Representing Scenes as Neural Radiance Fields for View Synthesis

Adarsh S Menon 2017ME10563

COV887

Instructor: Prof. Rahul Narain

Paper Presented

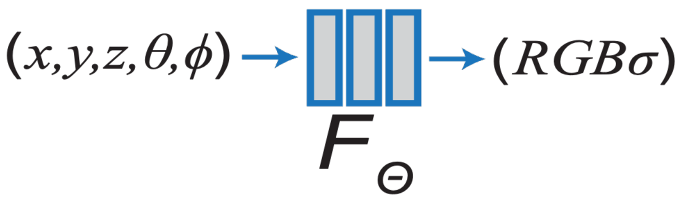
- Mildenhall et al., Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV 2020.

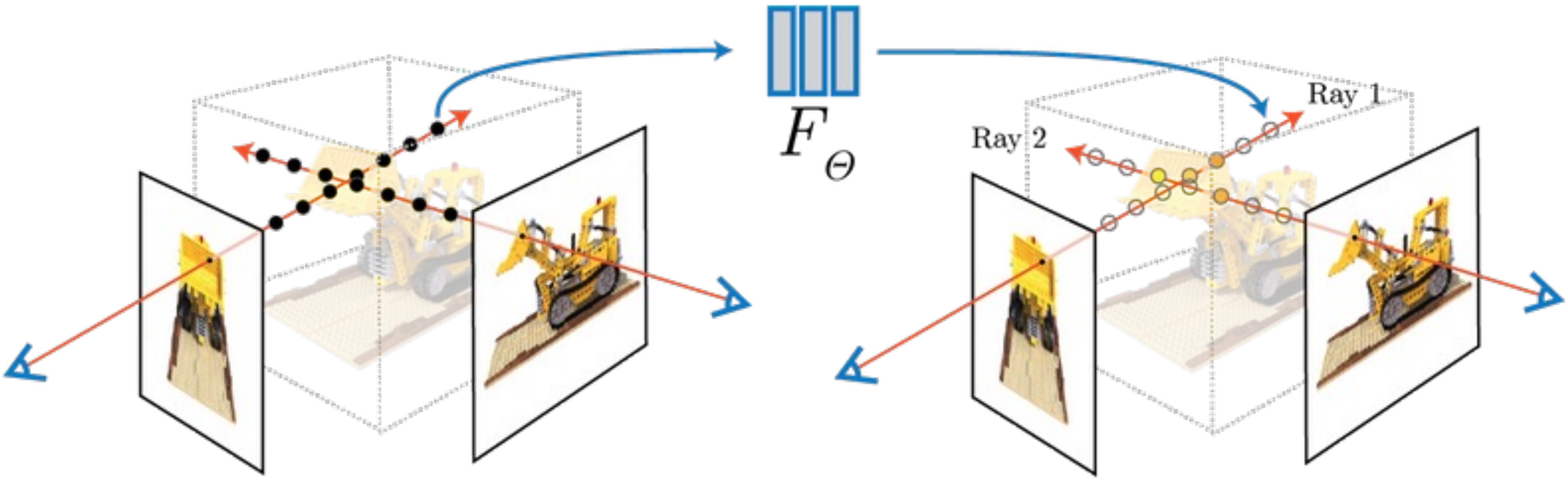
Problem and Solution

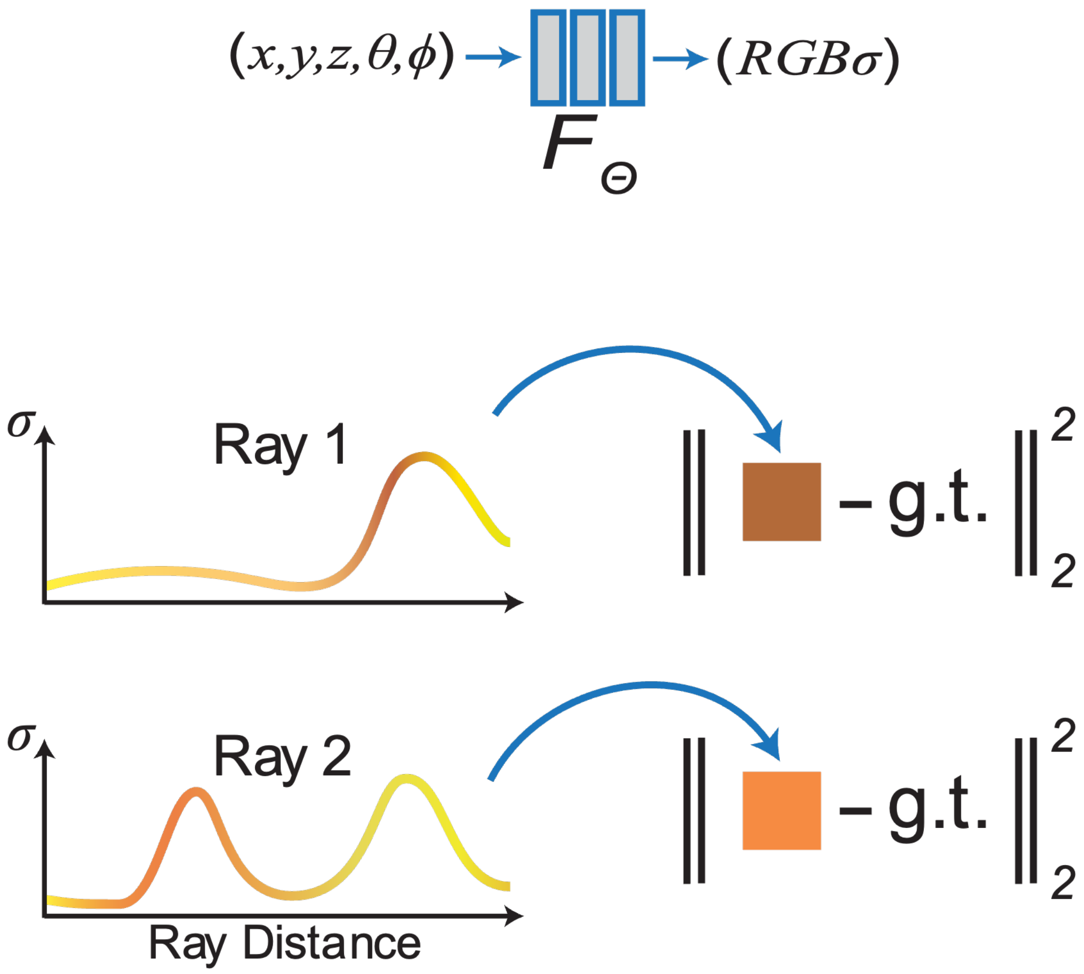
- NeRF present a method that achieves state-of-the-art results for synthesizing novel views of complex scenes by optimizing an underlying continuous volumetric scene function using a sparse set of input views.

Vector equation

$$(x,y,z,\theta,\phi) \rightarrow F_{\Theta} \rightarrow (RGB\sigma)$$


(x,y,z,θ,ϕ)
(RGBσ)
FΘ


FΘ
Ray 1
Ray 2

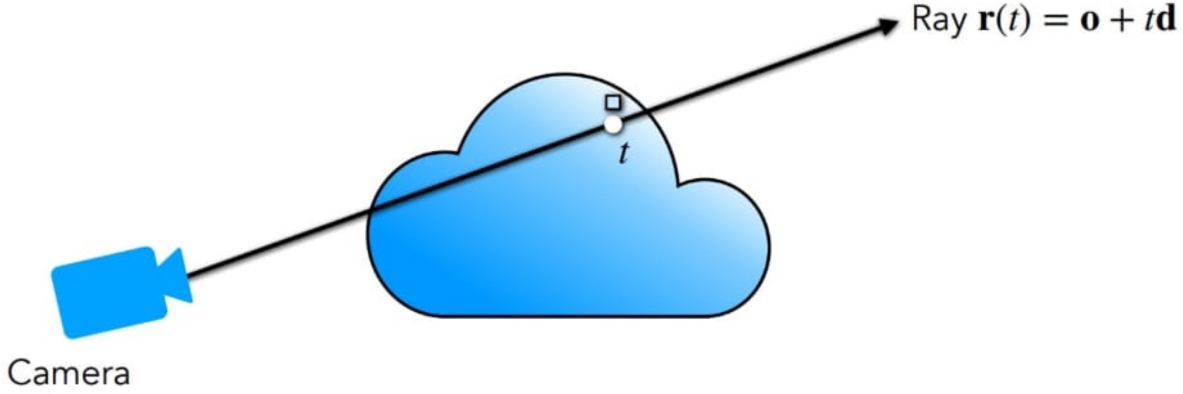


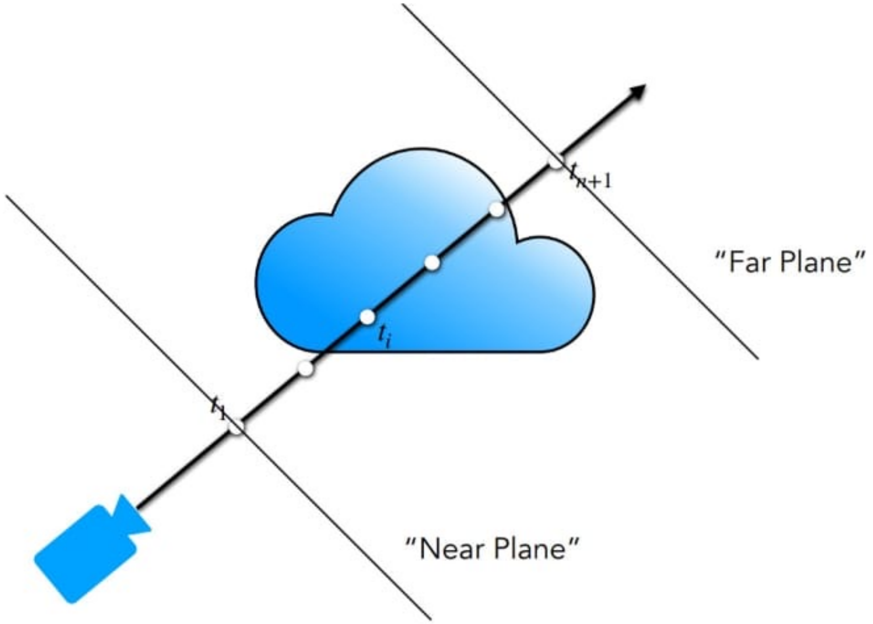
(x,y,z,θ,ϕ)
$(RGB\sigma)$
F_Θ
σ
Ray 1
σ
Ray 2
Ray Distance
$\| \; - \text{g.t.} \|_2^2$
$\| \; - \text{g.t.} \|_2^2$

Volumetric representations

What if we treat the scene as a cloud of tiny coloured particles?

- Density $\sigma = 0$: empty, $\sigma \gg 1$: looks opaque
- With probability $\sigma(\mathbf{x})\,dt$, ray hits at t and sees colour $\mathbf{c}(\mathbf{x})$

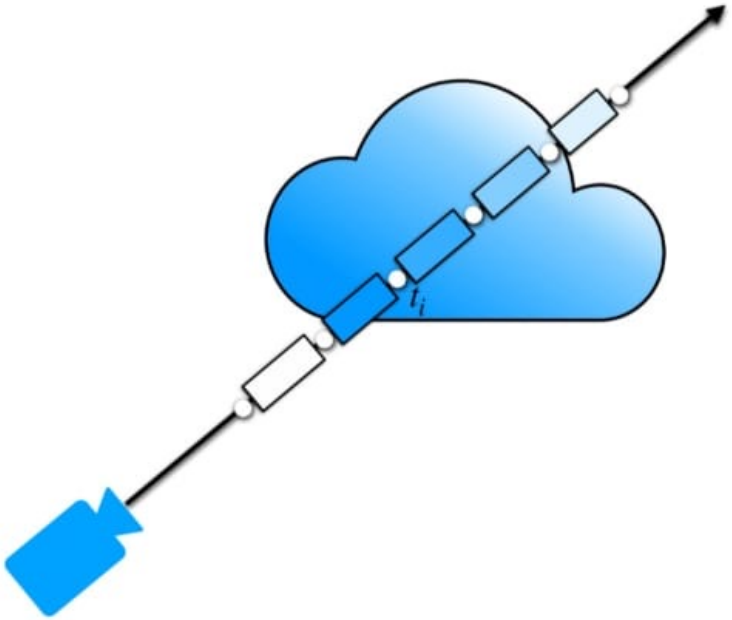




How to render a volume?

Divide the ray into n intervals $[t_i, t_{i+1}]$ of length $\delta_i = t_{i+1} - t_i$

Assume each interval has constant density σ_i and colour $\mathbf{c}_i$



Opacity of each segment:

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

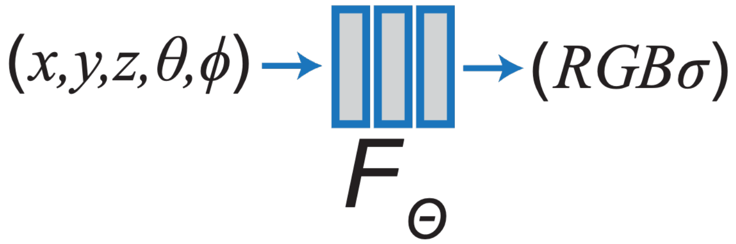
So, probability of seeing *i*th segment:

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

Expected colour seen by ray:

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

Functional Representation in 3D: TSDF and NeRF

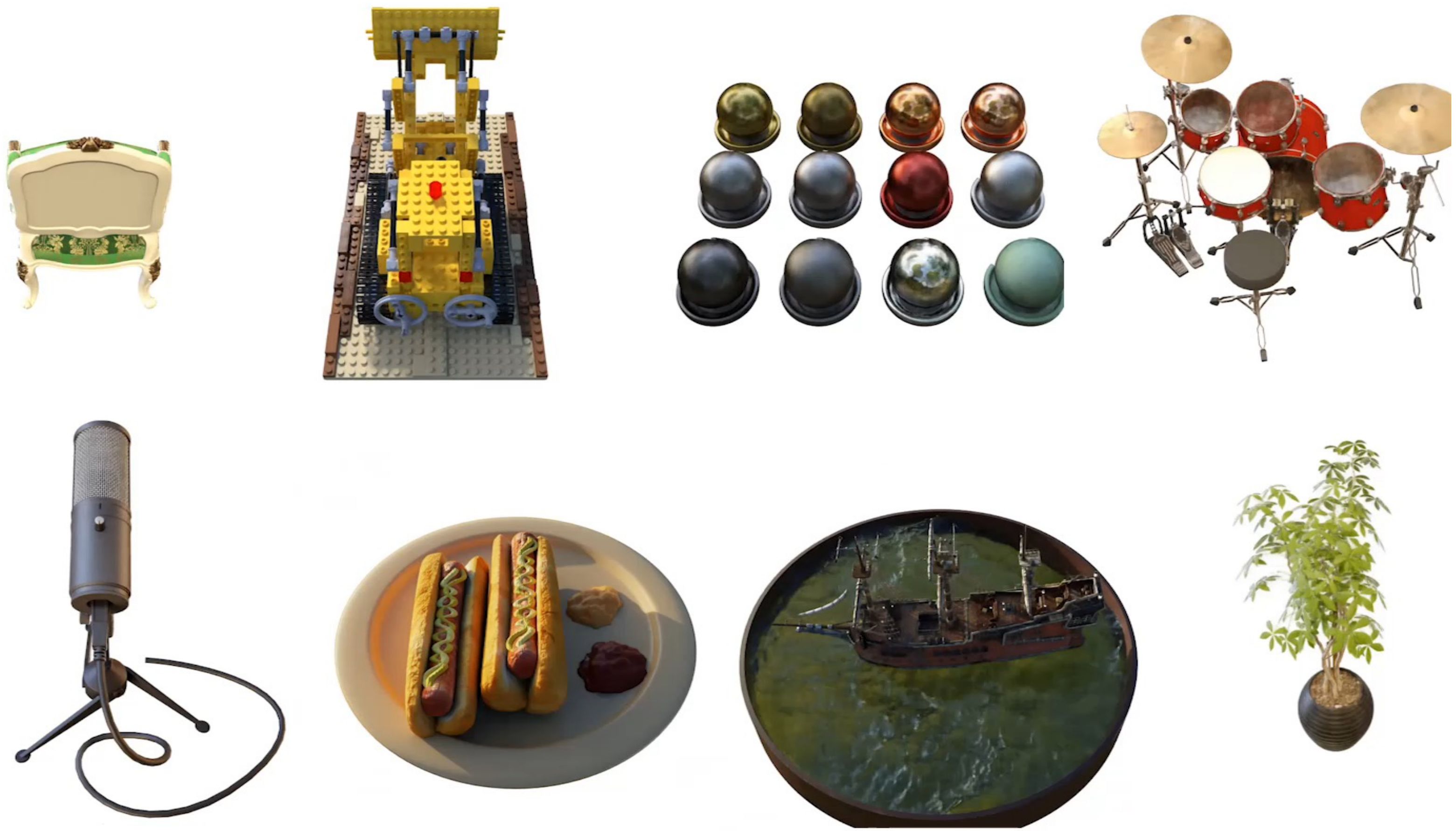


This algorithm represents a scene using a fully-connected (non-convolutional) deep network, whose input is a single continuous 5D coordinate (spatial location (x, y, z) and viewing direction (θ, ϕ)) and whose output is the volume density and view-dependent emitted radiance at that spatial location.

- It is shown in the video that this method(NeRF) is better than existing SRN Method as it offers higher resolution and clarity.

Synthetic Results

Here are results on our synthetic dataset of pathtraced objects with realistic non-Lambertian materials.



View-Dependent Appearance

Here we visualize the view-dependent appearance encoded in our NeRF representation by fixing the camera viewpoint but changing the queried viewing direction.

View-Dependent Appearance

Here we visualize the view-dependent appearance encoded in our NeRF representation by fixing the camera viewpoint but changing the queried viewing direction.

Geometry Visualization

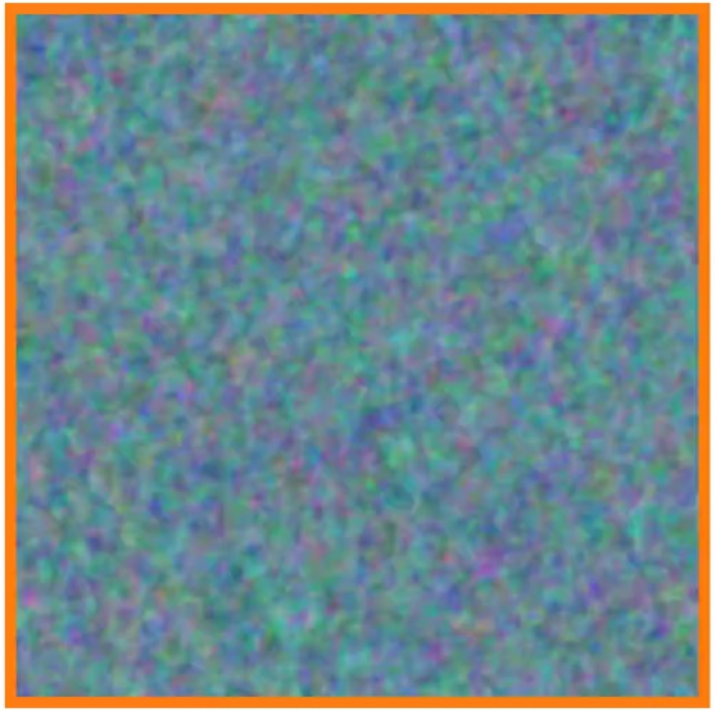
NeRFs are able to represent detailed scene geometry with complex occlusions. Here we visualize depth maps for rendered novel views computed as the expected termination of each camera ray in the encoded volume.

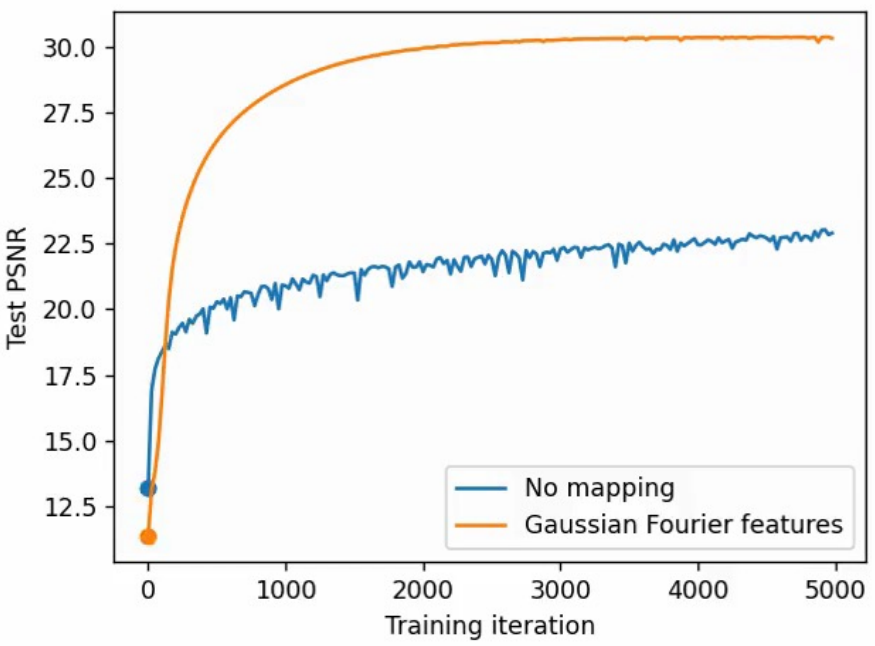
360° Scene Capture with Real Data

NeRFs can even represent real objects captured by a set of inward-facing views, without any background isolation or masking.

Positional Encoding

Fully-connected deep networks are biased to learn low frequencies faster. Surprisingly, applying a simple mapping to the network input is able to mitigate this issue.





Limitations

- The main limitation of NeRFs is that they only model a single scene at a time and are expensive to train (i.e., 2 days on a single GPU for each scene).
- Can only be rendered with an Nvidia GPU with CUDA.
- No standard “NeRF scene” file format.
- Aliasing-like artifacts.
- Strictly a static scene with static lighting.
- Requires accurate camera poses.
- Models are not editable or composable (or animatable, or deformable).

Thank You