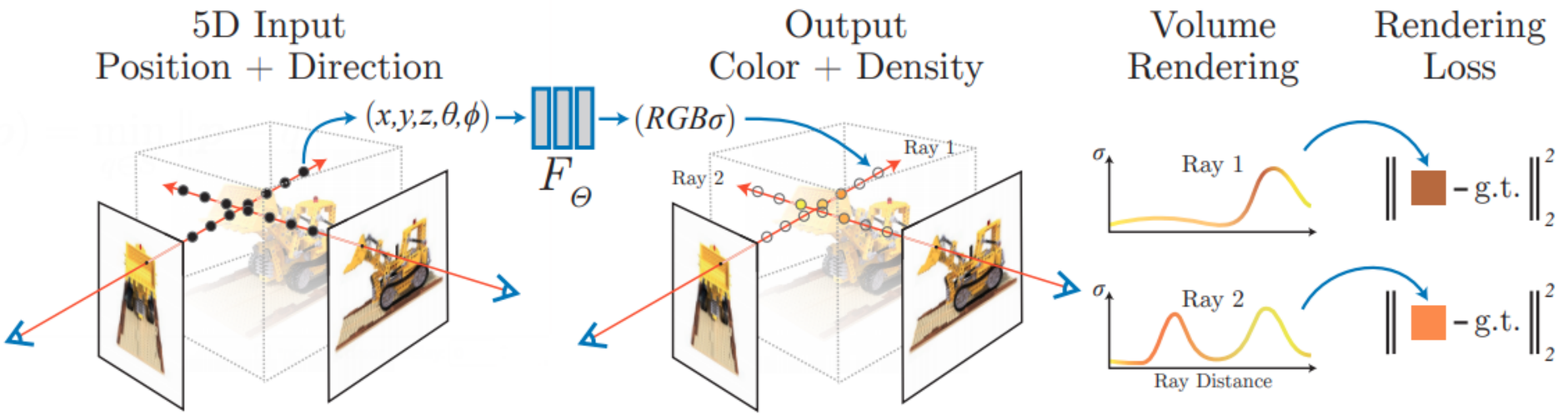
$$T(t_i \to t_{i+1}) = e^{-\sigma_i(t_{i+1}-t_i)}$$

$$T(t_i) = e^{\sum_{i=0}^{n} -\sigma_i(t_{i+1}-t_i)}$$

$$\mathbb{E}_{t\sim p(t)}[\mathbb{I}_{[t_i,t_{i+1}]}(t)] = T(t_i)(1 - T(t_i \to t_{i+1}))$$



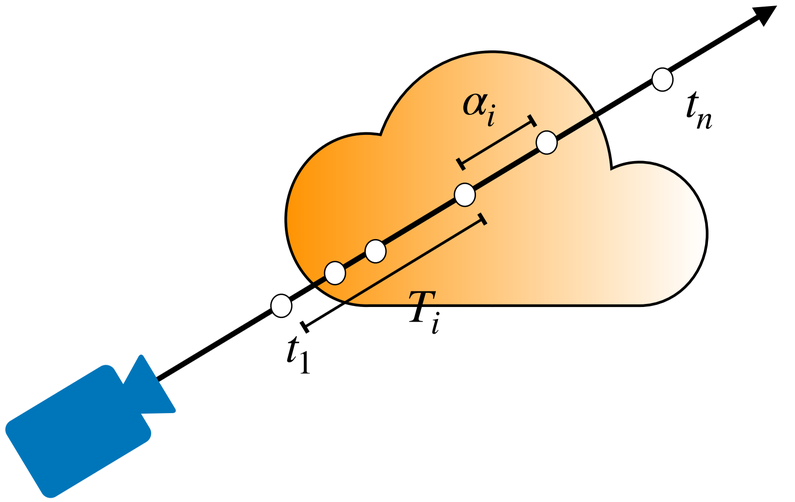
NERF in a nutshell



$$C(\mathbf{r},\theta) = \sum_{i=0}^{n} \prod_{k=0}^{i} (1-\alpha_k) \cdot \alpha_i \cdot c_i$$

$$\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

$$\mathscr{L}(\theta) = \mathbb{E}_{\mathbf{C}\sim\mathbf{C}_i} \mathbb{E}_{\mathbf{r}\sim\mathbf{C}} \| \mathbf{C}(\mathbf{r},\theta) - \mathbf{C}_i(\mathbf{r}) \|_2^2$$

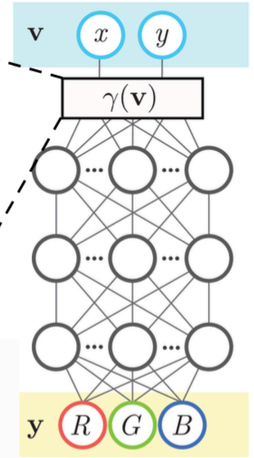


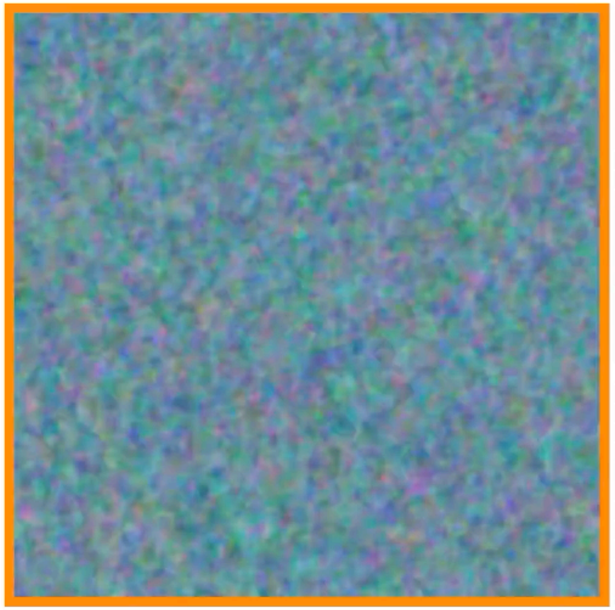
Mildenhall et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. ECCV 2020

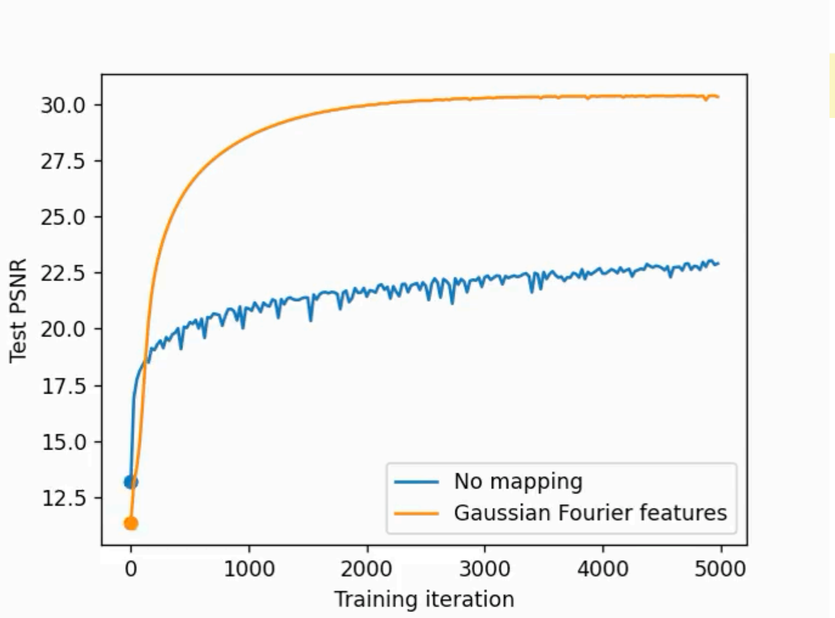
Positional Encoding

- Spectral bias: neural networks are biased to fit lower frequency signals

$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \gamma_2(\mathbf{x}), ..., \gamma_m(\mathbf{x})]$$
$$\gamma_{(2i)}(x) = sin(2^{i-1}\pi x),$$
$$\gamma_{(2i+1)}(x) = cos(2^{i-1}\pi x)$$





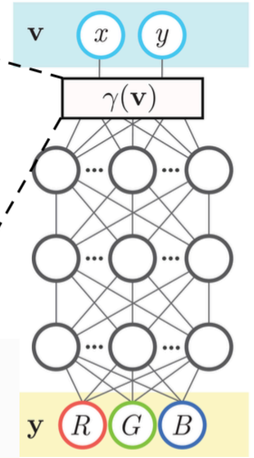


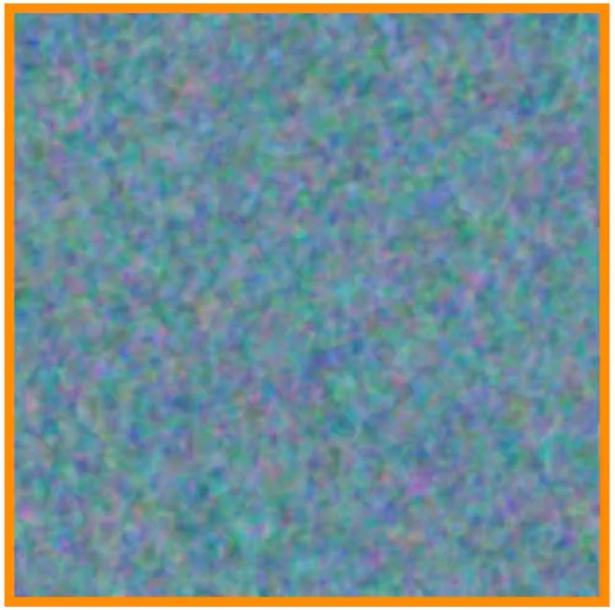
Tancik et al “Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains” NeurIPS 2020

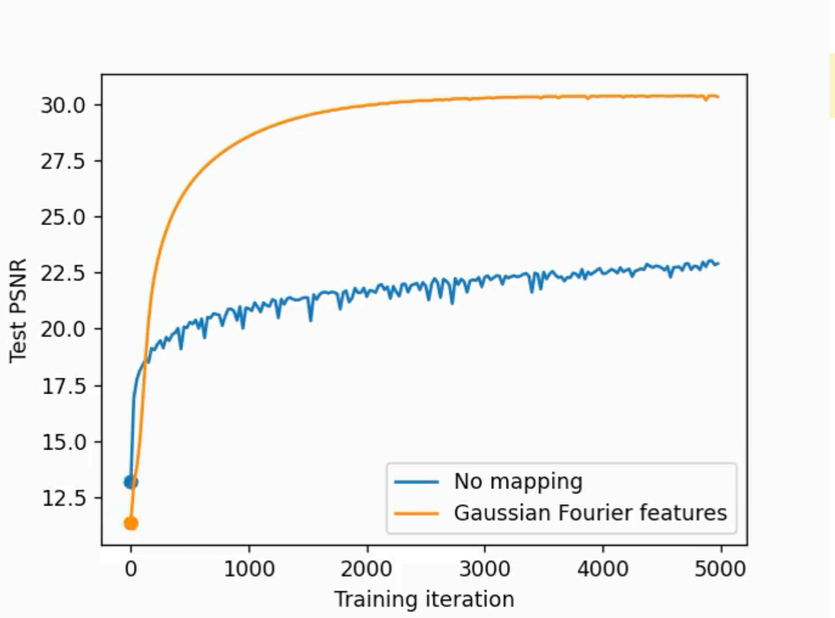
Positional Encoding

- Spectral bias: neural networks are biased to fit lower frequency signals

$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \gamma_2(\mathbf{x}), ..., \gamma_m(\mathbf{x})]$$
$$\gamma_{(2i)}(x) = sin(2^{i-1}\pi x),$$
$$\gamma_{(2i+1)}(x) = cos(2^{i-1}\pi x)$$



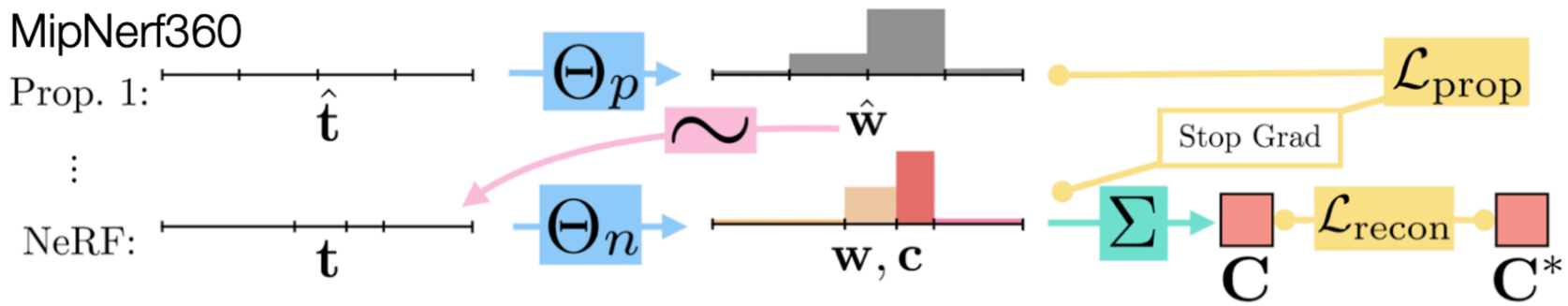




Tancik et al “Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains” NeurIPS 2020

Proposal Network

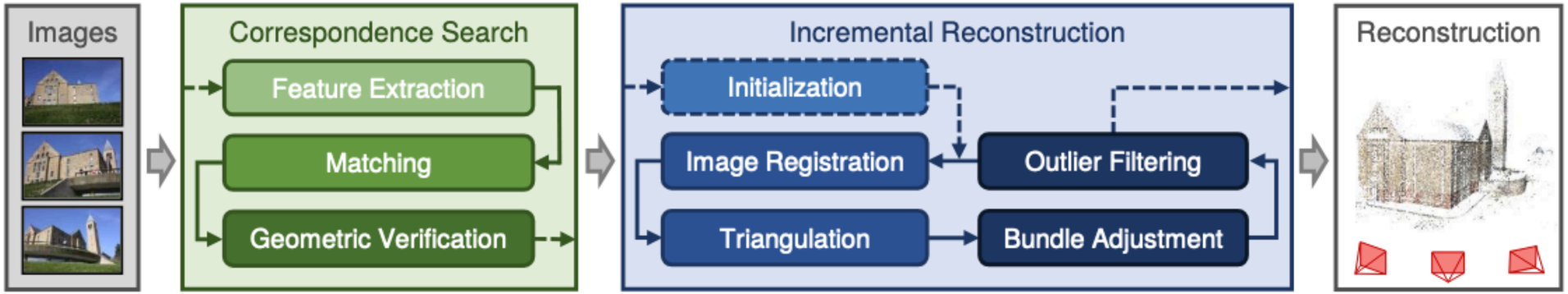
- Must decide where to place integration quadrature points
- How can we know where to place samples? … train a proposal network



Barron et al. “**Mip-NeRF 360**: Unbounded Anti-Aliased Neural Radiance Fields” CVPR 2022 or Li et al. “**NerfAcc**: A General NeRF Acceleration Toolbox” arXiv

Structure from Motion

- How do we get camera positions given a set of images ?

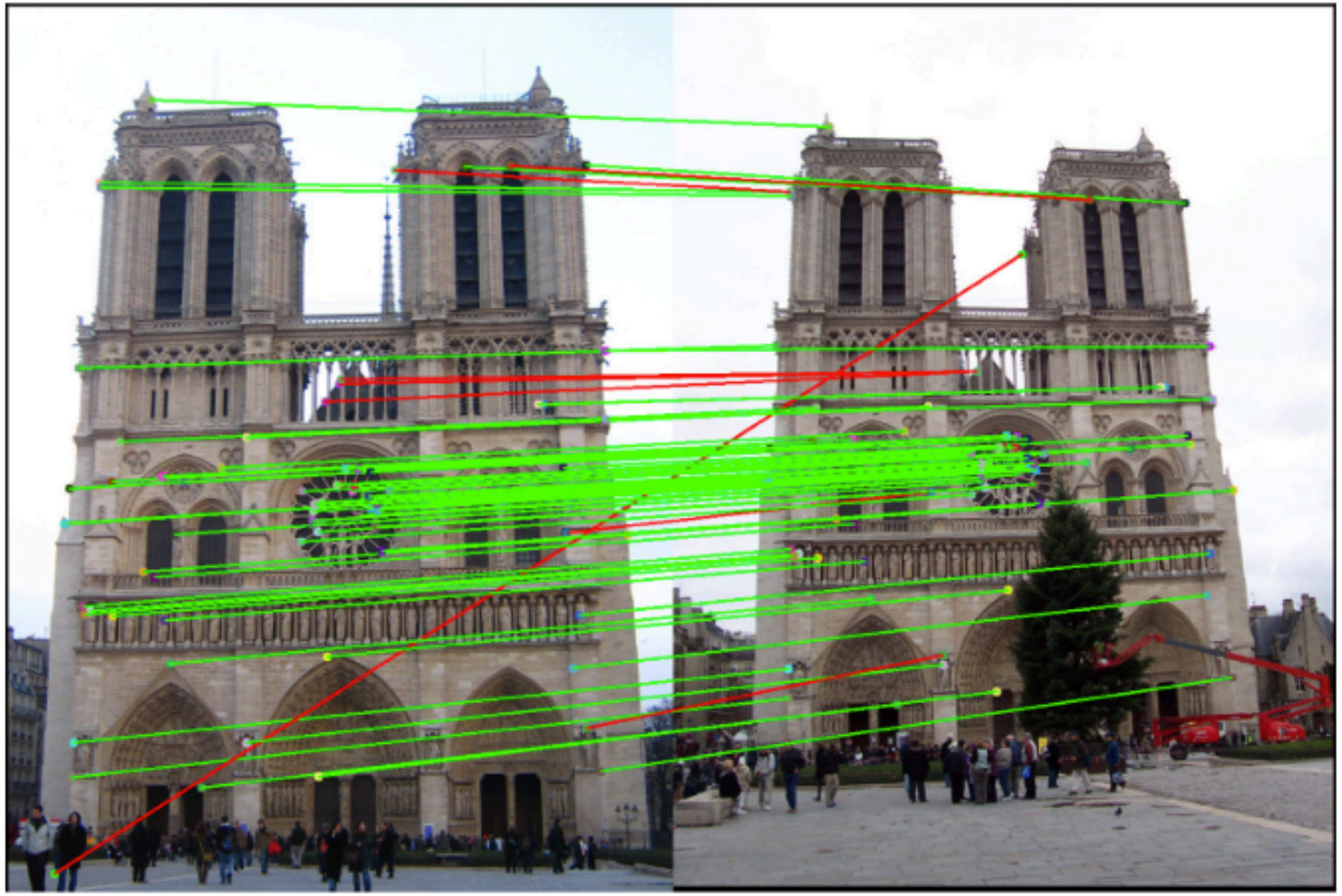


- **Correspondence Search**: Find and match robust 2D features across images
- **Incremental Reconstruction**: Start with 2 views, incrementally add cameras

Sch¨onberger and Frahm: Structure-from-Motion Revisited. CVPR, 2016.

https://drive.google.com/file/d/1QCDEM5l-wi7VLi4b8cw_ErmCAyJOxb10/view

Correspondence search



Schönberger and Frahm: Structure-from-Motion Revisited. CVPR, 2016.

https://drive.google.com/file/d/1QCDEM5l-wi7VLi4b8cw_ErmCAyJOxb10/view

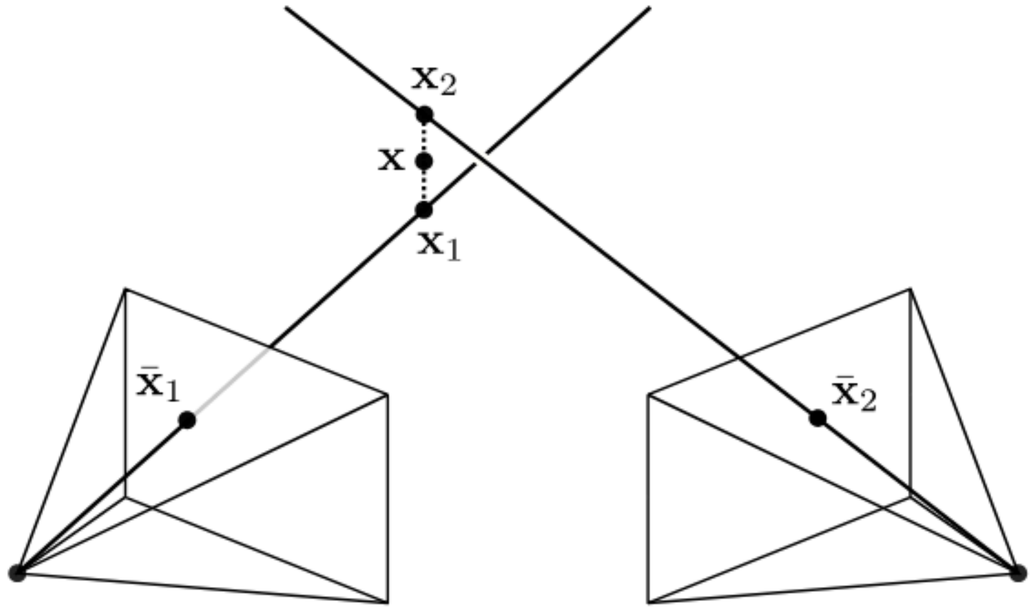
Triangulation

- Given the camera intrinsics and extrinsics, how can we recover 3D geometry?

- Minimize the reprojection error of corresponding points:

$$\bar{\mathbf{x}}_w^* = \underset{\bar{\mathbf{x}}_w}{\operatorname{argmin}} \sum_{i=1}^{N} \|\bar{\mathbf{x}}_i^s(\bar{\mathbf{x}}_w) - \bar{\mathbf{x}}_i^o\|_2^2$$

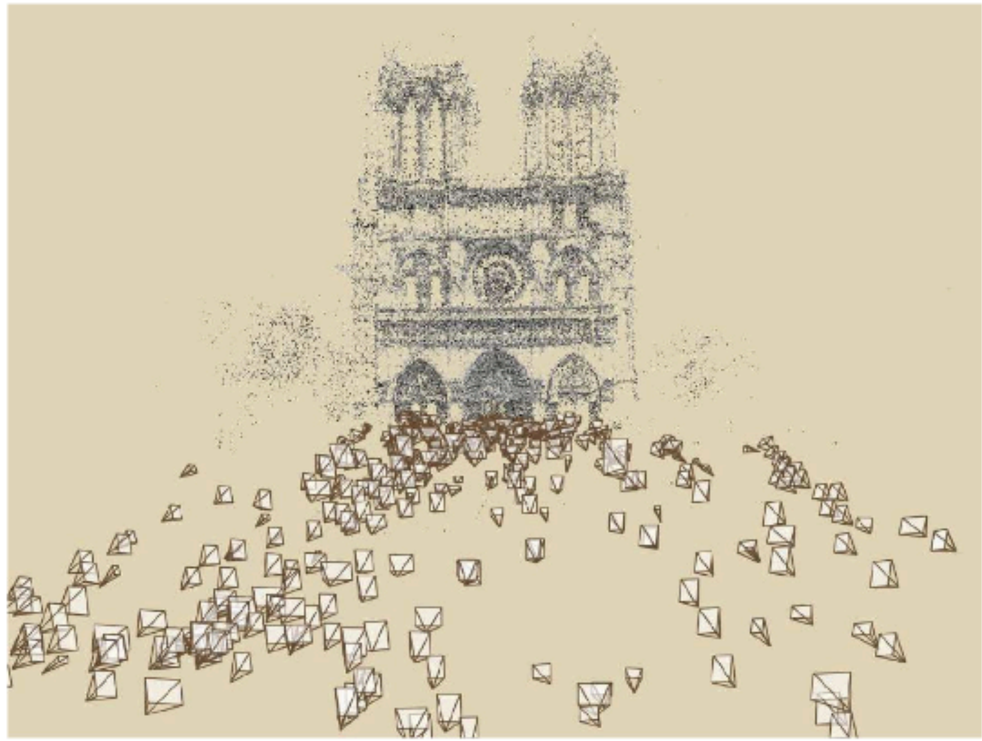
- In COLMAP, a robust triangulation method is proposed that handles outliers



Schönberger and Frahm: Structure-from-Motion Revisited. CVPR, 2016.

https://drive.google.com/file/d/1QCDEM5l-wi7VLi4b8cw_ErmCAyJOxb10/view

Bundle adjustement



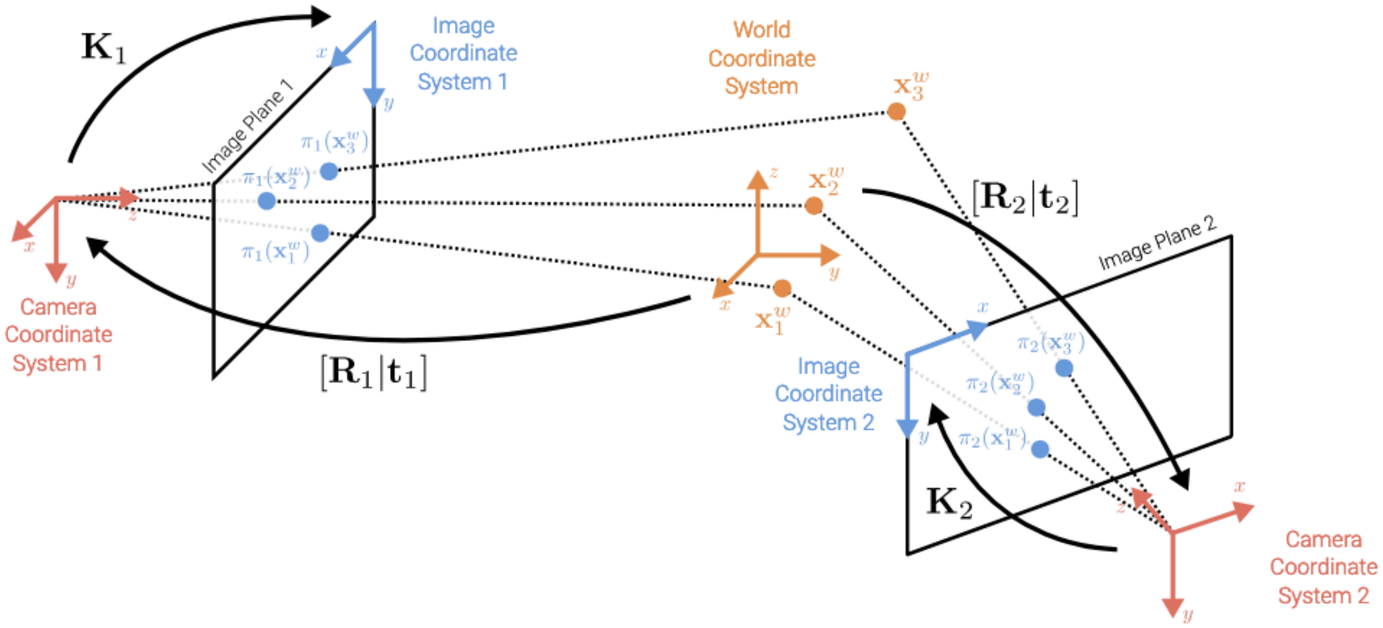
Goal: Optimize reprojection errors (distance between observed feature and projected 3D point in image plane) wrt. camera parameters and 3D point cloud

Snavely, Seitz and Szeliski: Photo tourism: exploring photo collections in 3D. SIGGRAPH, 2006.

Bundle adjustement

- Minimize the reprojection error of corresponding points **wrt. camera parameters and 3D point cloud**:

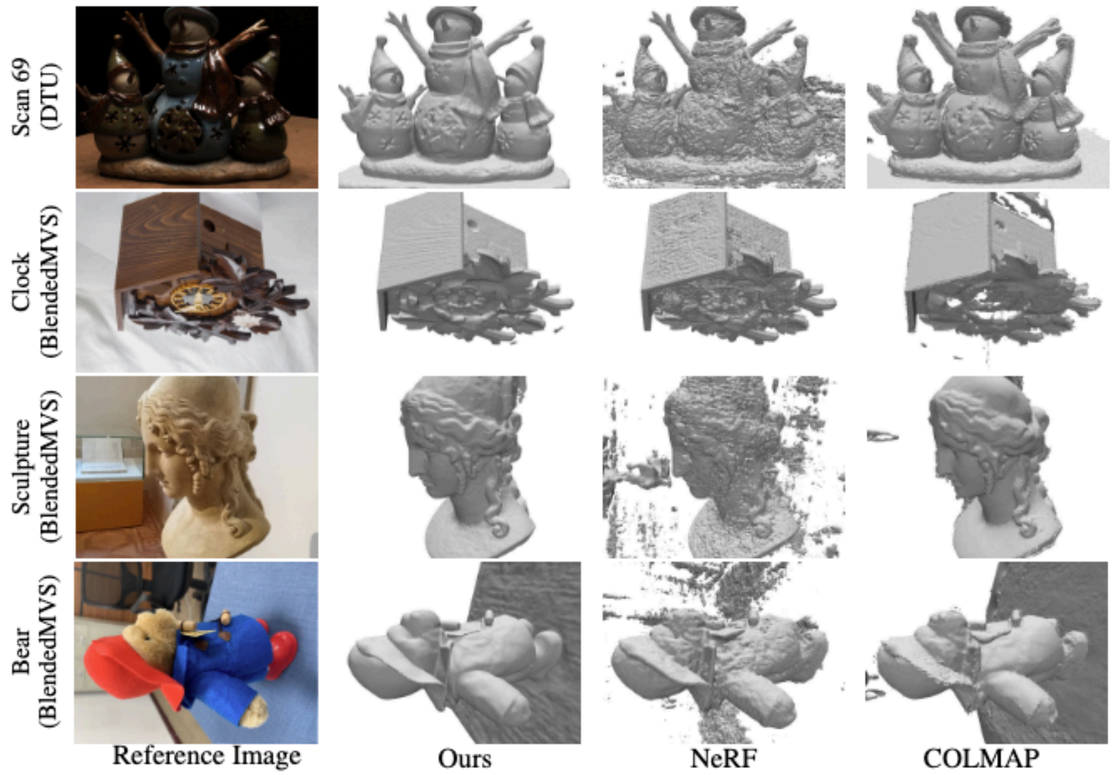
$$\Pi^*, \mathcal{X}_w^* = \underset{\Pi, \mathcal{X}_w}{\operatorname{argmin}} \sum_{i=1}^{N} \sum_{p=1}^{P} w_{ip} \left\| \mathbf{x}_{ip}^s - \pi_i(\mathbf{x}_p^w) \right\|_2^2$$



https://drive.google.com/file/d/1QCDEM5l-wi7VLi4b8cw_ErmCAyJOxb10/view

Reconstruction

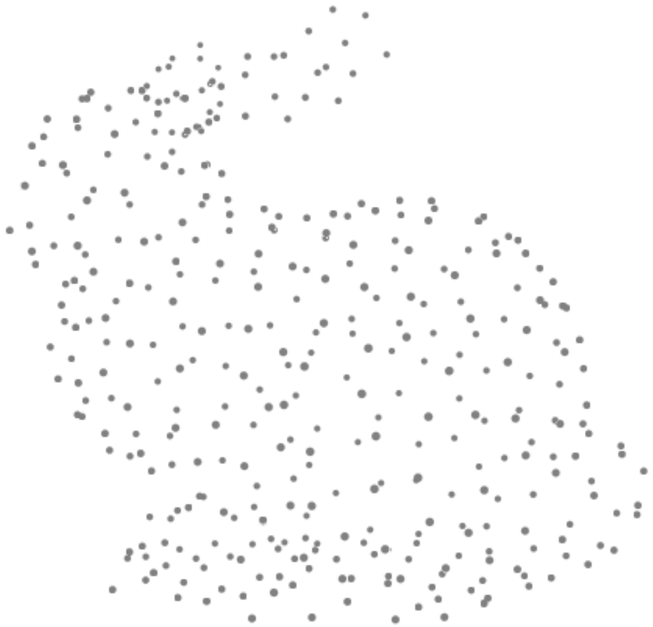
- Can we recover a mesh from NERF ?

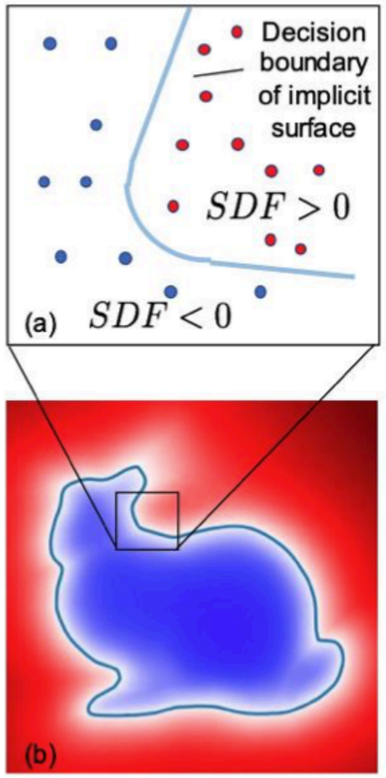


Wang at al. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. NeurIPS 2021

Signed distance function

- Using Neural Networks as a Continuous Shape Representation ?



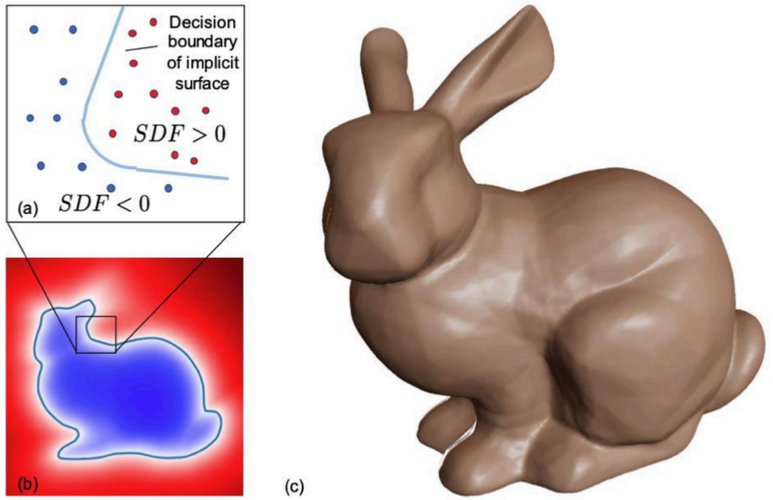


(c)

Transmittance from SDF

- How can we relate Transmittance to SDF ?

$$T(t) = \phi(f(\mathbf{r}(t)) = \frac{1}{1 + e^{-s \cdot f(\mathbf{r}(t)}}$$



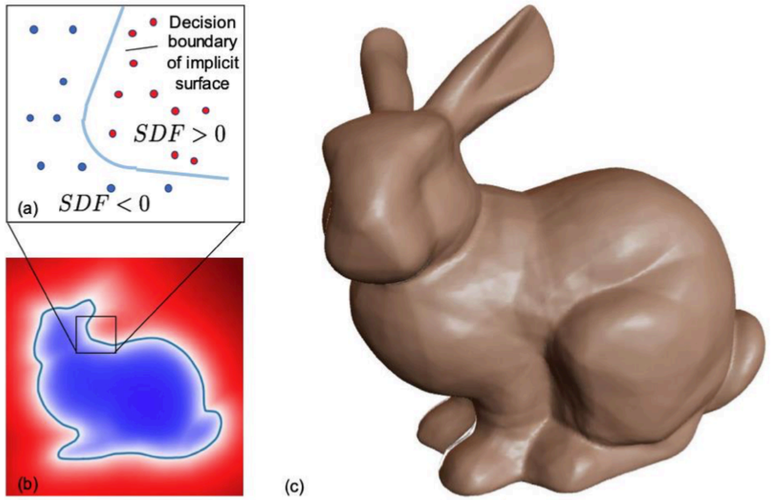
- The tansmittance starts at 1 at t = tn and is a monotonic decreasing function to 0 inside the object.

Wang et al. HF-NeuS: Improved Surface Reconstruction Using High-Frequency Details. NeurIPS 2022

Density from Transmittance

- The volume density σ can be calculated as :

$$\sigma(t) = -\frac{T'(t)}{T(t)}$$

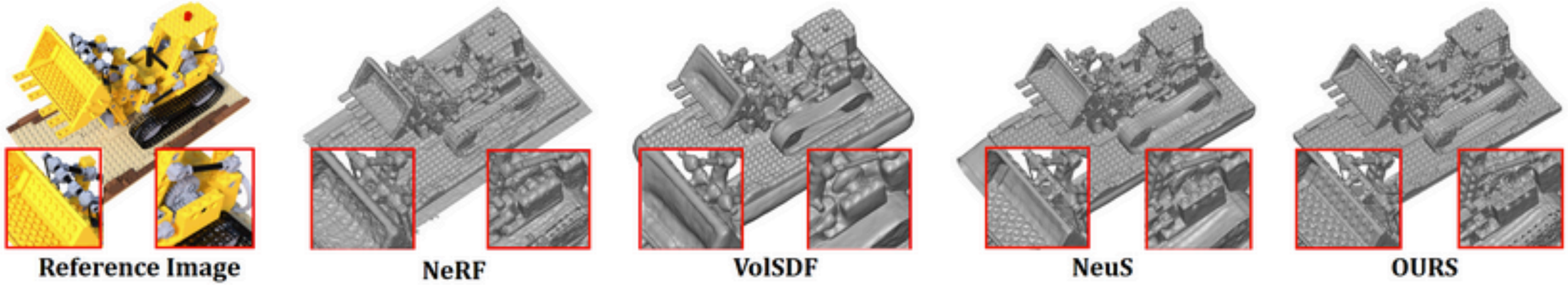


- After discretization:

$$\sigma(t_i) = s(\phi(f(\mathbf{r}(t_i)) - 1)\nabla f(t_i) \cdot d$$

Wang et al. HF-NeuS: Improved Surface Reconstruction Using High-Frequency Details. NeurIPS 2022

Qualitative Results



Wang et al. HF-NeuS: Improved Surface Reconstruction Using High-Frequency Details. NeurIPS 2022

Reconstruction from sparse inputs

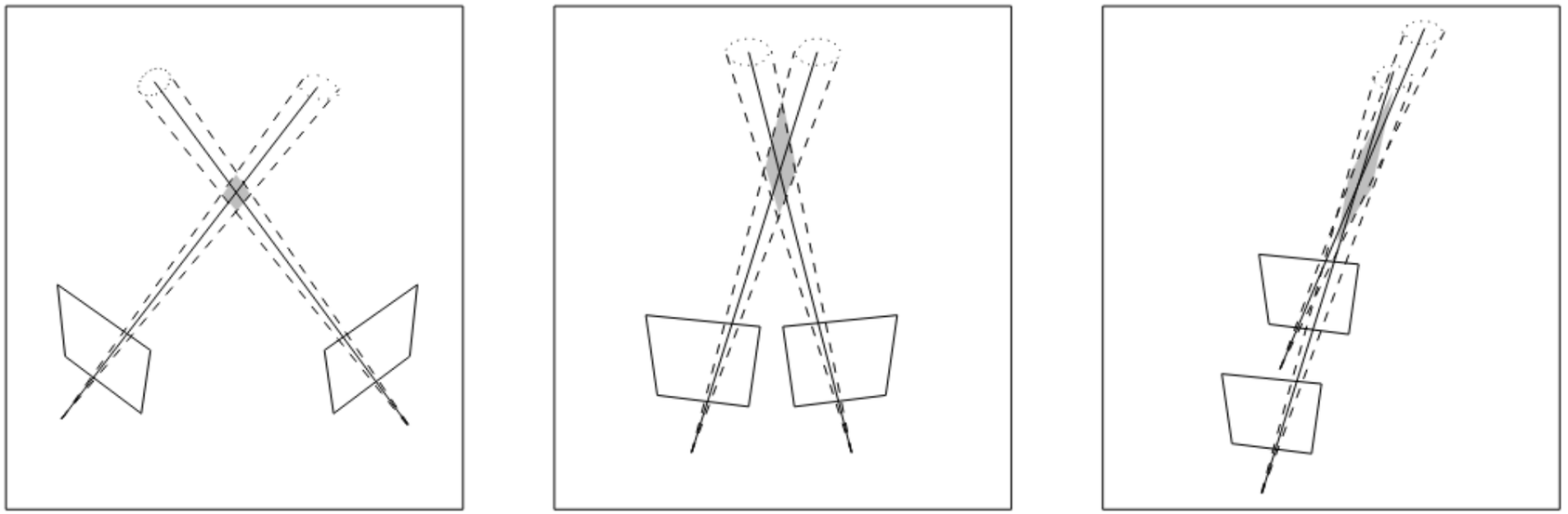
- What happens when only have 3 images ?

T&T dataset

SparseCraft: Few-Shot Neural Reconstruction through Stereopsis Guided Geometric Linearization. ECCV 2024

Multi-view stereo supervision

- Colmap provides and additionnal step than uses multi-view stereo to extract dense surface points with normals and colors.
- However this supervision can be very noisy



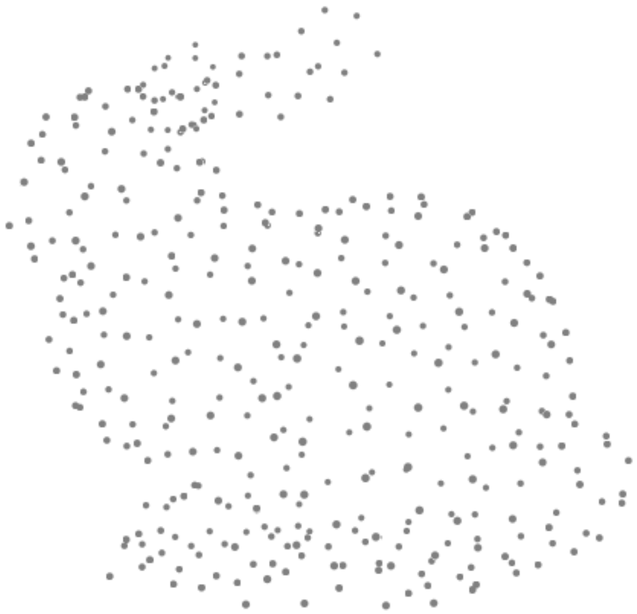
- **Uncertainty** (shaded region) increases as the rays become more parallel
- **Tradeoff**: feature matching easier for nearby views, but triangulation is harder

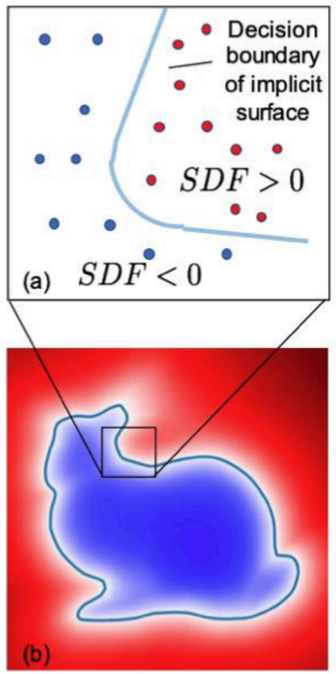
SparseCraft: Few-Shot Neural Reconstruction through Stereopsis Guided Geometric Linearization. ECCV 2024

https://drive.google.com/file/d/1QCDEM5l-wi7VLi4b8cw_ErmCAyJOxb10/view

Robust supervision

- **Simplify the problem:** How to learn an SDF from a noisy and sparse pointcloud ?





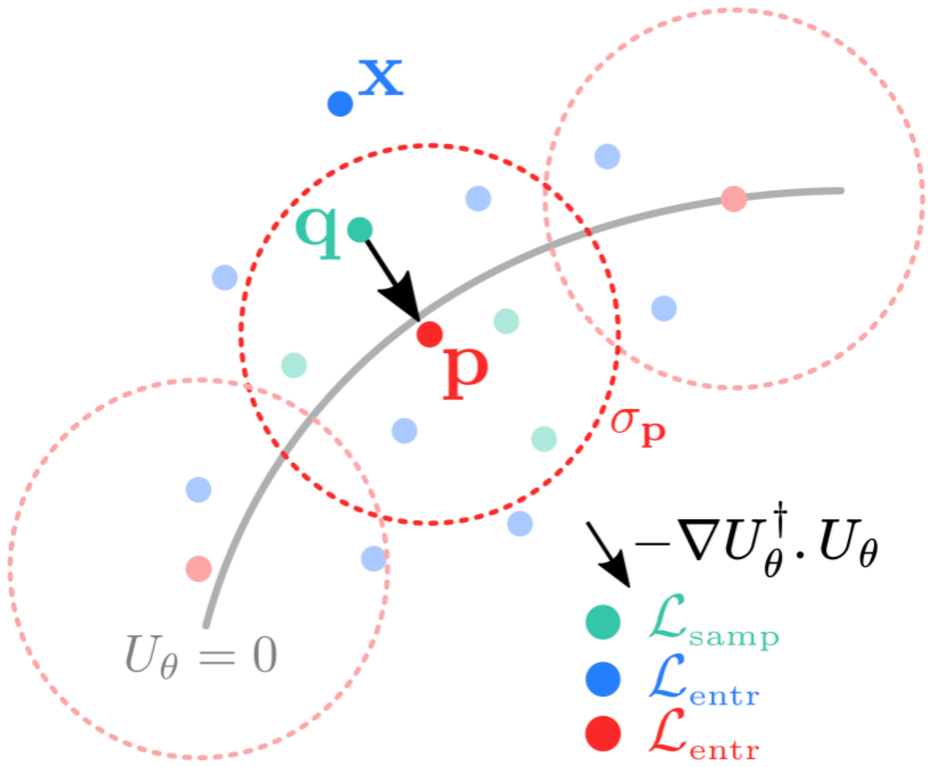
(c)

Unsupervised Occupancy Learning from Sparse Point Cloud. CVPR 2024

Robust supervision

- **Local linearization :** Supervise the function with it's first order Taylor approximation near the surface.

- Stabilize the training by biasing the occupancy function towards **minimal entropy field**,
  while maximizing its entropy at the input point cloud.

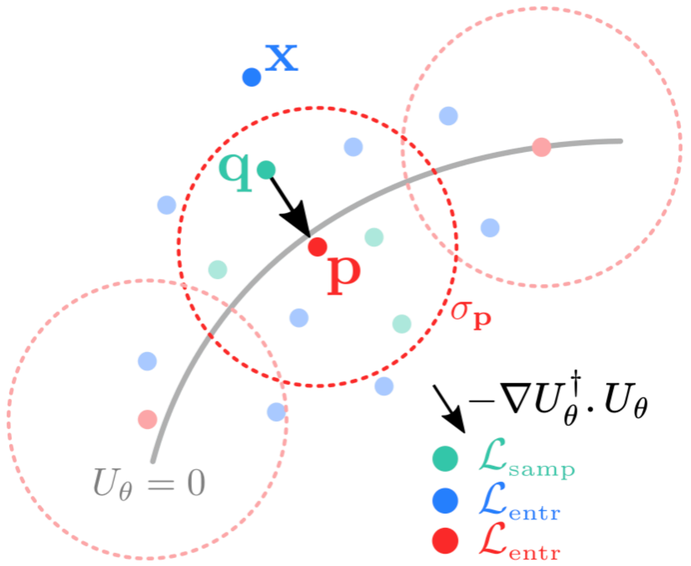


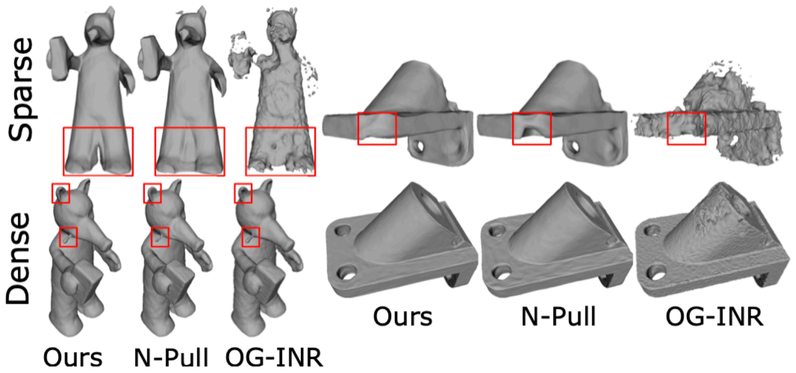
Unsupervised Occupancy Learning from Sparse Point Cloud. CVPR 2024

Robust supervision

- **Local linearization :** Supervise the function with it's first order Taylor approximation near the surface.

- Stabilize the training by biasing the occupancy function towards **minimal entropy field**,
  while maximizing its entropy at the input point cloud.

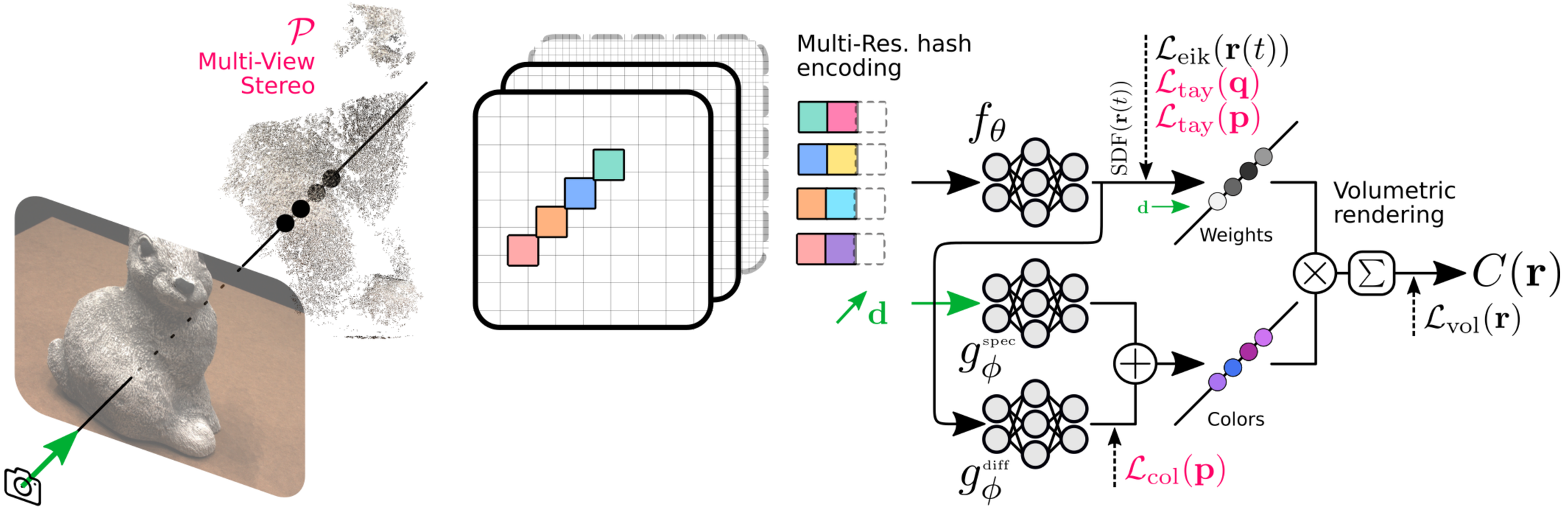




SRB Dataset

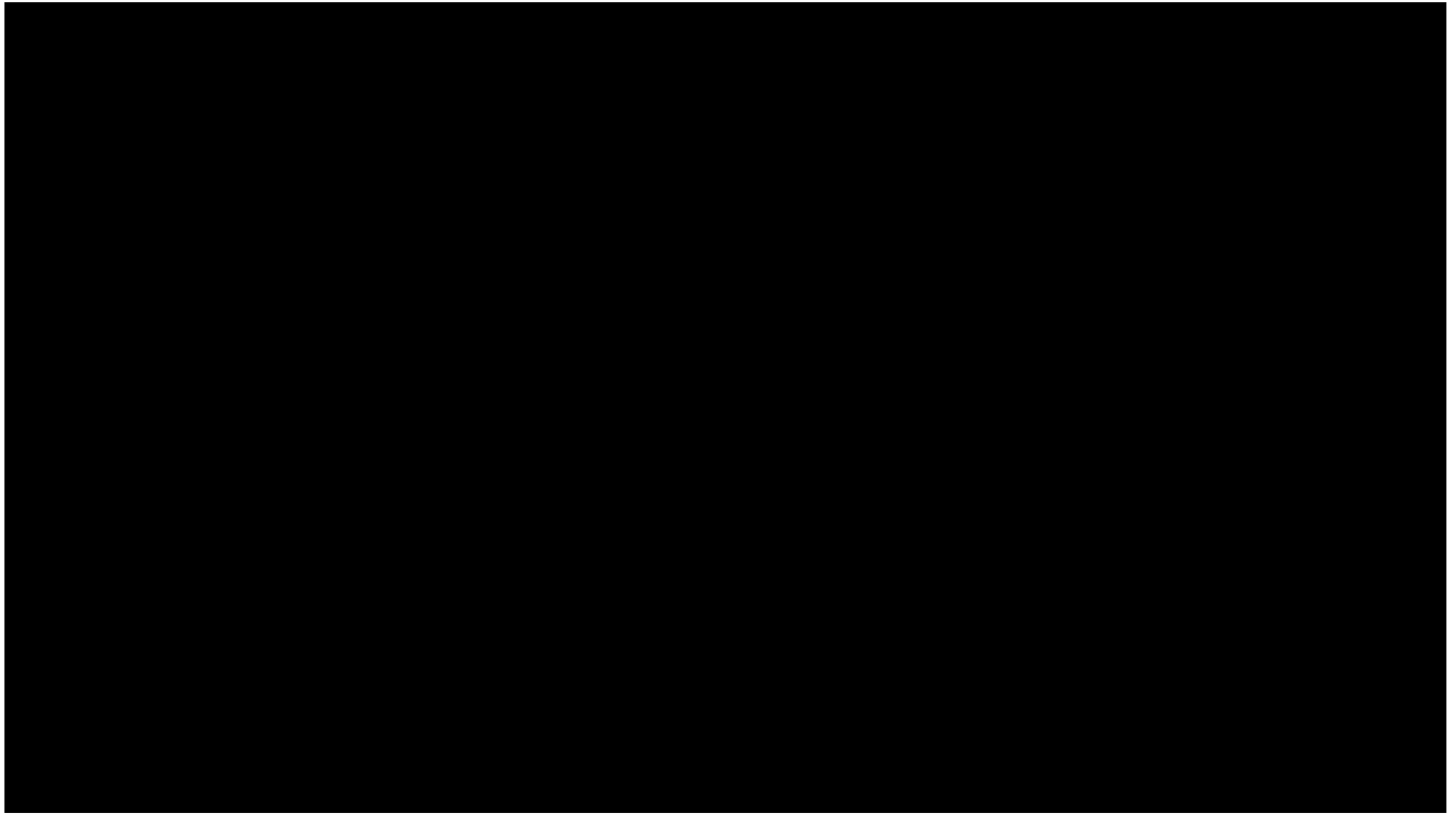
Unsupervised Occupancy Learning from Sparse Point Cloud. CVPR 2024

SparseCraft



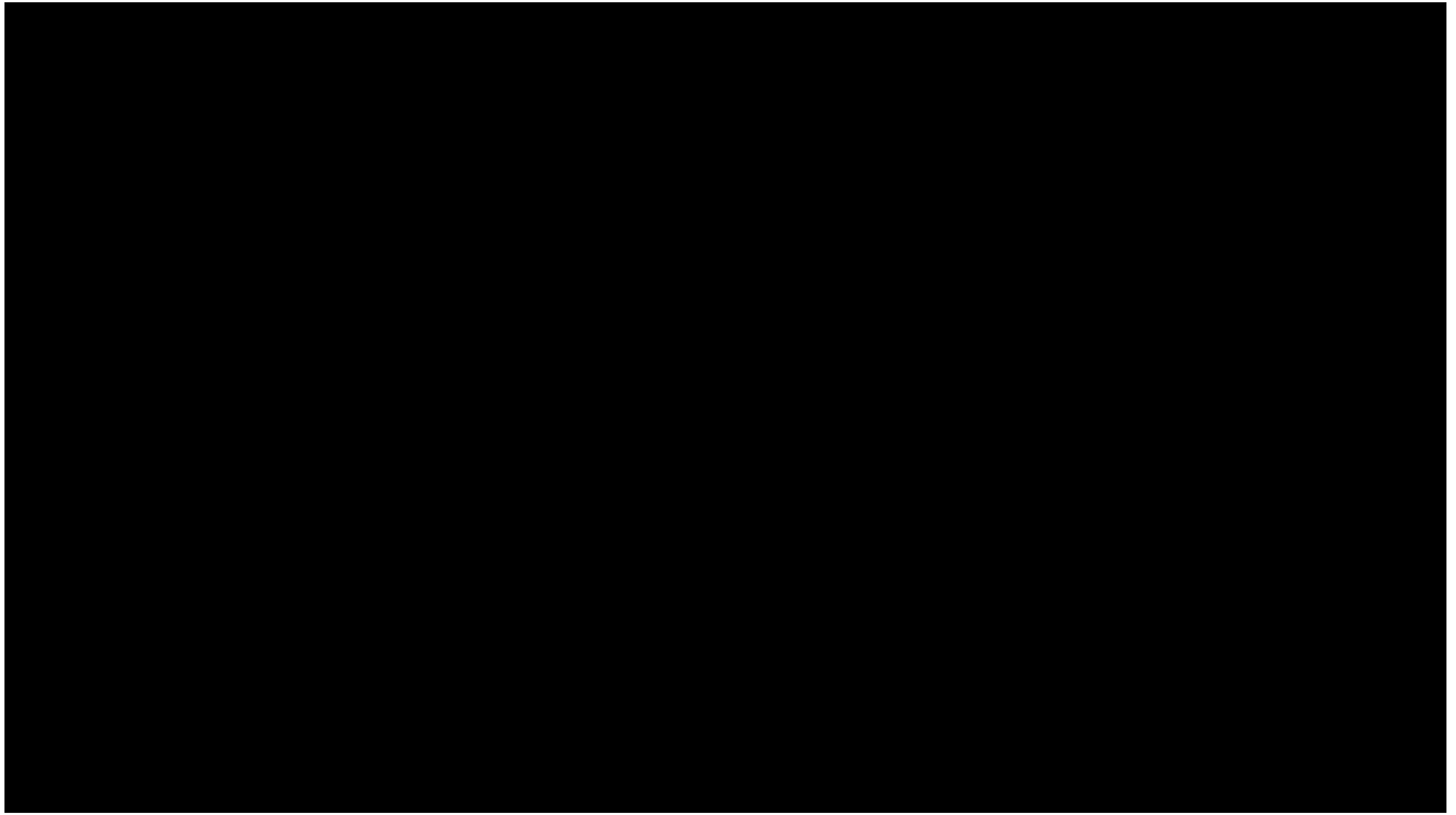
SparseCraft: Few-Shot Neural Reconstruction through Stereopsis Guided Geometric Linearization. ECCV 2024

SparseCraft



SparseCraft: Few-Shot Neural Reconstruction through Stereopsis Guided Geometric Linearization. ECCV 2024

SparseCraft



SparseCraft: Few-Shot Neural Reconstruction through Stereopsis Guided Geometric Linearization. ECCV 2024