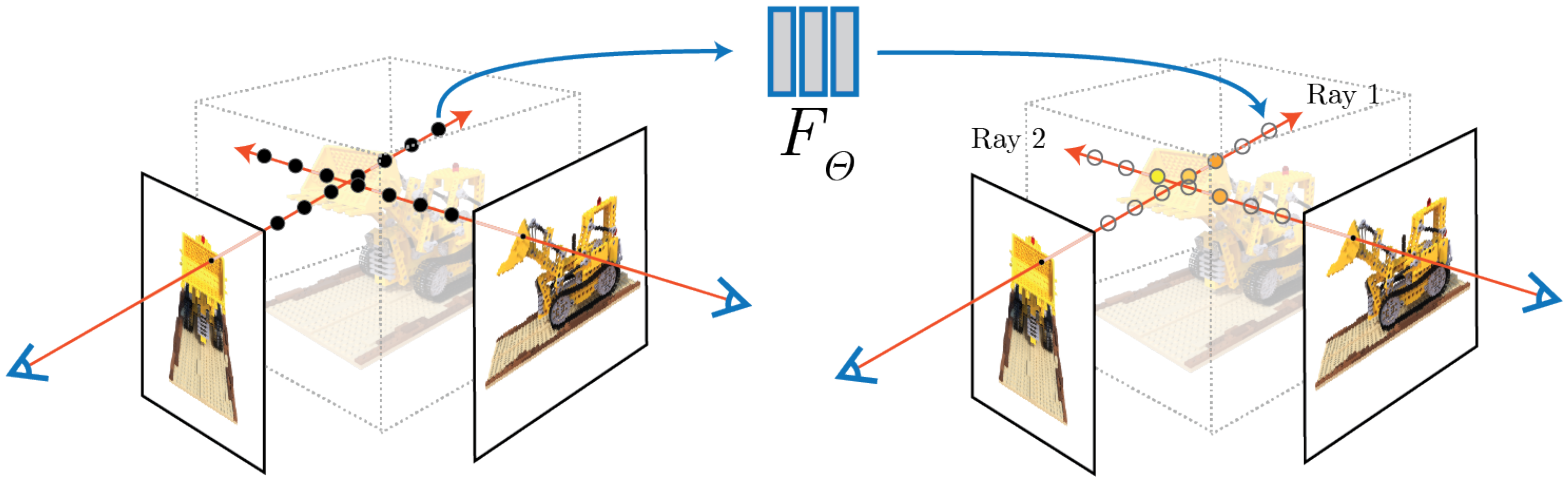


NeRF

Representing Scenes as Neural Radiance Fields for View Synthesis

Seungwoo Yoo, KAIST SoC

Problem: Novel View Synthesis

Problem: Novel View Synthesis

3D Photo? Interesting.

An example of posting 3D photo on Facebook timeline. Source: Facebook

Problem: Novel View Synthesis

3D Photo was Interesting.

In 2018.

An example of posting 3D photo on Facebook timeline. Source: Facebook

Problem: Novel View Synthesis

“Can’t we move around freely?”

“I want to get 360 view of my dog!”

An example of posting 3D photo on Facebook timeline. Source: Facebook

Problem: Novel View Synthesis

An example of posting 3D photo on Facebook timeline. Source: Facebook

"Can't we move around freely?"

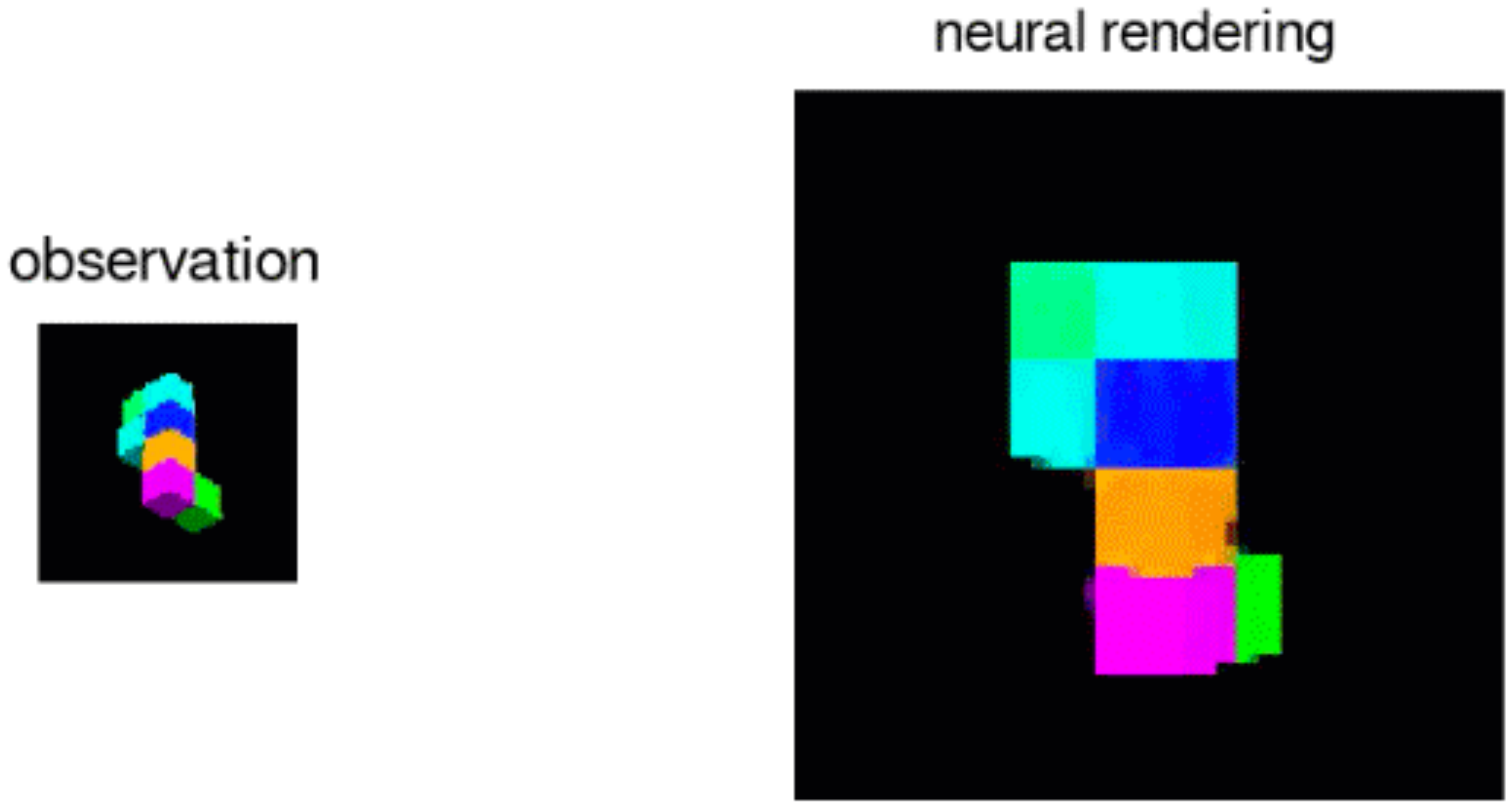
"I want to get 360 view of my dog!"

Interpolating frames of a video wouldn't work

No explicit information of the scene

ALL WE HAVE IS FEW PHOTOS

Problem: Novel View Synthesis

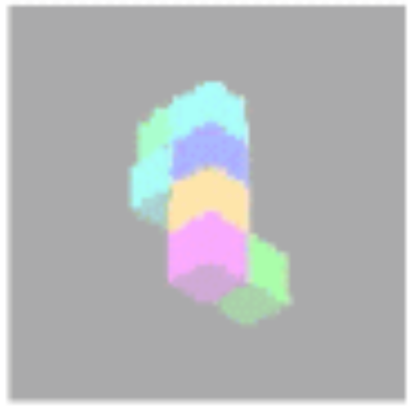


Source: https://github.com/wohlert/generative-query-network-pytorch
Original paper: Neural Scene Representation and Rendering, Eslami et al., Science 2018

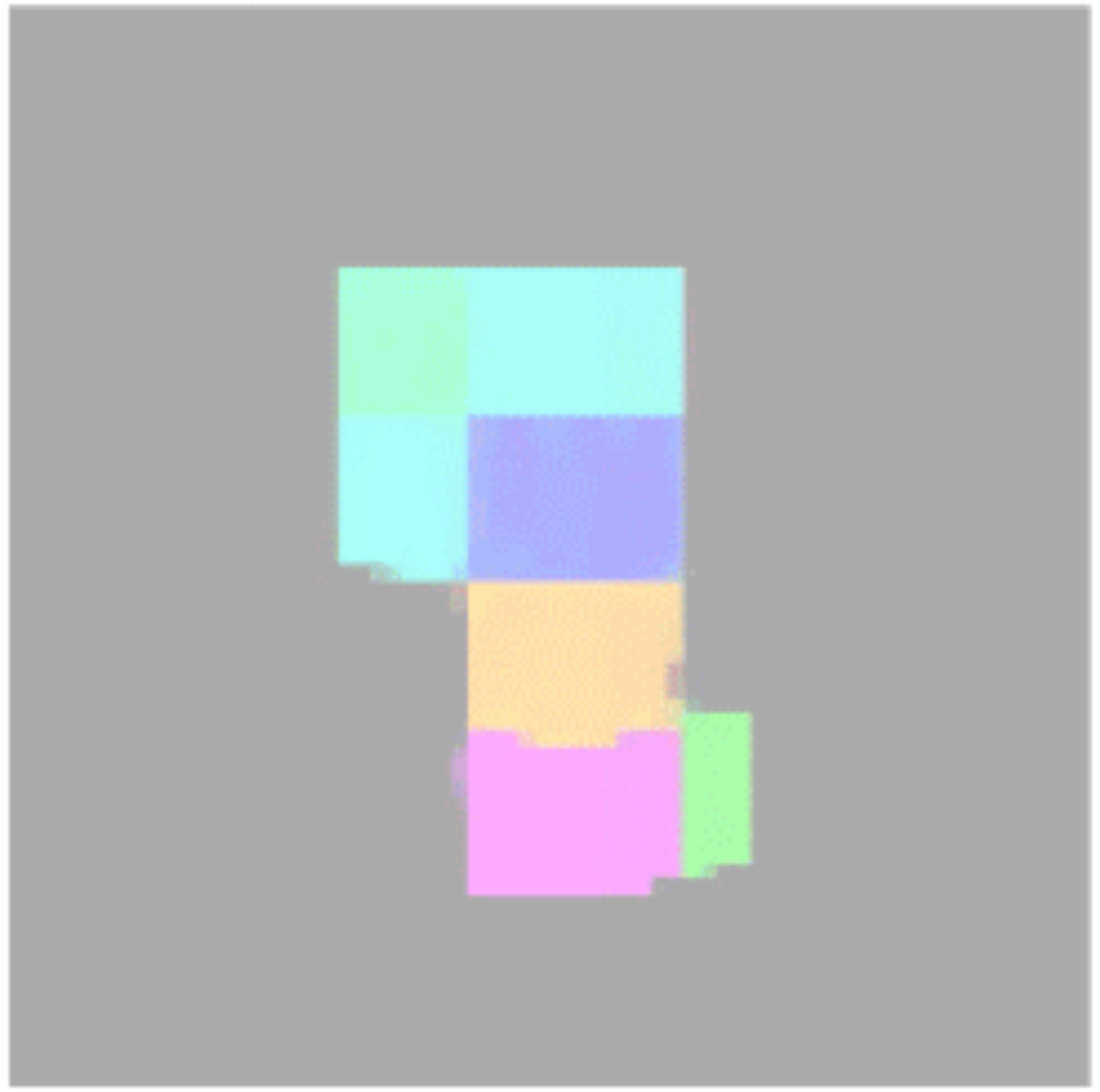
Is it possible to estimate views of a scene seen from unknown viewpoints?

Problem: Novel View Synthesis

observation



neural rendering

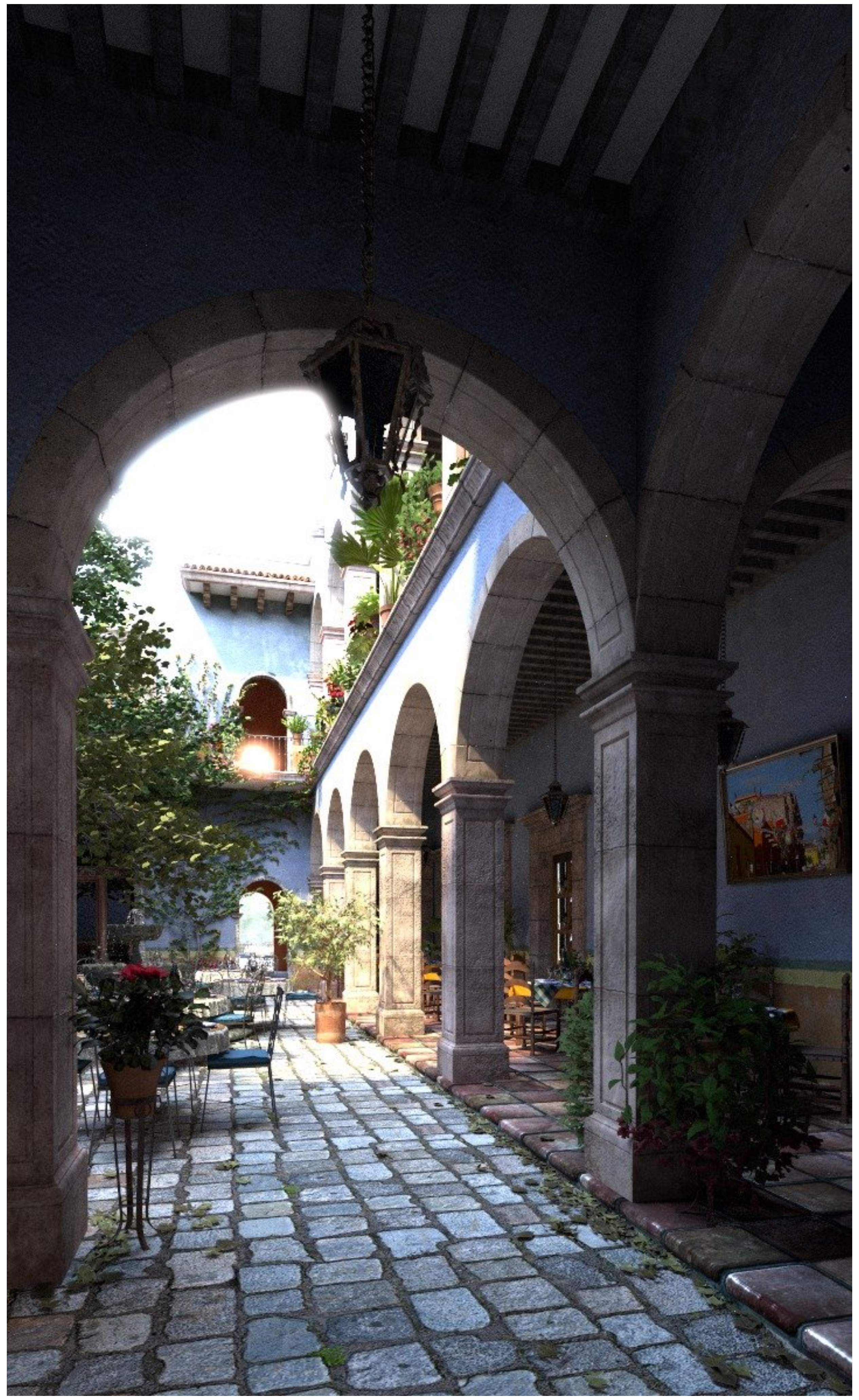


Source: https://github.com/wohlert/generative-query-network-pytorch
Original paper: Neural Scene Representation and Rendering, Eslami et al., Science 2018

Is it possible to estimate views of a scene seen from unknown viewpoints?

Radiance Fields as Scene Representation

Radiance Fields as Scene Representation



San Miguel, Guillermo M. Leal Llaguno. Image from PBRT website

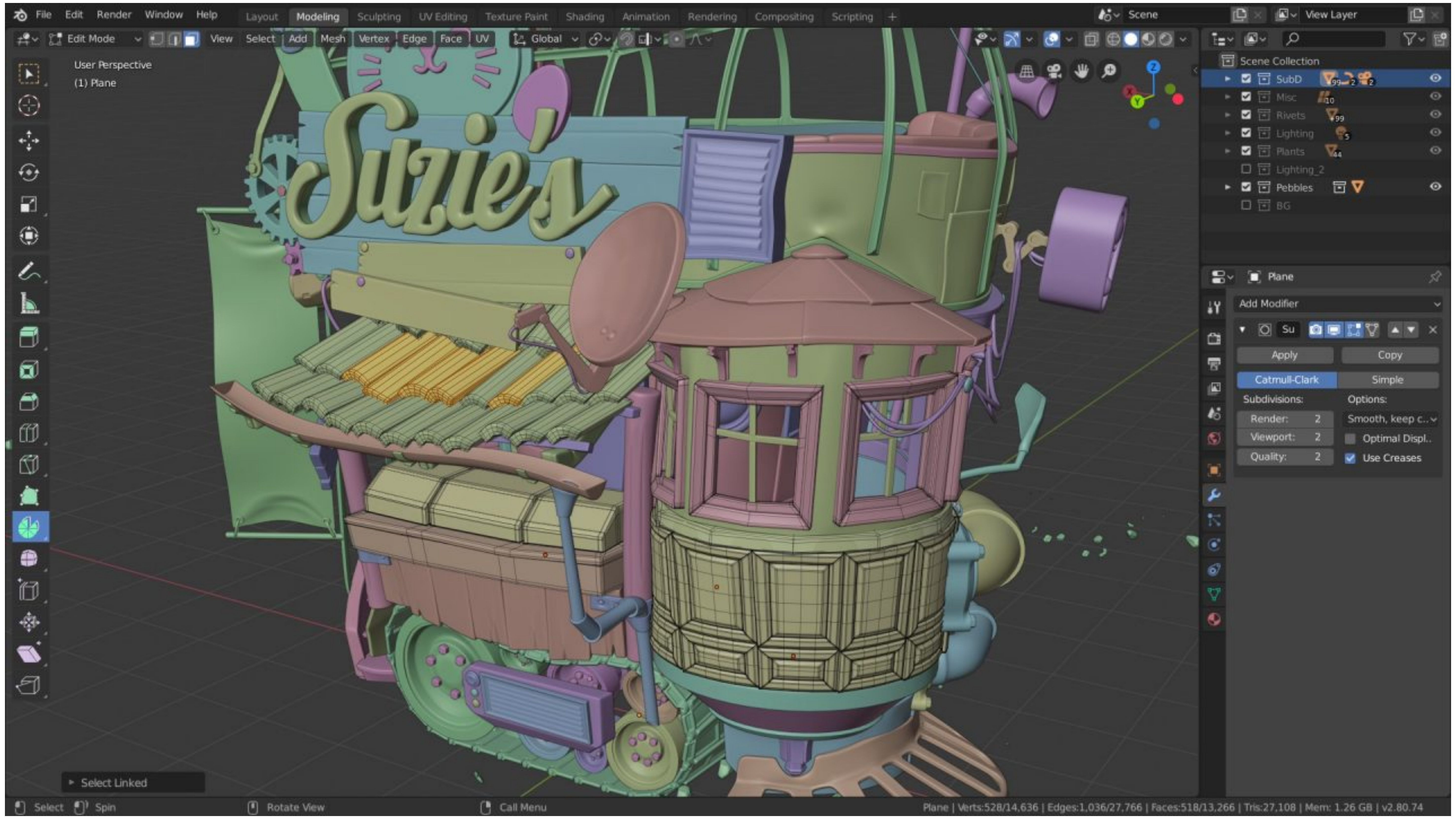
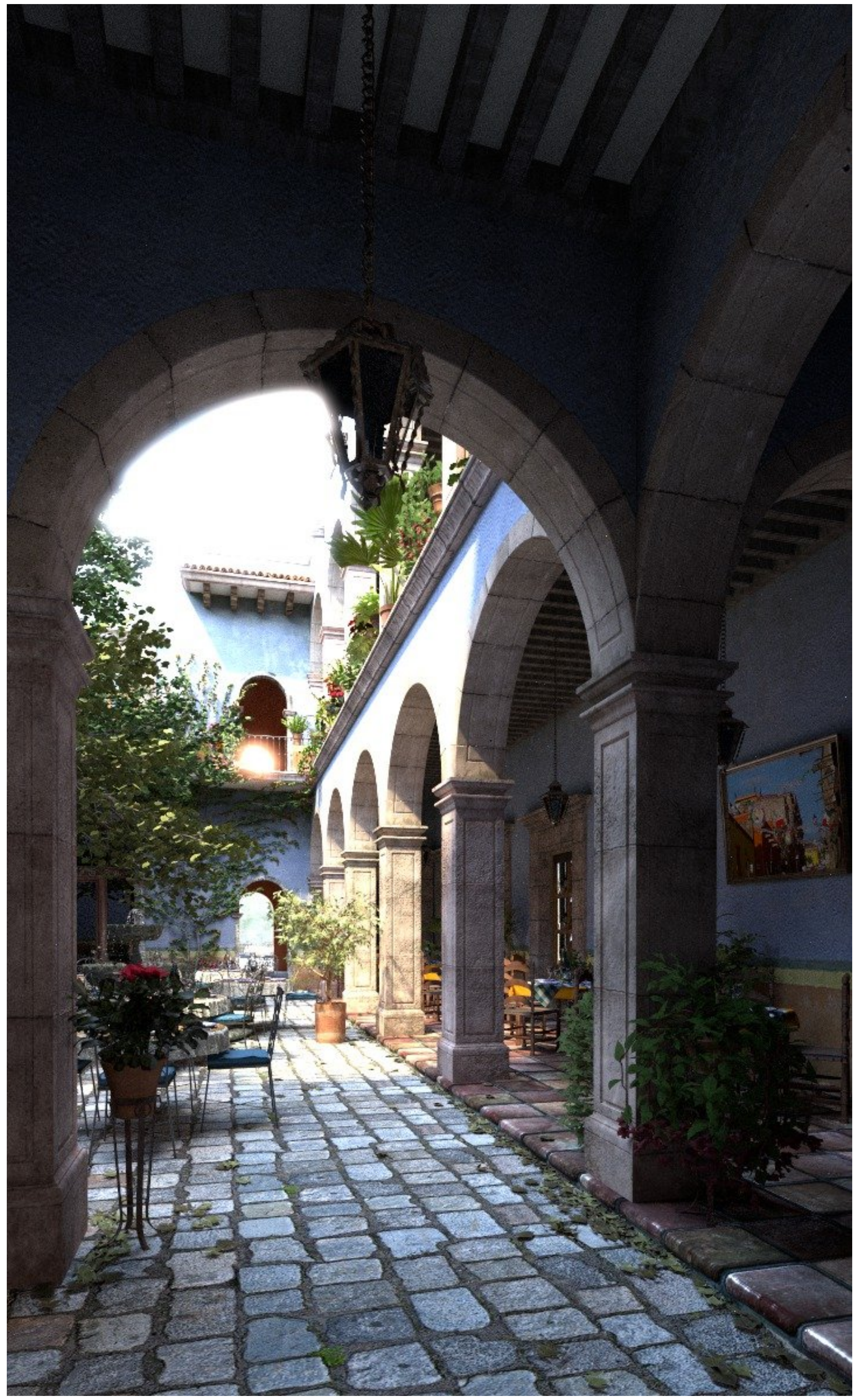


Image from blender.org

Scene = Geometry + Texture + BRDFs + and more!

Radiance Fields as Scene Representation



San Miguel, Guillermo M. Leal Llaguno. Image from PBRT website

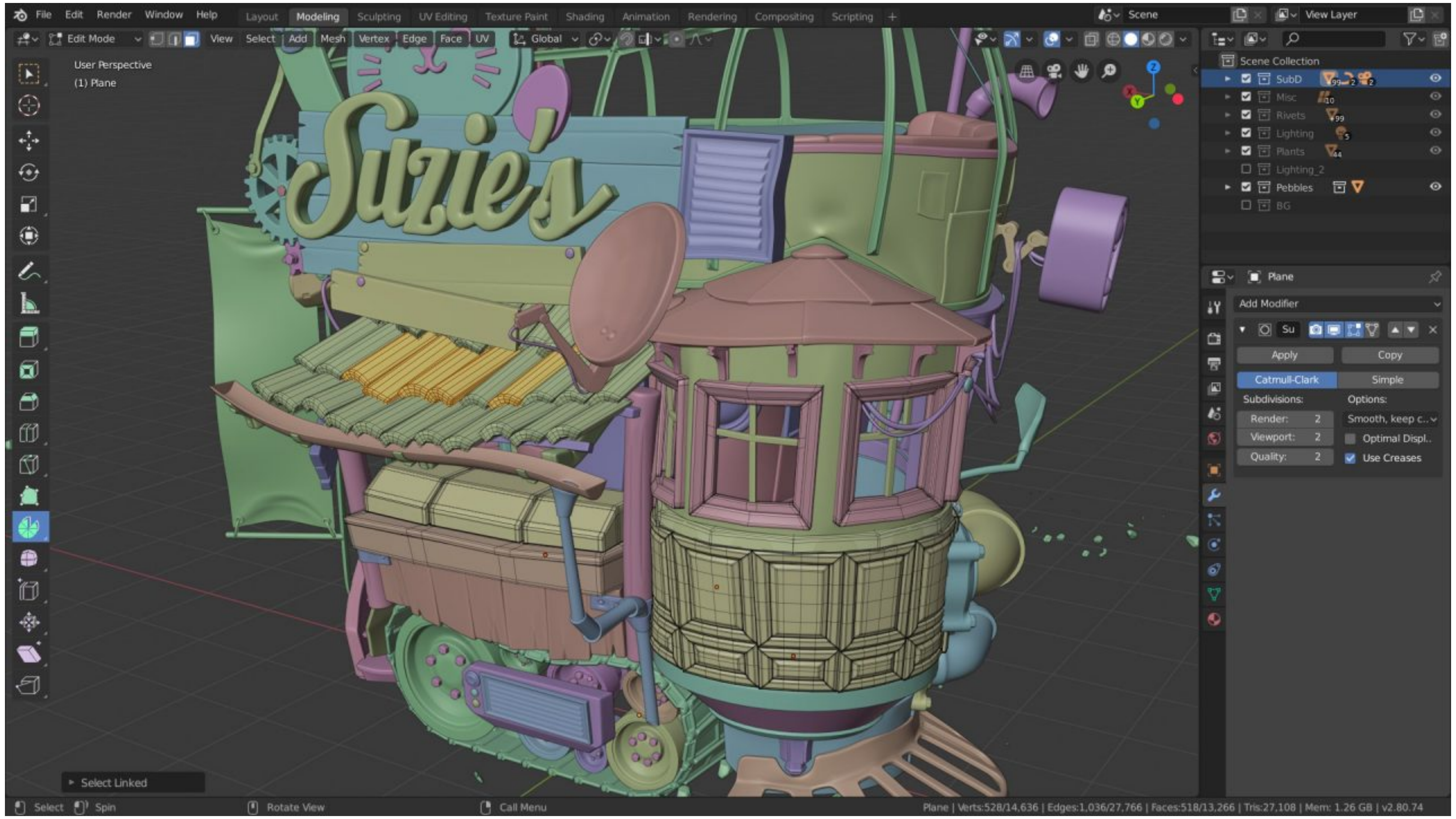


Image from blender.org

<u>It's just one possible representation of a scene</u>

Radiance Fields as Scene Representation

$$\mathbf{F}(\mathbf{x}, \mathbf{v}) : \mathbb{R}^5 \rightarrow \mathbb{R}^3$$

A function which maps **3D position** and **2D viewing direction** to **3D vector in color space**.

Imagine infinitely many light bulbs filling space

Each light bulb looks differently depending on your viewpoint

Imagine infinitely many light bulbs filling space

Each light bulb looks differently depending on your viewpoint

$$\mathbf{F}(\mathbf{x}, \mathbf{v}) : \mathbb{R}^5 \rightarrow \mathbb{R}^3$$

Radiance Fields as Scene Representation

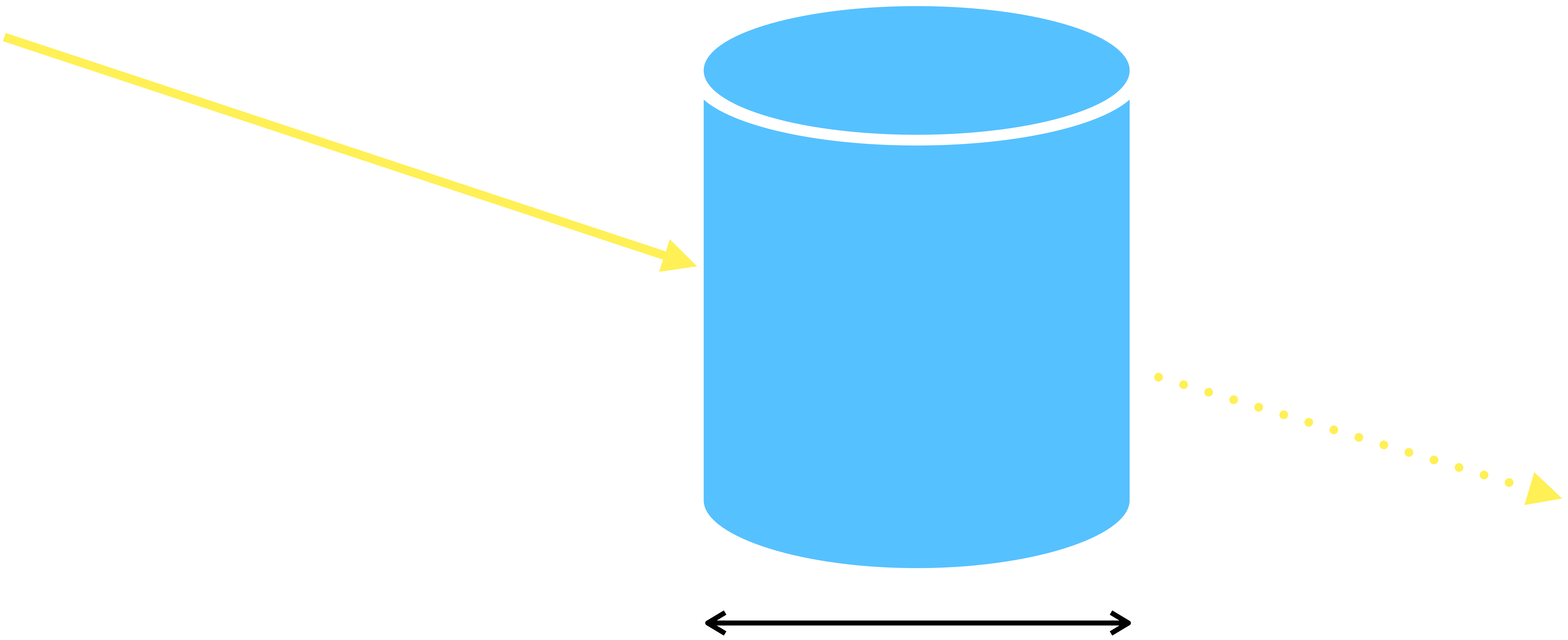
$$\sigma(\mathbf{x}) : \mathbb{R}^3 \rightarrow \mathbb{R}$$

Each point is assigned specific density (i.e. opacity) value

The higher the density, the harder for light to pass through

Models “occlusion”

Radiance Fields as Scene Representation



Matters are **concentrated**

Rays are likely to be reflected, absorbed at the surface

Radiance Fields as Scene Representation

$$\mathbf{F}(\mathbf{x}, \mathbf{v}) : \mathbb{R}^5 \rightarrow \mathbb{R}^3$$

$$\sigma(\mathbf{x}) : \mathbb{R}^3 \rightarrow \mathbb{R}$$

Radiance Fields as Scene Representation

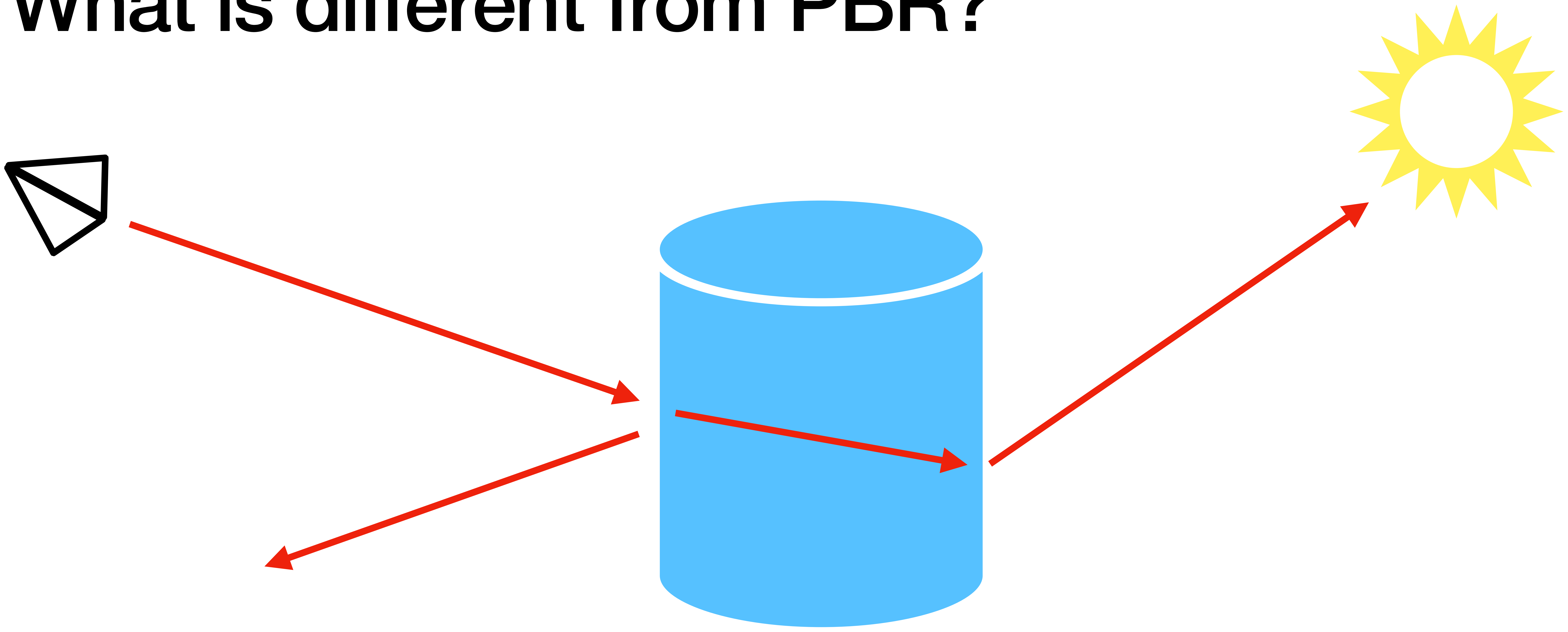
$$\mathbf{F}(\mathbf{x}, \mathbf{v}) : \mathbb{R}^5 \rightarrow \mathbb{R}^3$$

$$\sigma(\mathbf{x}) : \mathbb{R}^3 \rightarrow \mathbb{R}$$

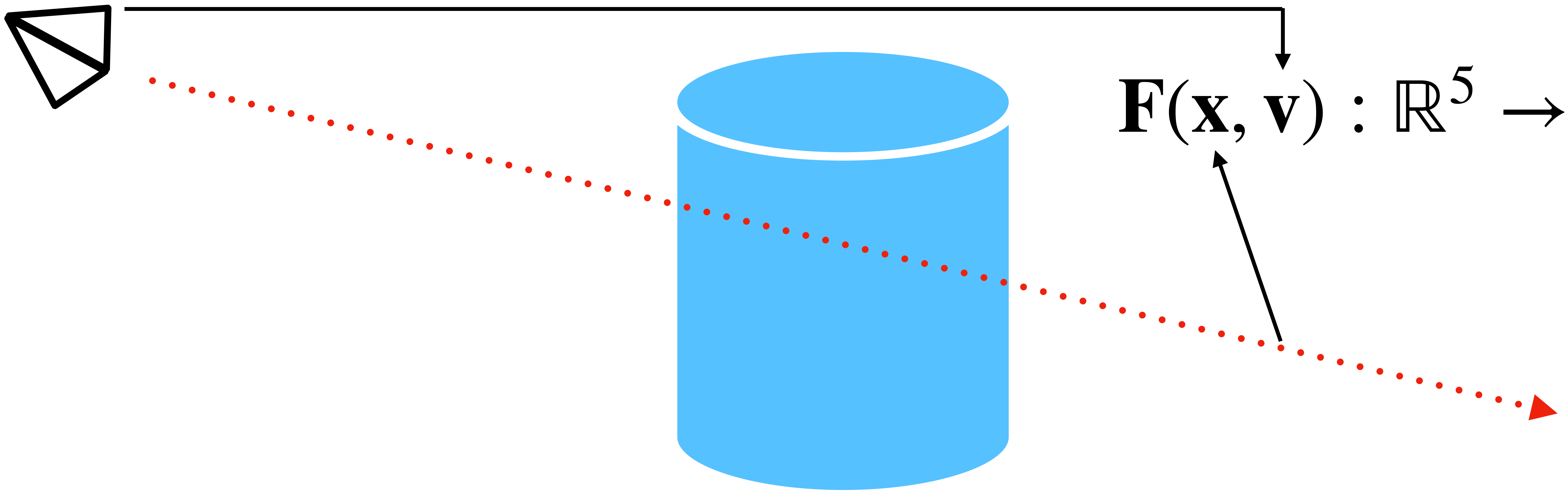
How to render an image?

Volume Rendering.
What is different from PBR?

Volume Rendering.
What is different from PBR?



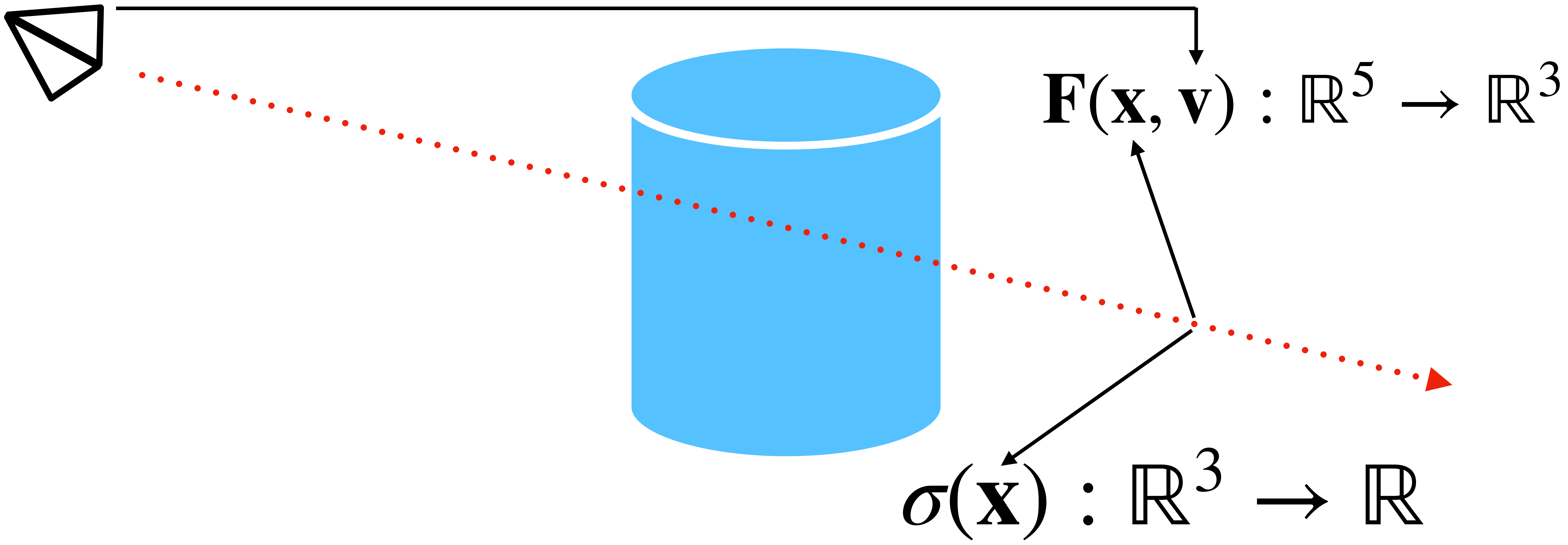
Volume Rendering.
What is different from PBR?



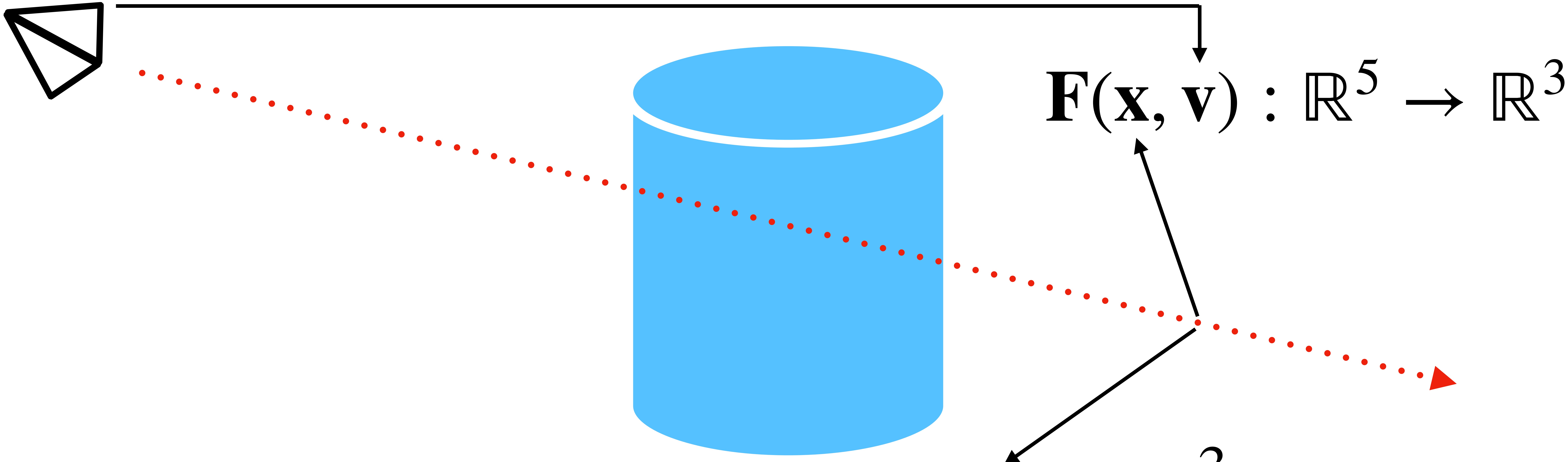
$$\mathbf{F}(\mathbf{x}, \mathbf{v}) : \mathbb{R}^5 \rightarrow \mathbb{R}^3$$

Every point is individual radiance source!

Volume Rendering.
What is different from PBR?



Volume Rendering.
What is different from PBR?

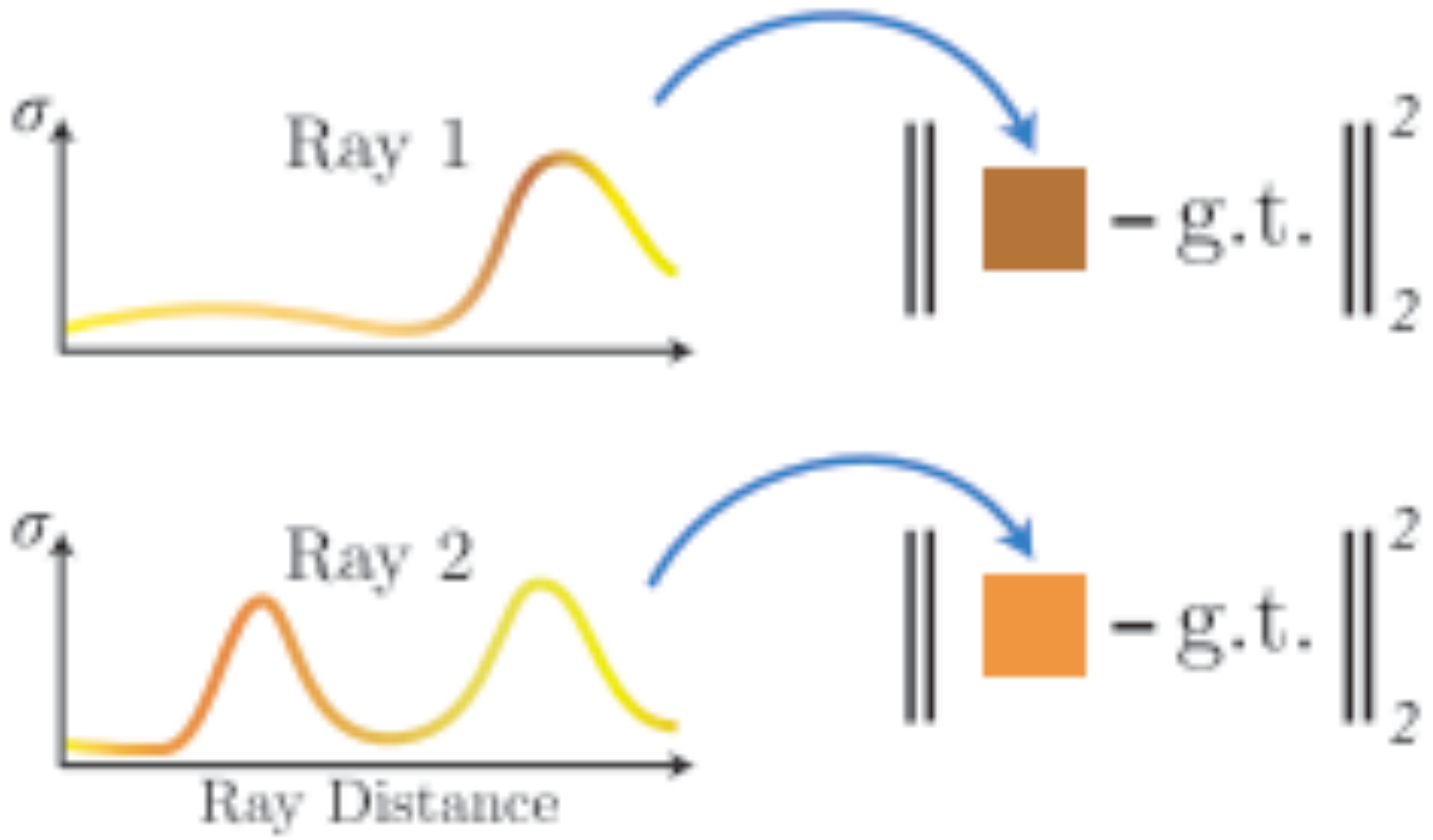


$$\mathbf{F}(\mathbf{x}, \mathbf{v}) : \mathbb{R}^5 \rightarrow \mathbb{R}^3$$

$$\sigma(\mathbf{x}) : \mathbb{R}^3 \rightarrow \mathbb{R}$$

How to determine the contribution of each sample to the final radiance?

Volume Rendering.
What is different from PBR?



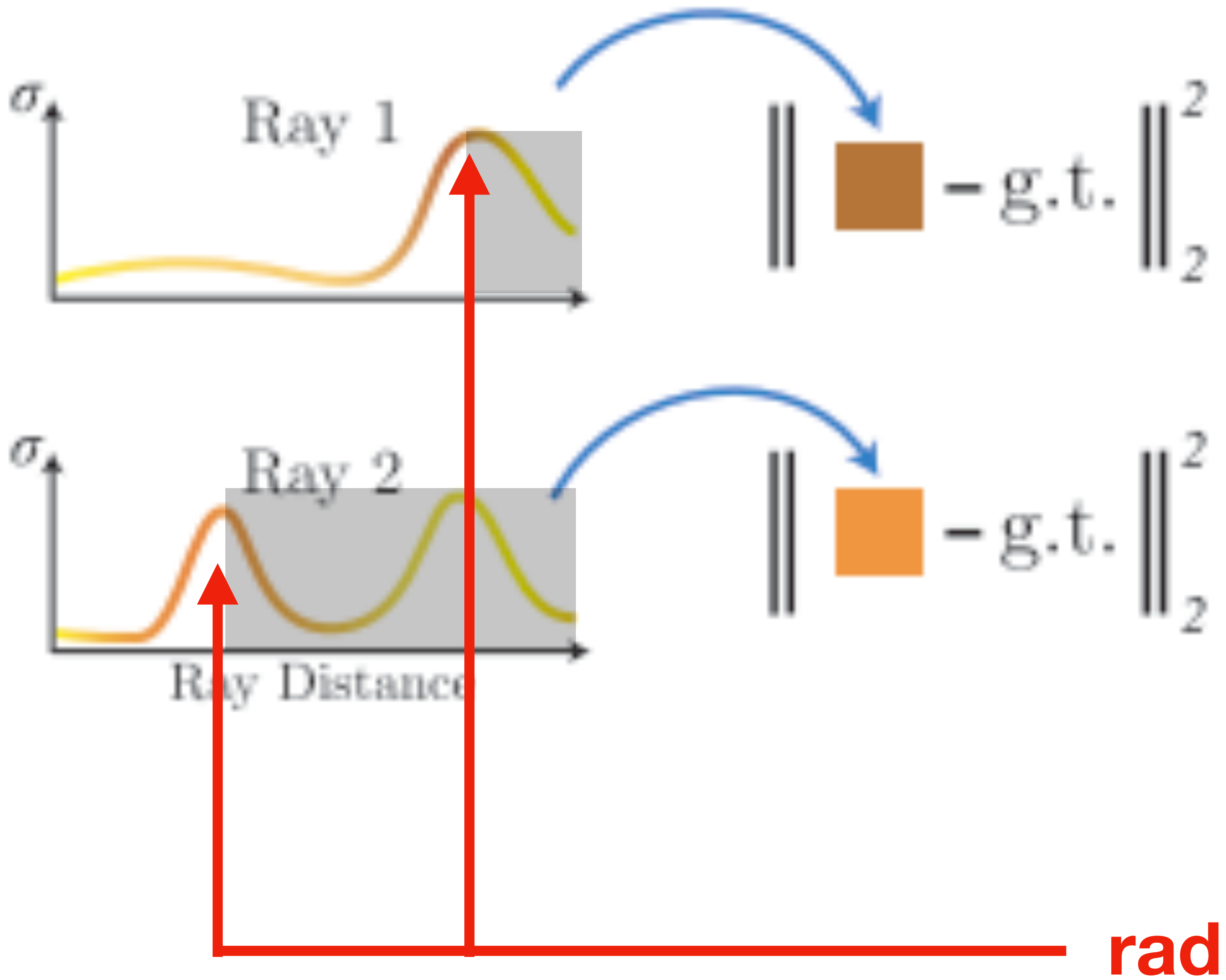
ray — density — radiance field — direction

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

where $T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$

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

Volume Rendering.
What is different from PBR?



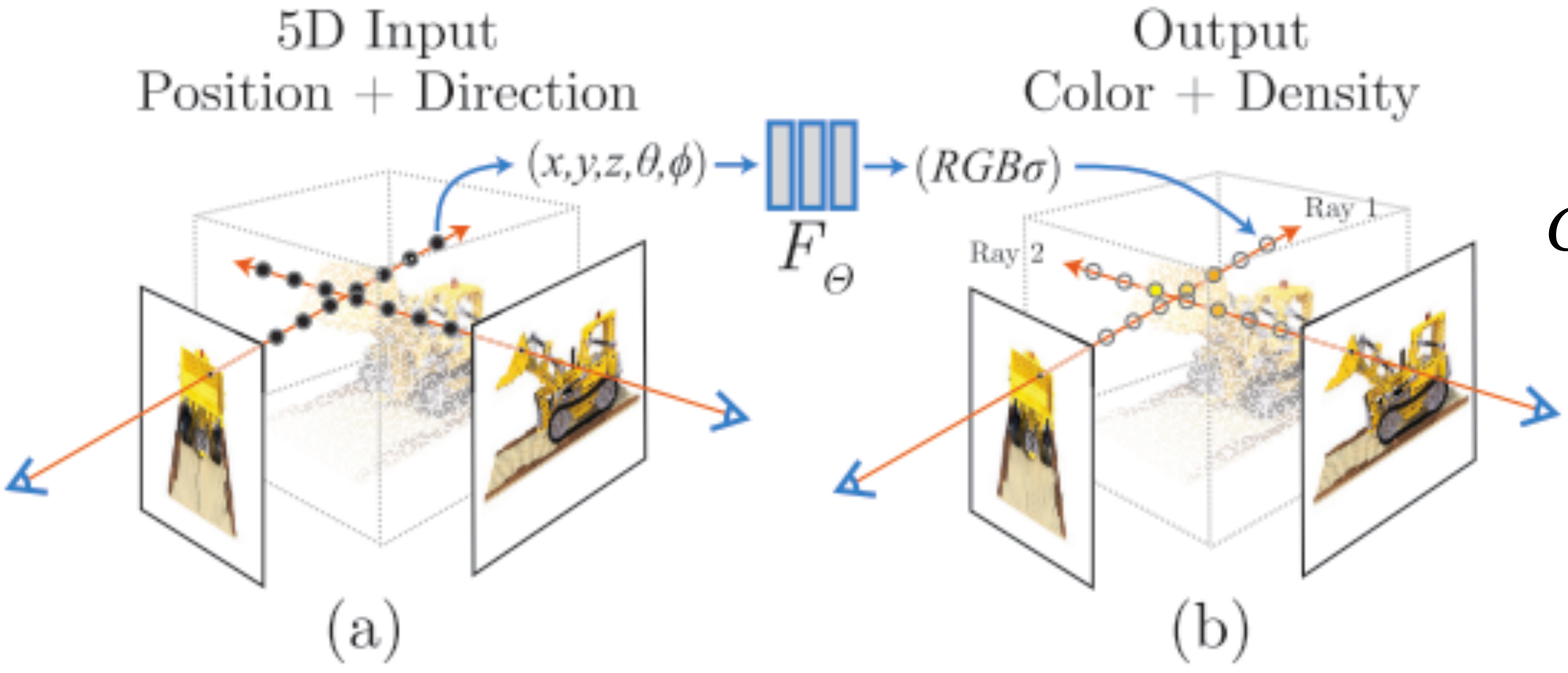
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

where $T(t) = \exp\left(- \underline{\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds} \right)$

accumulates density values encountered until t

radiance after this point will barely contribute to the final output!

Volume Rendering.
What is different from PBR?

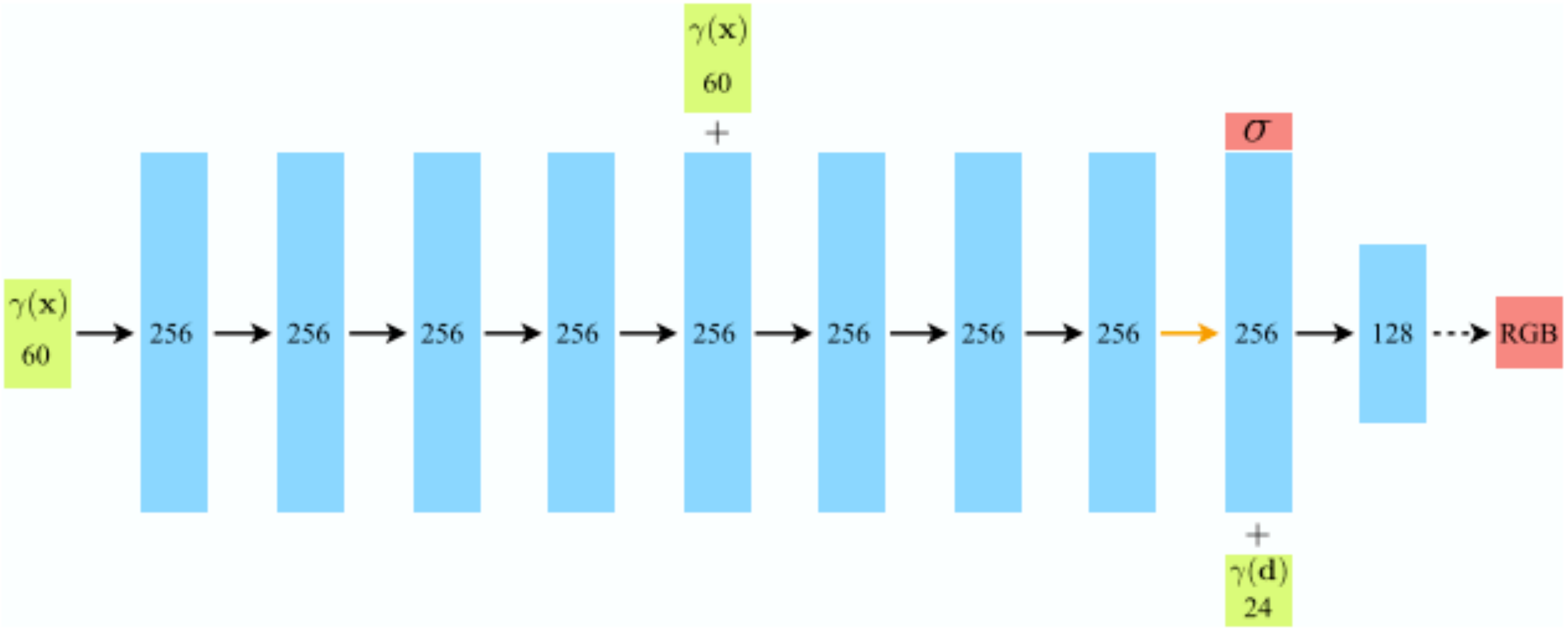


$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

where $T(t) = \exp\left(- \int_{t_n}^{t} \sigma(\mathbf{r}(s))ds \right)$

Learning Radiance Fields by Minimizing Loss Function

Learning Radiance Fields by Minimizing Loss Function

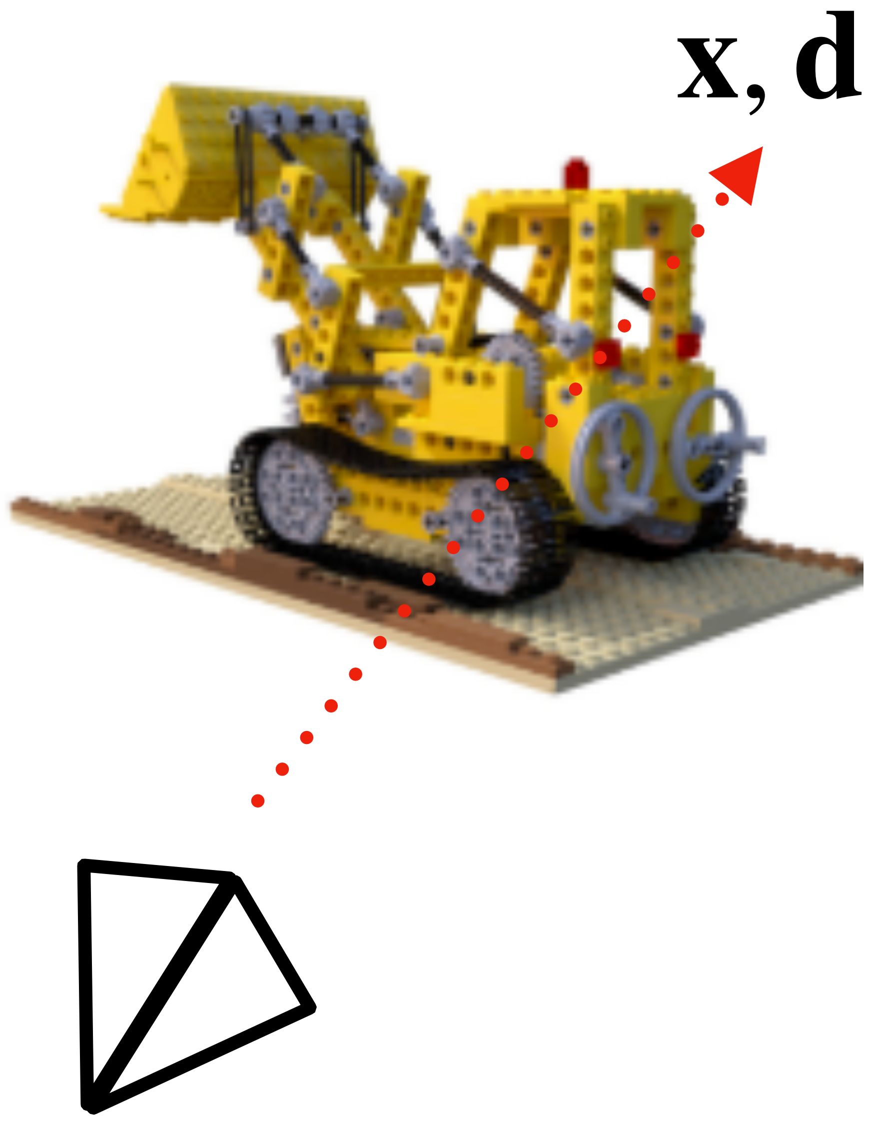


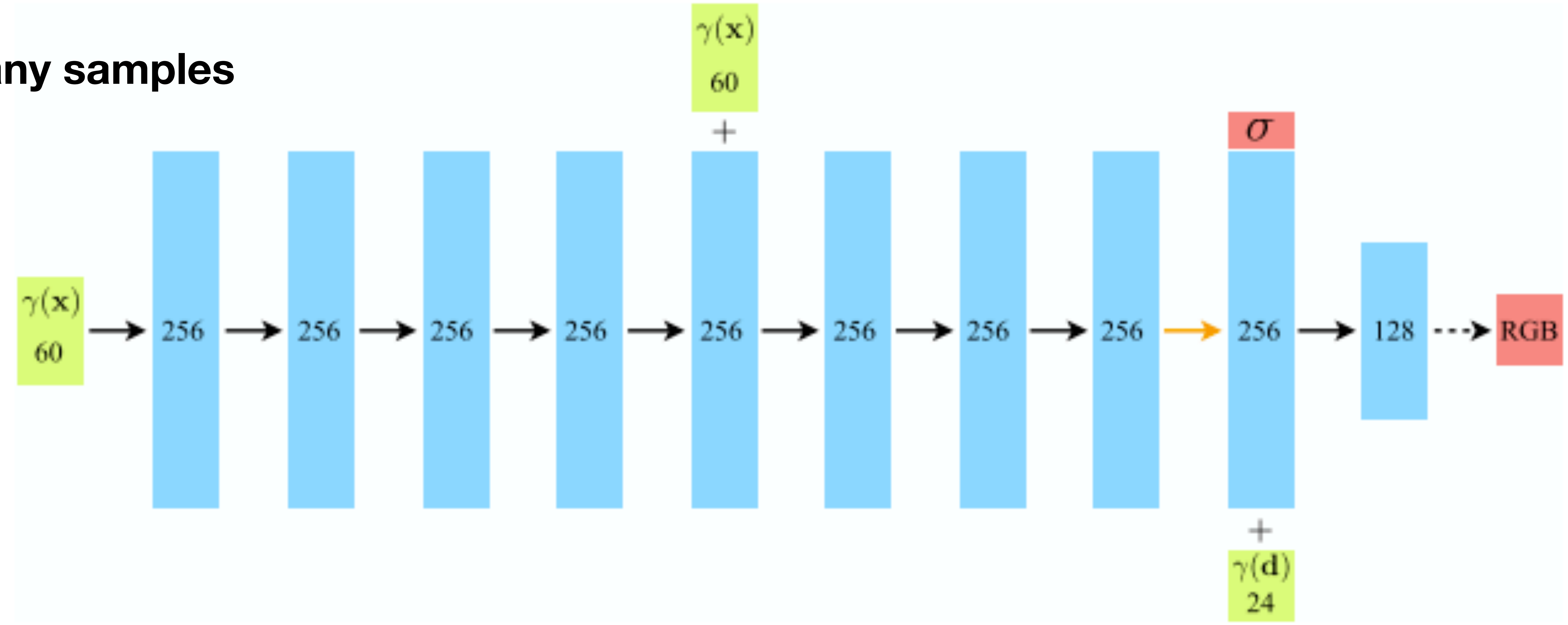
8 fully-connected layers (i.e., linear transformations)

This network approximates a radiance field which maps 3D + 2D input to:
3D color vector & scalar density value

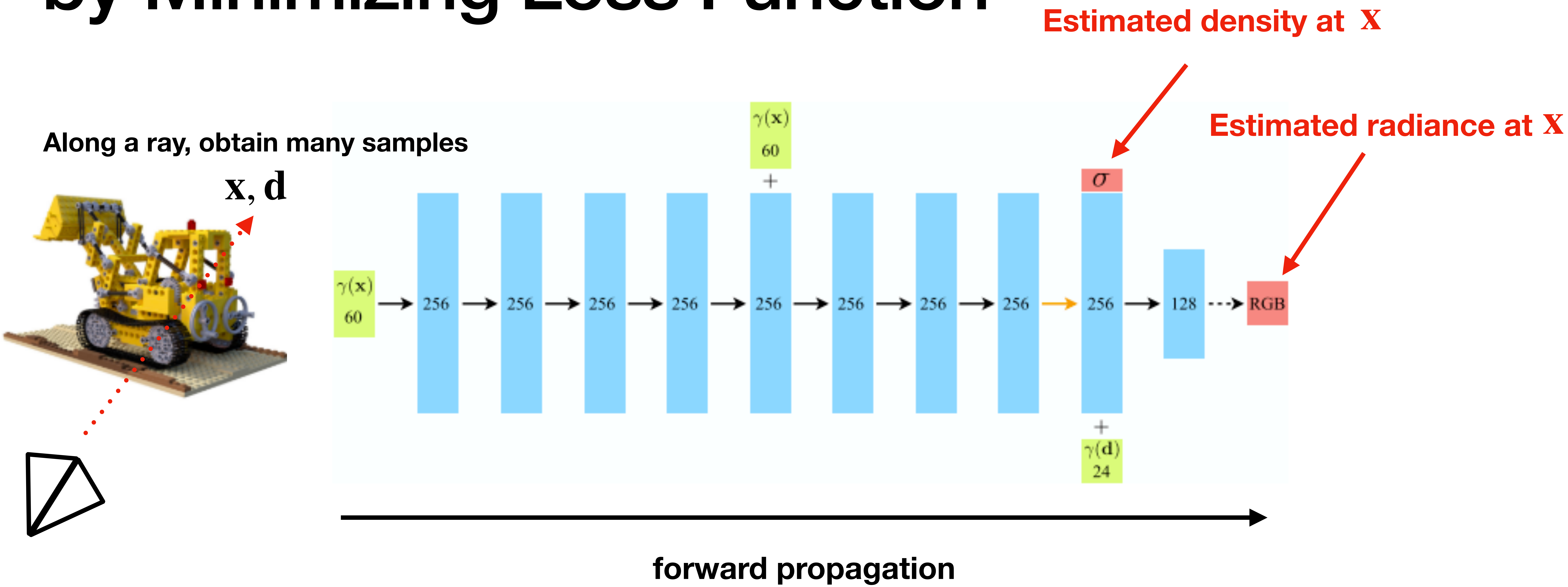
Learning Radiance Fields by Minimizing Loss Function

Along a ray, obtain many samples



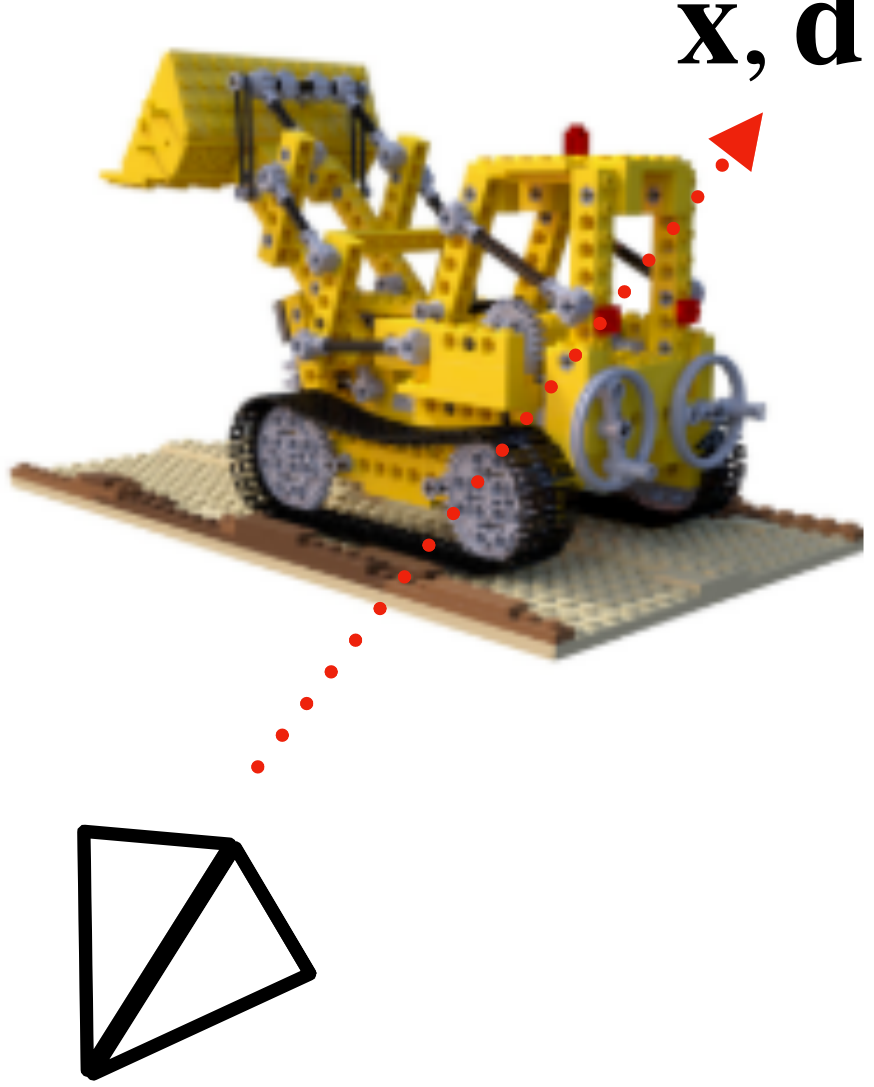


Learning Radiance Fields by Minimizing Loss Function



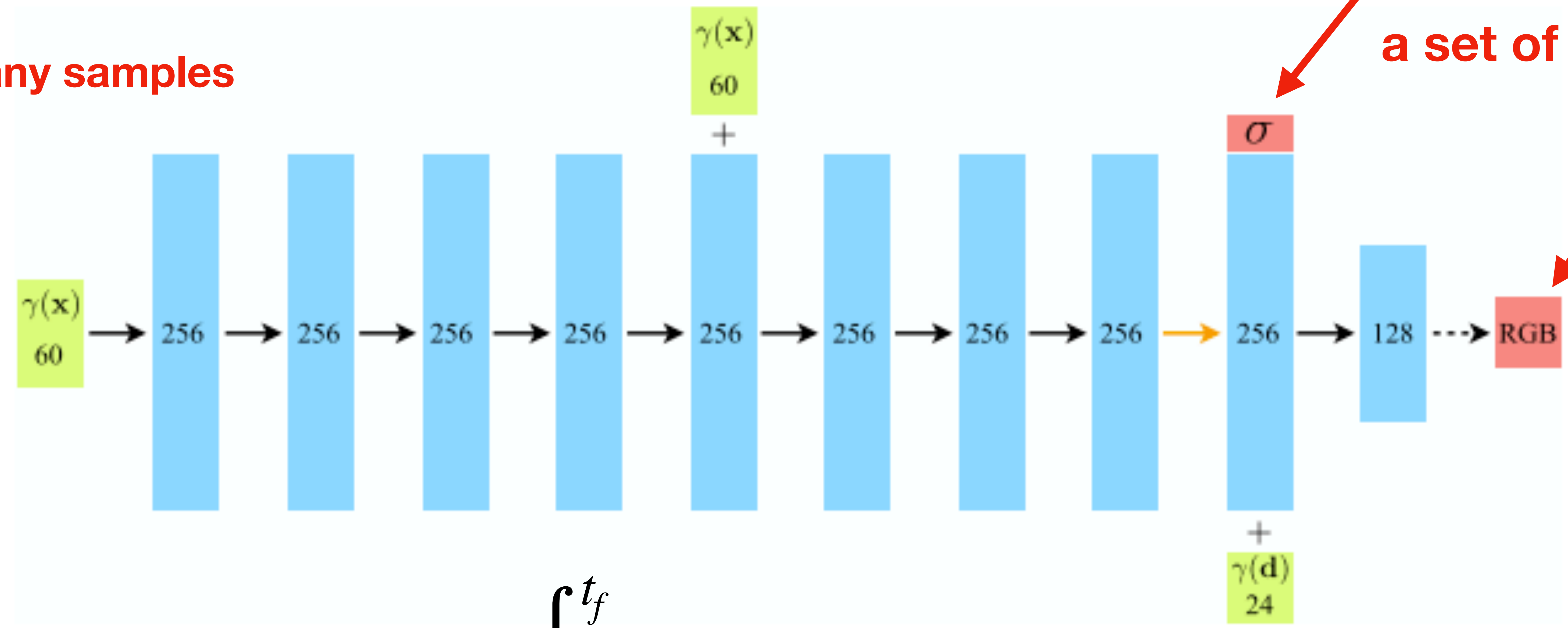
Learning Radiance Fields by Minimizing Loss Function

Along a ray, obtain many samples



a set of $\sigma(\mathbf{r}(t))$

a set of $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$



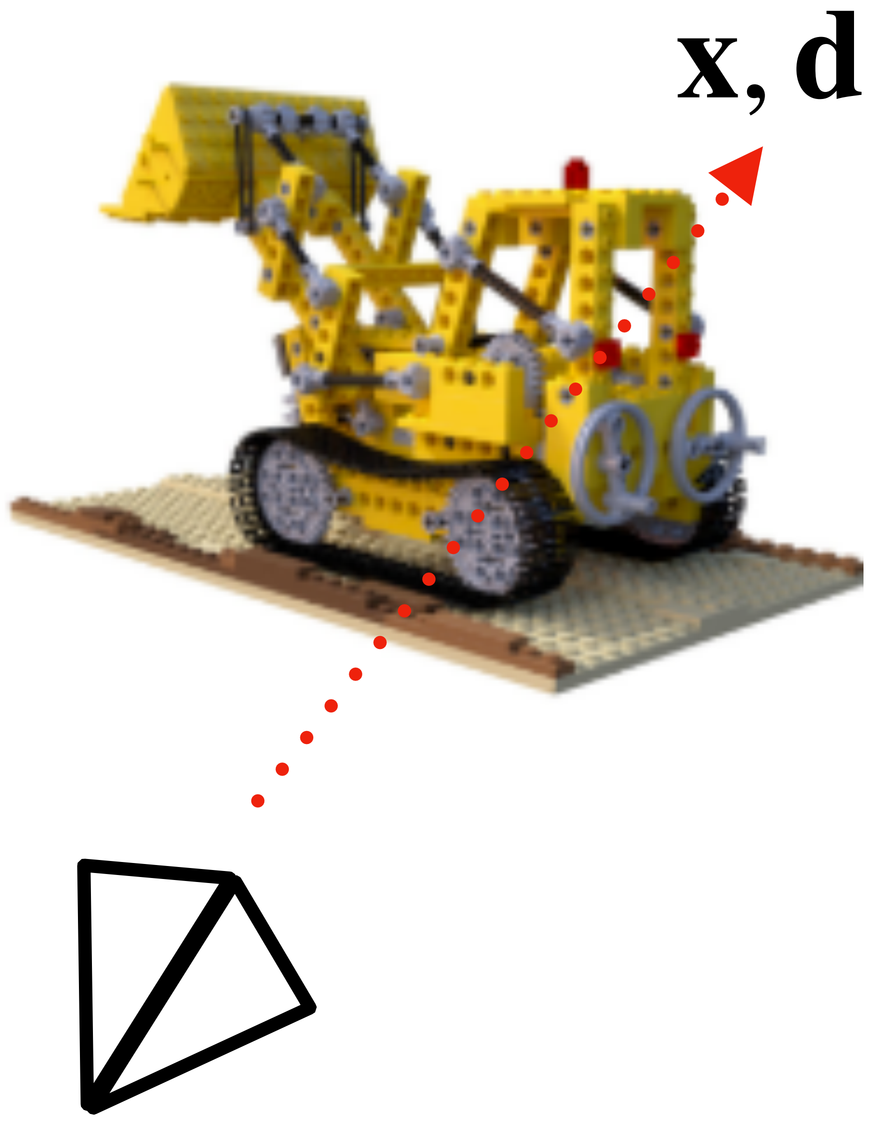
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

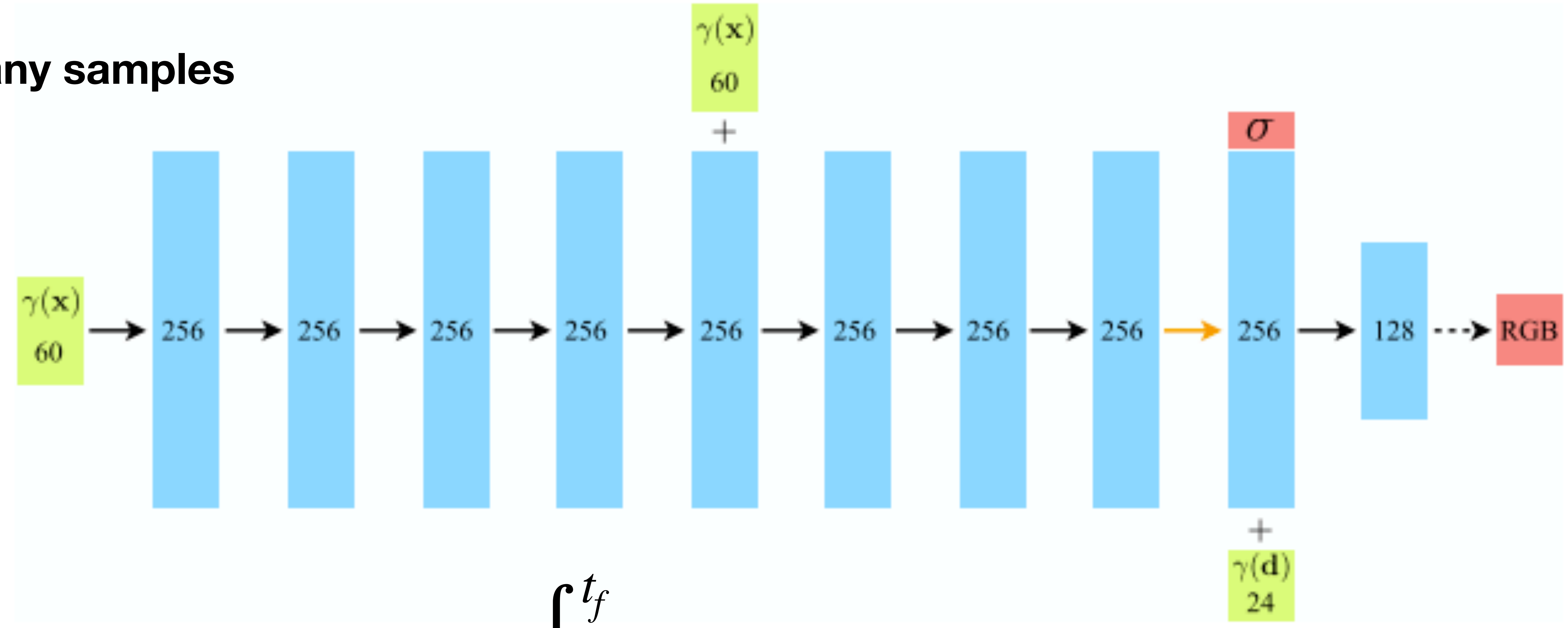
where $$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds \right)$$

Using multiple radiance samples, determine the pixel color!

Learning Radiance Fields by Minimizing Loss Function

Along a ray, obtain many samples





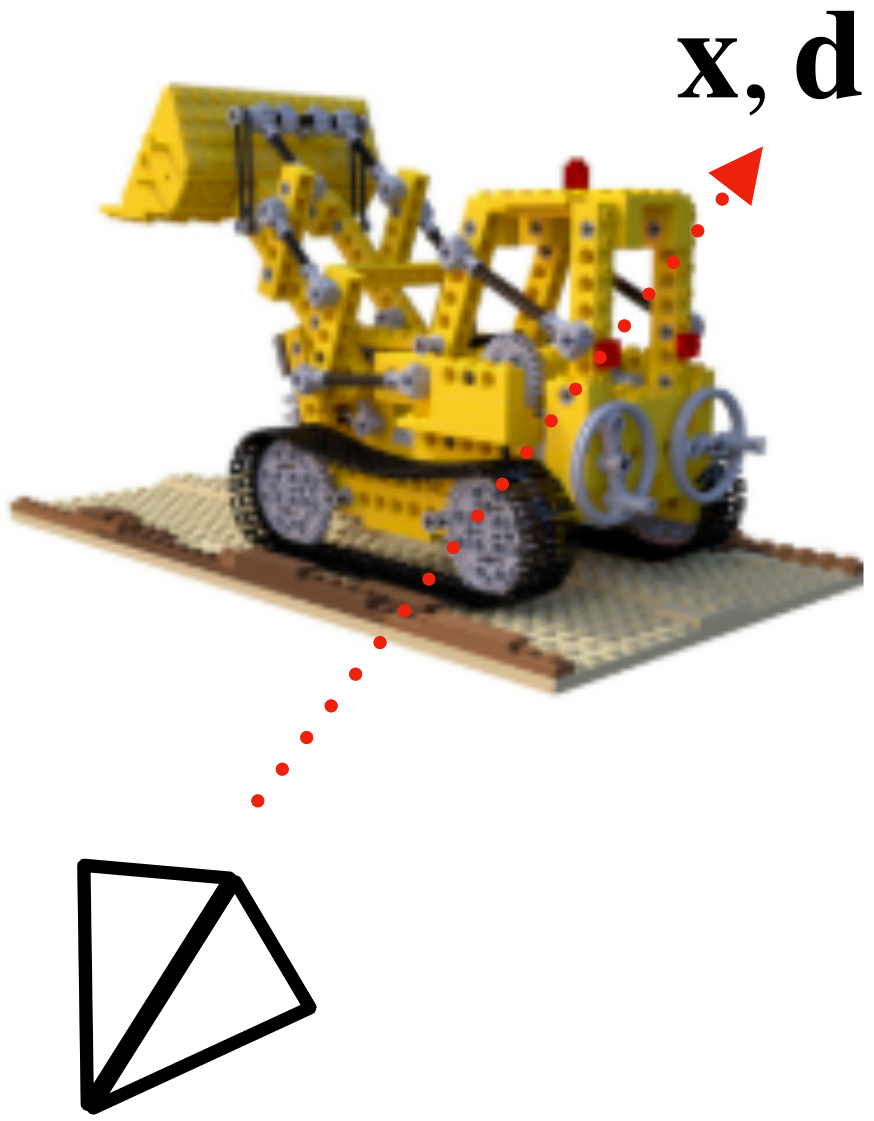
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

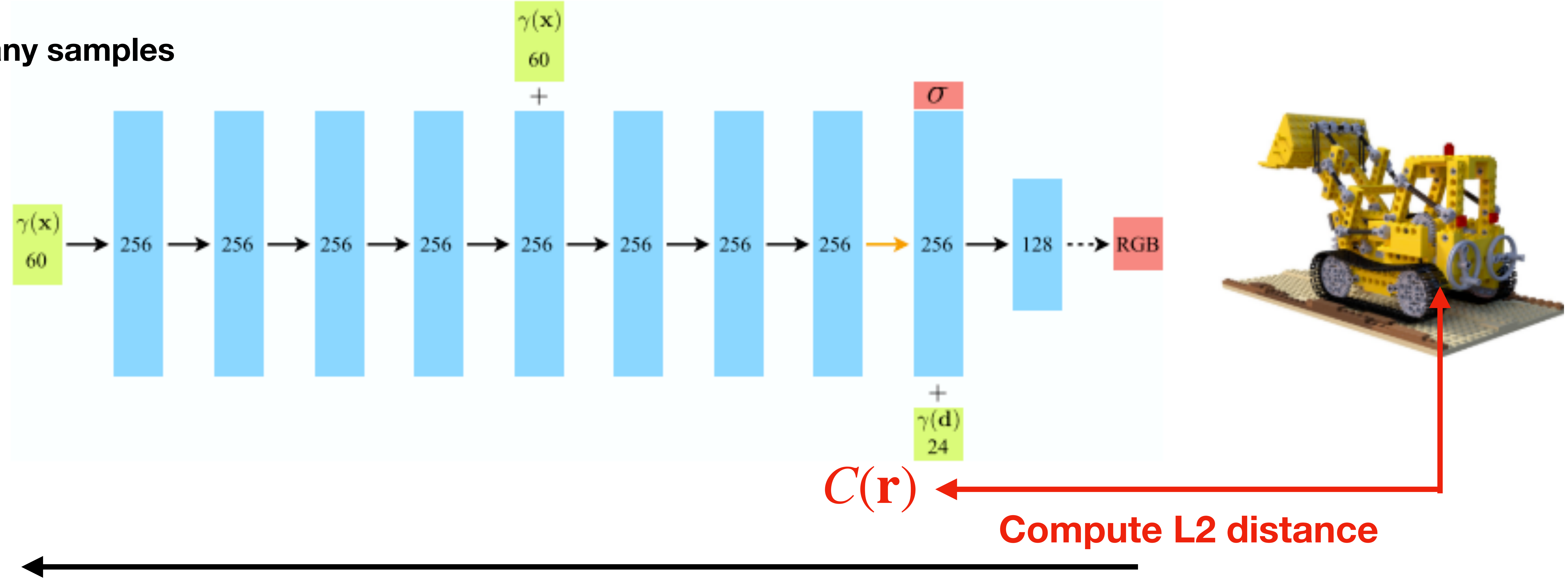
Compute L2 distance

where $T(t) = \exp\left(- \int_{t_n}^{t} \sigma(\mathbf{r}(s))ds \right)$

Learning Radiance Fields by Minimizing Loss Function

Along a ray, obtain many samples

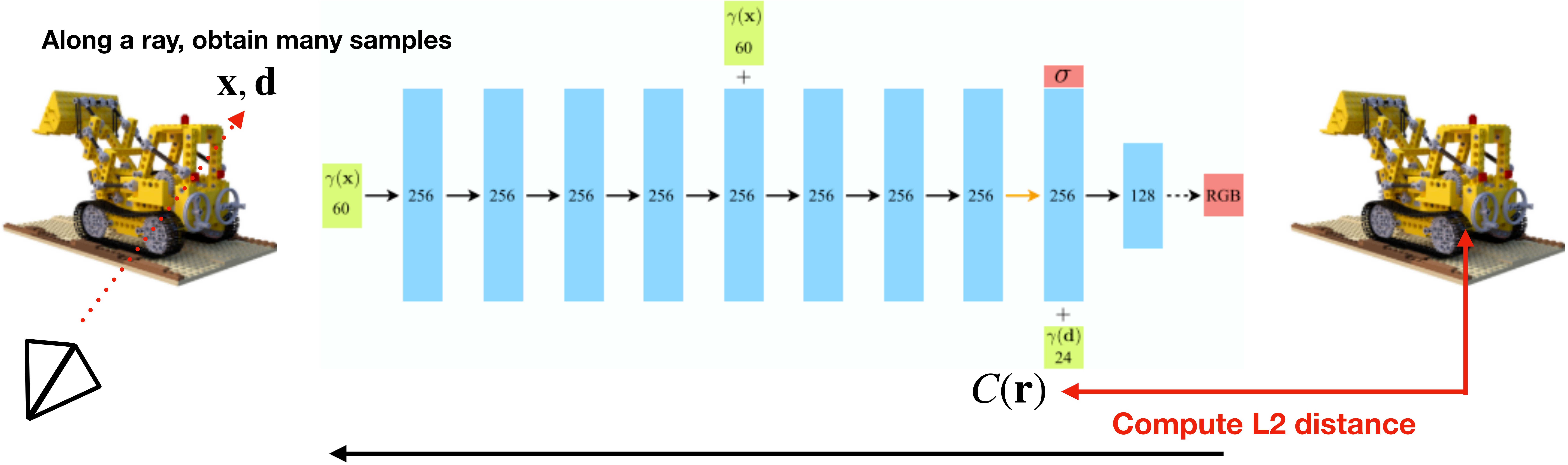




Update parameters of the neural network
in a way it **minimizes the L2 distance**

a.k.a back propagation

Learning Radiance Fields by Minimizing Loss Function



As training progresses, the **neural network “learns” the radiance field by minimizing the difference between known & synthesized images!**

Yet Important Ideas Remain

Yet Important Ideas Remain

Positional Encoding

For each component of $\mathbf{x} = (x, y, z),\ \mathbf{d} = (u, v)$

Define mapping $\gamma : \mathbb{R} \rightarrow \mathbb{R}^{2L}$

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

Yet Important Ideas Remain

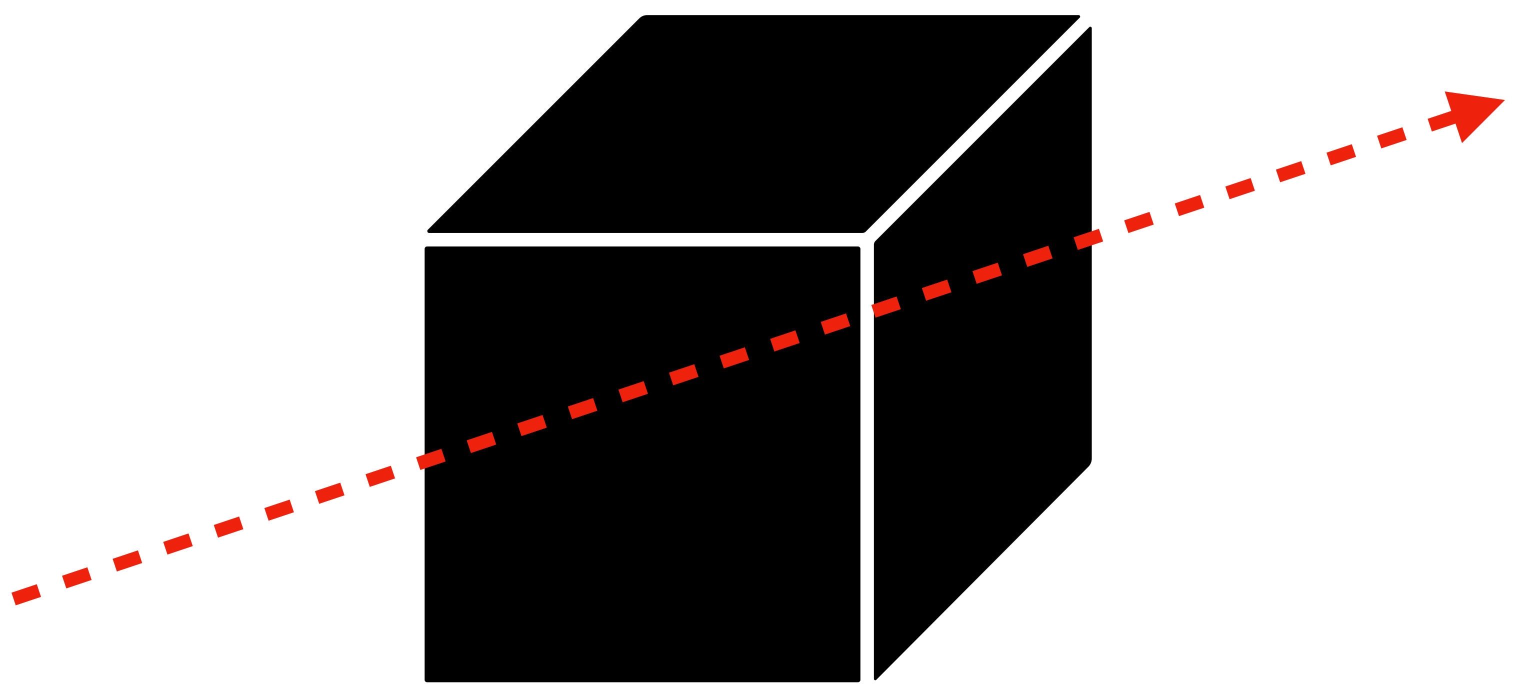
Positional Encoding

For each component of $\mathbf{x} = (x, y, z),\ \mathbf{d} = (u, v)$

Define mapping $\gamma : \mathbb{R} \to \mathbb{R}^{2L}$

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

Hierarchical Sampling



Yet Important Ideas Remain

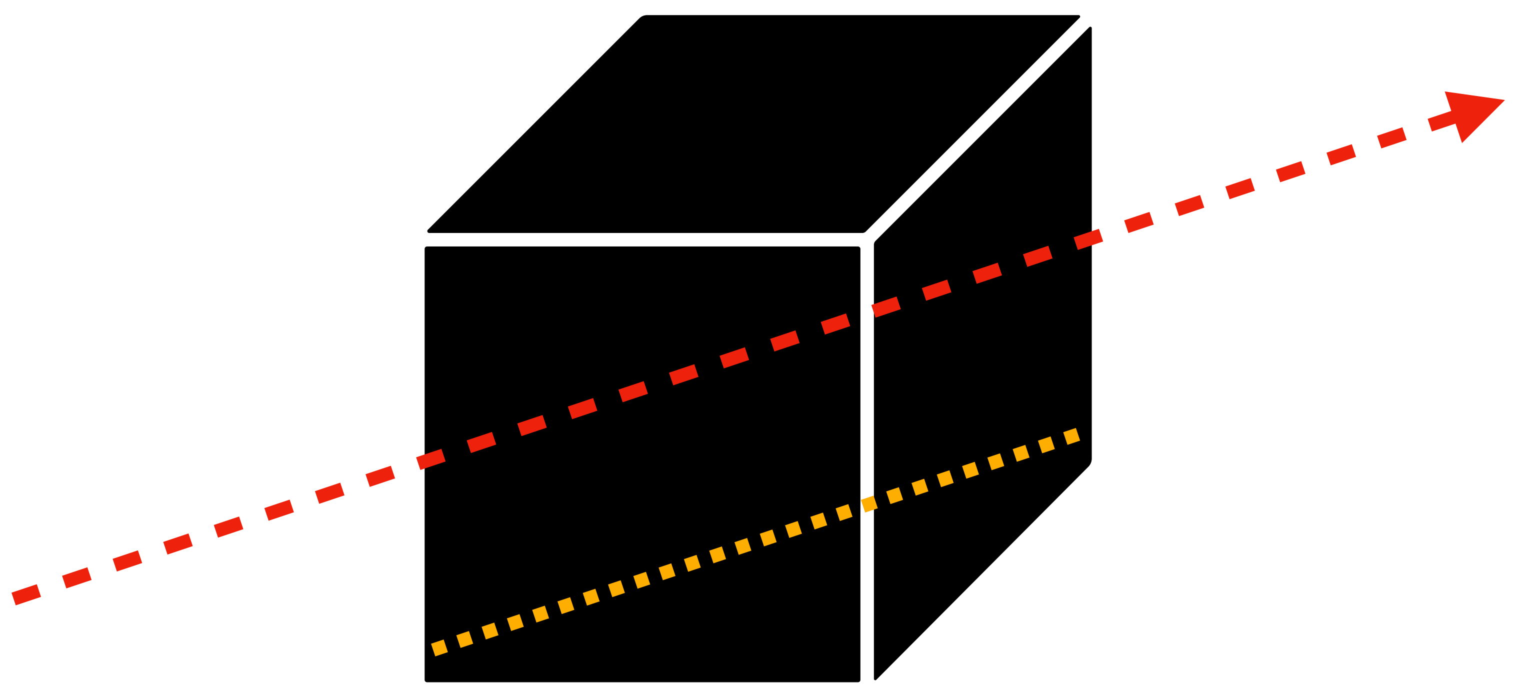
Positional Encoding

For each component of $\mathbf{x} = (x, y, z)$, $\mathbf{d} = (u, v)$

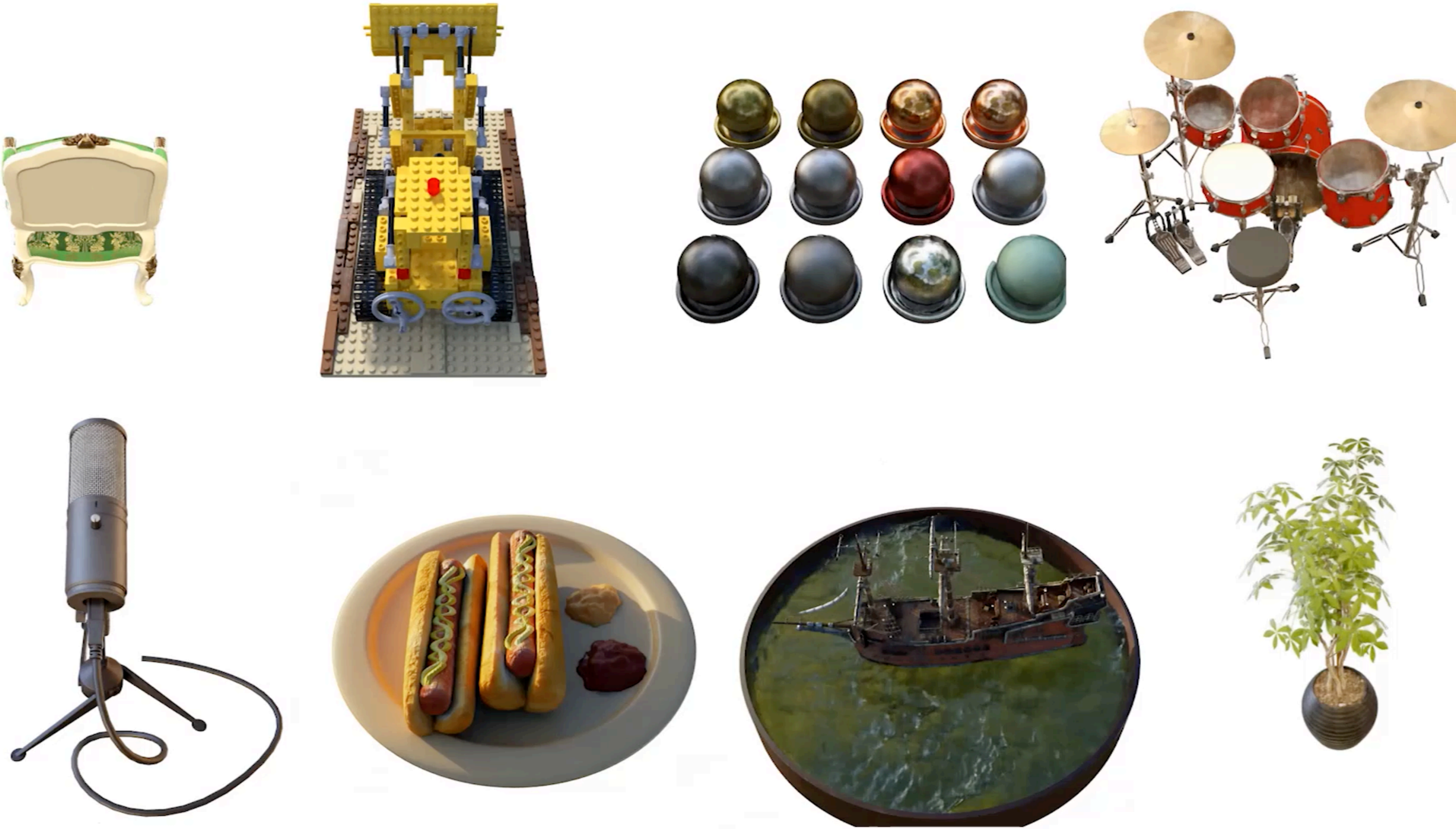
Define mapping $\gamma : \mathbb{R} \rightarrow \mathbb{R}^{2L}$

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

~ Importance Sampling
Hierarchical Sampling



Qualitative Results

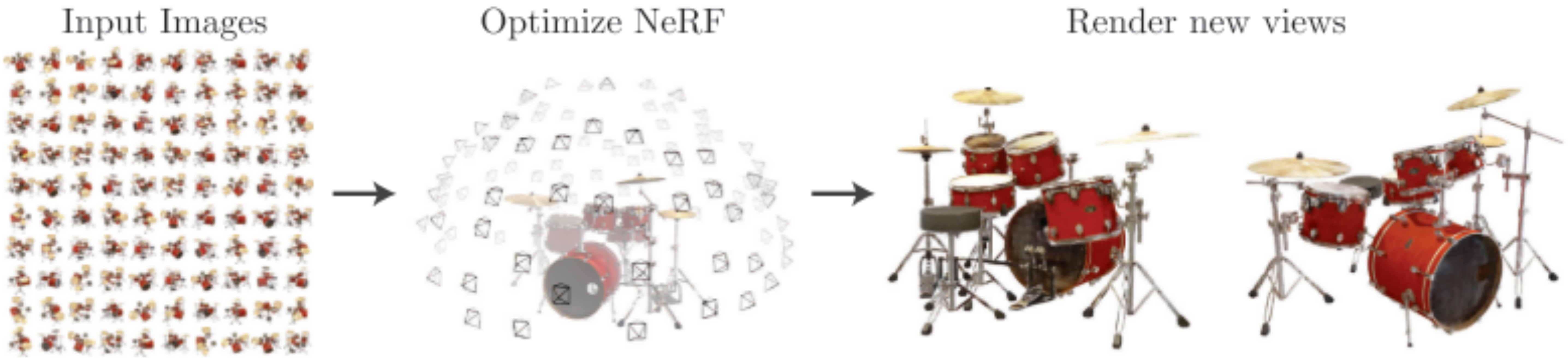



NOR-CAL
MOVING SERVICES


NOR-CAL
MOVING SERVICES


Closing Remark

Closing Remark



NeRF optimizes radiance field by minimizing error between known image and predicted image

A radiance field is approximated as a simple neural network, and optimized using deep learning methods

With NeRF, we can generate 360 videos from sets of images taken with calibrated cameras

Closing Remark

NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections

Ricardo Martin-Brualla*, Noha Radwan*, Mehdi S. M. Sajjadi*,
Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth

Google Research

{rmbrualla, noharadwan, msajjadi, barron, adosovitskiy, duckworthd}@google.com

Abstract

We present a learning-based method for synthesizing novel views of complex scenes using only unstructured collections of in-the-wild photographs. We build on Neural Radiance Fields (NeRF), which uses the weights of a multilayer perceptron to model the density and color of a scene as a function of 3D coordinates. While NeRF works well on images of static subjects captured under controlled settings, it is incapable of modeling many ubiquitous, real-world phenomena in uncontrolled images, such as variable illumination or transient occluders. We introduce a series of

(a) Photos (b) Renderings

Figure 1: Given only an internet photo collection (a), our method

With NeRF, we can generate 360 videos from sets of images taken with calibrated cameras

Want to know how to apply NeRF to in-the-wild, unconstrained photos? Please stay tuned!

Thank You!