Novel View Synthesis using Radiance Field Methods: NeRF and Instant-NGP

Prakash K Naikade (7000433)
MSc - Media Informatics
prna00001@stud.uni-saarland.de

UNIVERSITÄT DES SAARLANDES

AI
DAM
Artificial Intelligence aided Design and Manufacturing

max planck institut informatik

What is Novel View Synthesis?

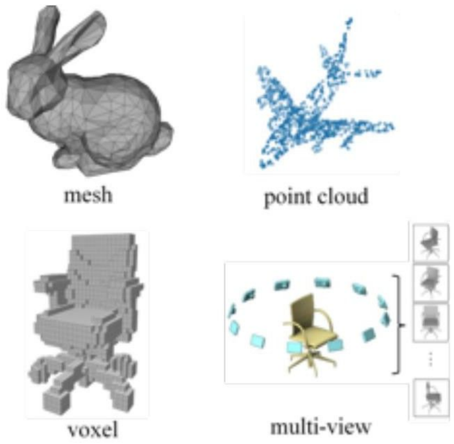
- Synthesizing a target image with an arbitrary target camera pose from given source images and camera poses
- Revolutionized Novel View Synthesis of captured scenes with multiple photos and videos
- Capture complex effects such as view-dependent reflections on an object's surface
- Visualization and Novel View Synthesis of laser printed structural-color image/painting on metal substrate

Novel View Synthesis

- **3D Deep Learning Overview**
  - Traditional Approach vs Neural Fields

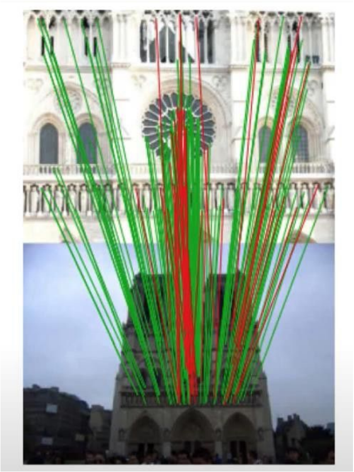
<u>Traditional 3D Representation</u>

- Explicit Representation
- (Given an input image), the network output is also (directly) an image (or a 3D representation)



<u>Neural Fields</u>

- Implicit Representation
- (Given an input coordinate) the network outputs the RGB value of the coordinate, which completes a flawless image when combined afterwards

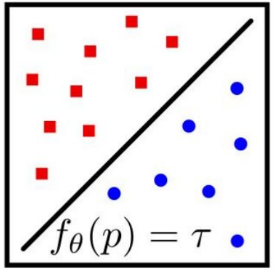


1. Slide credit: Aria Lee

Implicit Representation based Novel View Synthesis Methods

- Modeling Scene Geometry and Appearance (Color) using implicit function
- Model this implicit function using neural network
- E.g. Occupancy Network, Scene Representation Network, NeRF

$x^2 + y^2 + z^2 = 1$

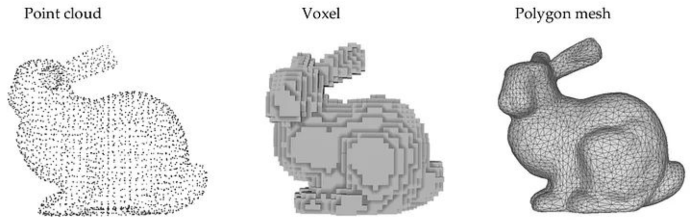


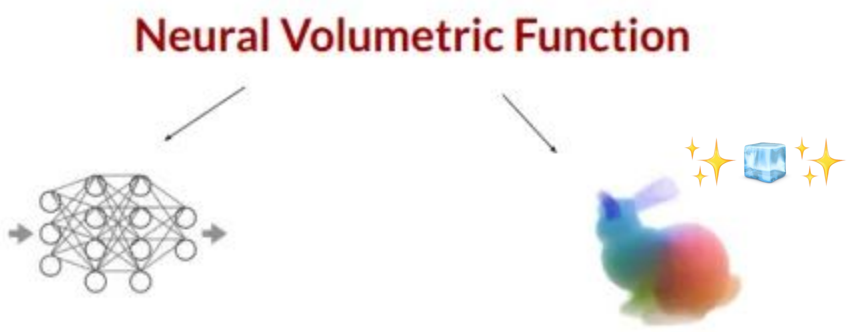
$$(\mathrm{x,y}) \rightarrow f_\theta(x, y) \rightarrow RGB$$

1. Mescheder et al. “Occupancy Networks: Learning 3D Reconstruction in Function Space”, CVPR, 2019
2. Sitzmann et al. “Scene representation networks: Continuous 3d-structure-aware neural scene representations”, NeurIPS, 2019

NeRF - Neural Radiance Fields

- **What kind of a 3D representation is this?**
- Not Mesh, Point Cloud or Voxels
- **Volumetric -** continuous voxels made of shiny transparent cubes





1. Mildenhall et al. "Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020
2. Slide credit: Angjoo Kanazawa

NeRF - Neural Radiance Fields

- **What kind of problem is being solved??**
- **Plenoptic Function**
  - 7D function full plenoptic describes light transport as light waves.
  - Recreate the visual reality.
  - Simplified version of Plenoptic Function.
  - 5D function that describes light transport in a 3D space.

$$P(\theta,\varphi,\cancel{\lambda},\cancel{t},V_X,V_Y,V_Z) \longrightarrow P(\theta,\varphi,V_X,V_Y,V_Z)$$

7D function:

2 – direction
1 – wavelength
1 – time
3 – location

1. https://radiancefields.com/history-of-neural-radiance-fields/

NeRF - Neural Radiance Fields

- **Training data:** A set of *(image, viewpoint)* pairs for a scene
- **Output:** An image for the scene as seen from new viewpoint not in the training data

Input
Set of calibrated Images

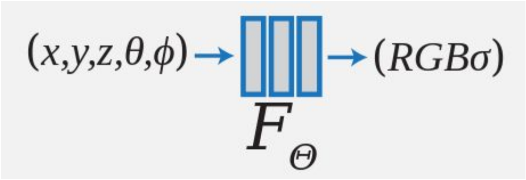
Output
3D scene representation that renders novel views
Aims to predict the multi-view images of the object

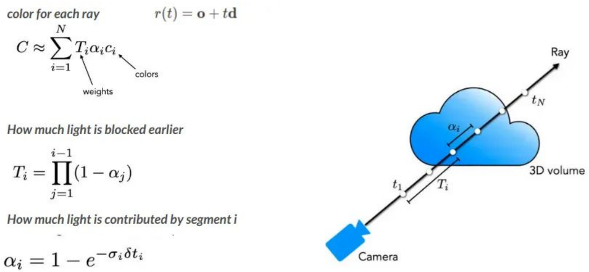
1. Mildenhall et al. "Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020

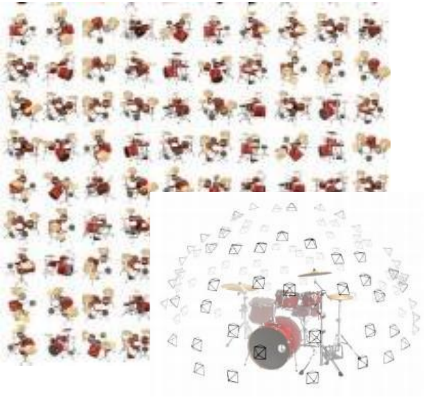
NeRF - High Level Overview

- **Three Key Components**

Objective: Synthesize all training views







Neural Volumetric 3D Scene Representation

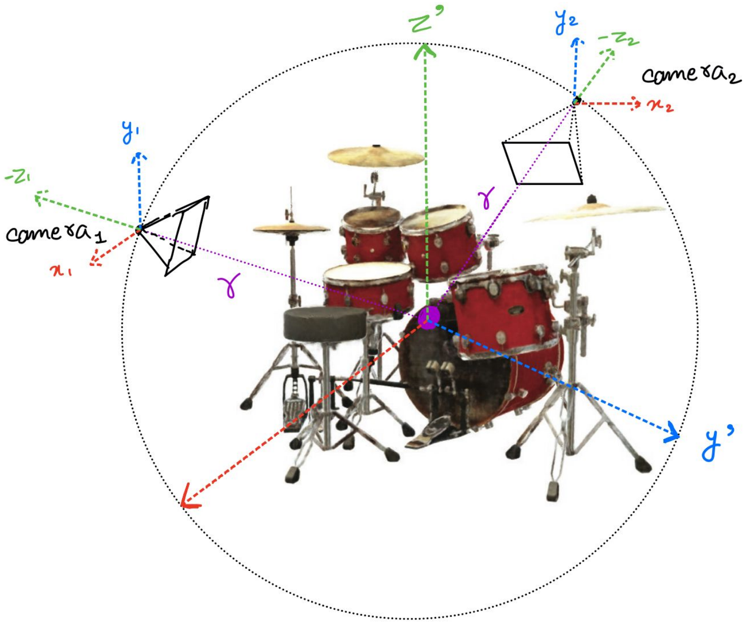
Differentiable Volumetric Rendering Function

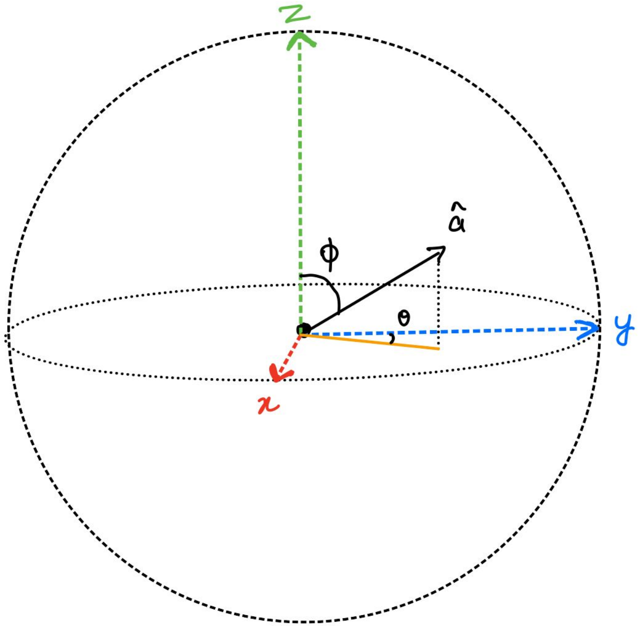
Optimization via Analysis-by-Synthesis

1. Mildenhall et al. "Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020
2. Slide credit: Angjoo Kanazawa

NeRF - Neural Radiance Fields

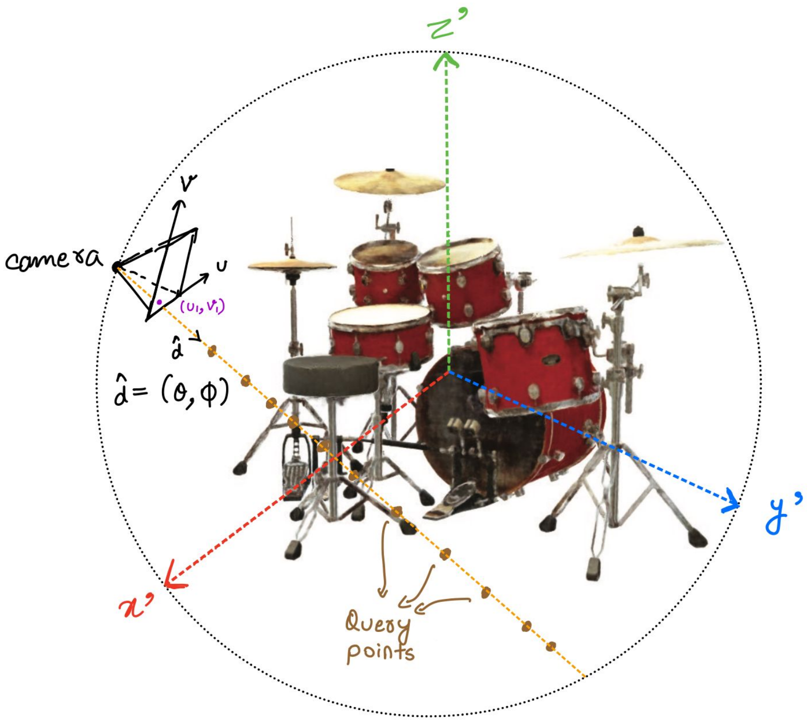
- **Viewpoints/View Directions**
  - Number of (θ, ϕ) pairs equal to number of pixels in the image





1. Mildenhall et al. "Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020
2. https://www.apoorvesinghal.com/blog/nerf/

NeRF - High Level Overview



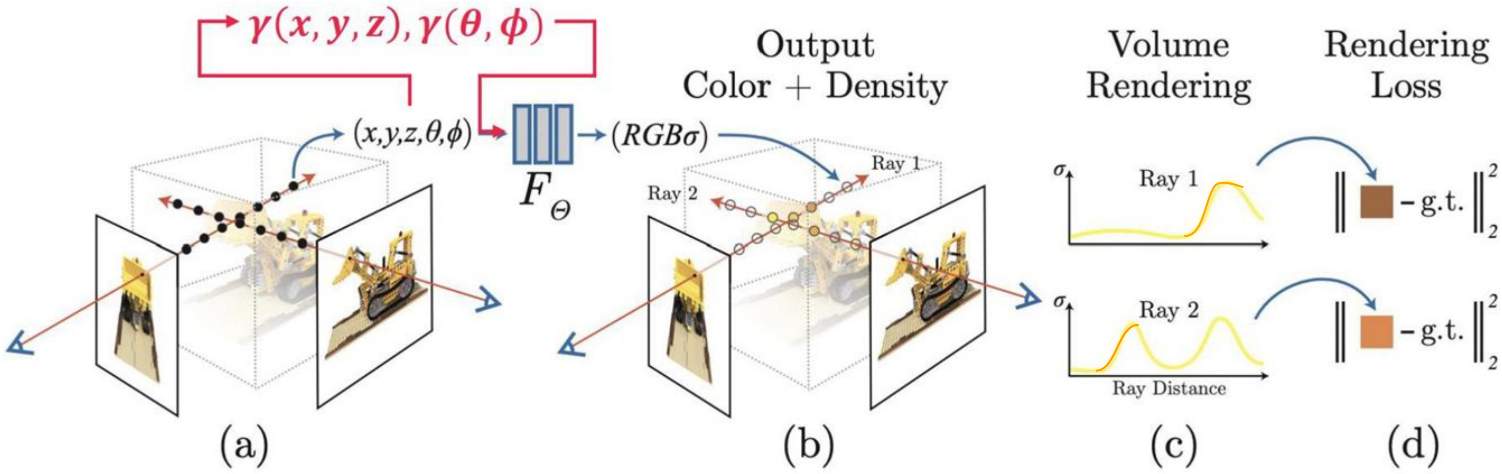
For each pixel (u_1, v_1) of the image:

1. a ray of light in the direction $\hat{\mathrm{d}}$, defined using two angles $\hat{\mathrm{d}} = (\theta, \phi)$
2. ray will hit several points of the scene.
3. final colour at (u_1, v_1) in the image will be a combination of the colour of those points and their respective opacity (how much light they allow to pass through them).
4. compare the calculated value of the colour of the pixel and the actual value from the input image using a loss function.

1. Mildenhall et al. “Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020
2. https://www.apoorvesinghal.com/blog/nerf/

NeRF - High Level Overview

- **Architecture: Representing a 3D scene as a continuous 5D function**



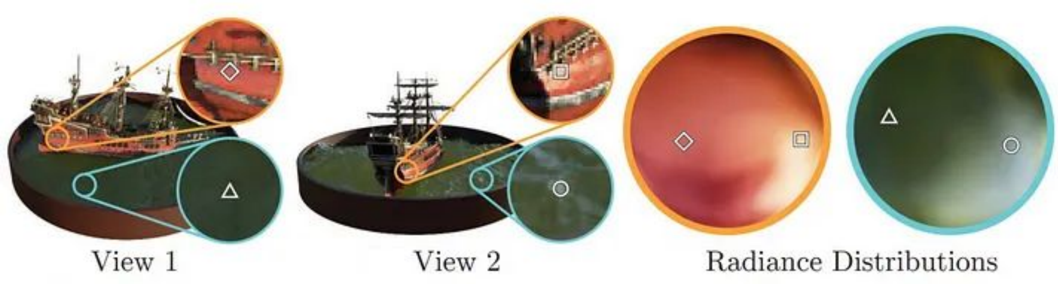
a. Sample 5D coordinates along camera rays
b. Feed those locations into an MLP to produce a color and volume density
c. Use volume rendering techniques to composite these values into an image
d. Optimize our scene representation by minimizing the residual between synthesized and ground truth observed images

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

NeRF - Deep Dive

- **(a) Sample 5D coordinates along camera rays**
  - **Color at a point**
    - Color is function of
      - Location of point $(x,\ y,\ z)$ and
      - View direction $(\theta,\ \phi)$
    - **Hence view dependent color**

$$F_{color} : (x, y, z, \theta, \phi) \rightarrow (R, G, B)$$



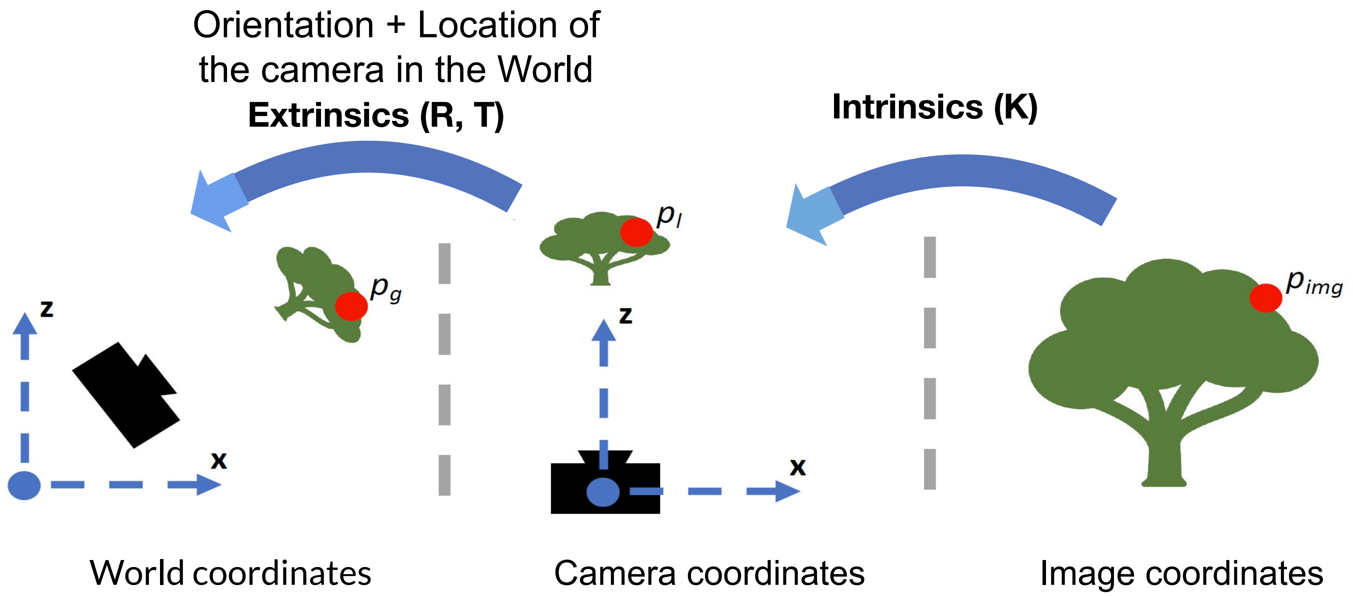
  - **Amount of light that any point reflects, refracts and transmits**
    - inherent property of the point
    - Density σ is only a function of the location of point $(x,\ y,\ z)$

$$F_{density} : (x, y, z) \rightarrow \sigma$$

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

NeRF - Neural Radiance Fields

- **(a) Sample 5D coordinates along camera rays**
  - Mapping from (camera, pixel) to ray

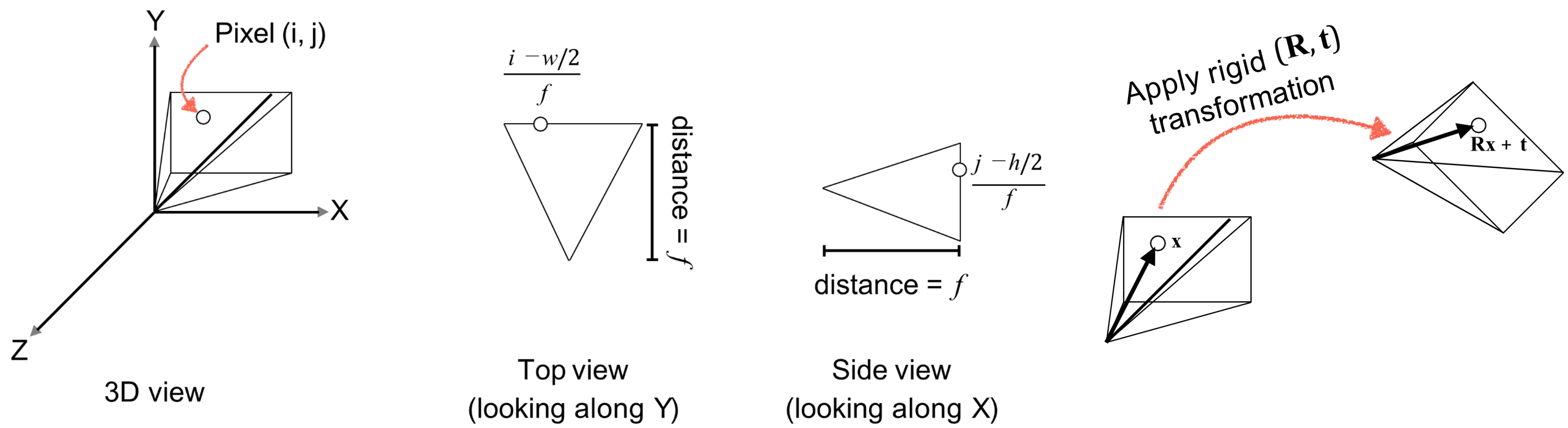


1. Figure credit: Ben Mildenhall

NeRF - Neural Radiance Fields

- **(a) Sample 5D coordinates along camera rays**
  - Mapping from (camera, pixel) ⟶ ray in camera coordinate frame
  - abstract underlying problem as learning the function ray ⟶ color

Pixel to Camera **Camera to World**

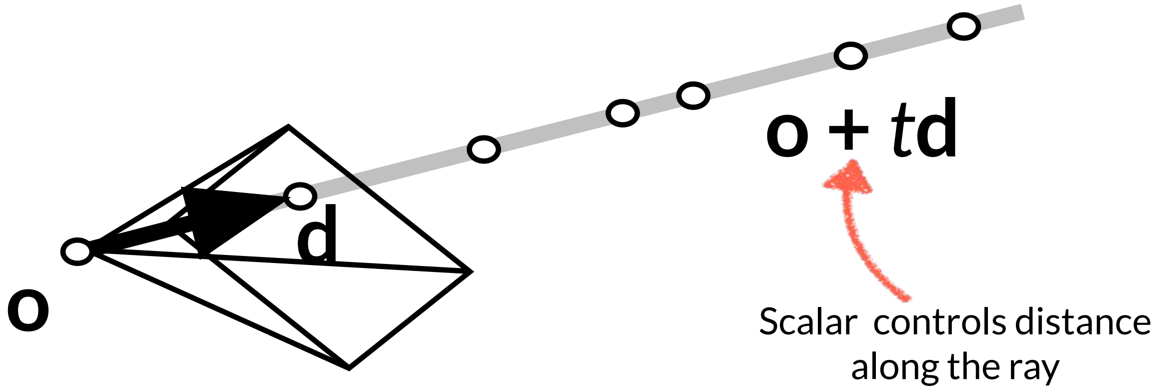


Full mapping is $(i, j) \rightarrow \left(\frac{i - w/2}{f}, \frac{j - h/2}{f}, -1 \right)$

1. Figure credit: Ben Mildenhall

NeRF - Neural Radiance Fields

- (a) Sample 5D coordinates along camera rays
  - Calculating points along a ray

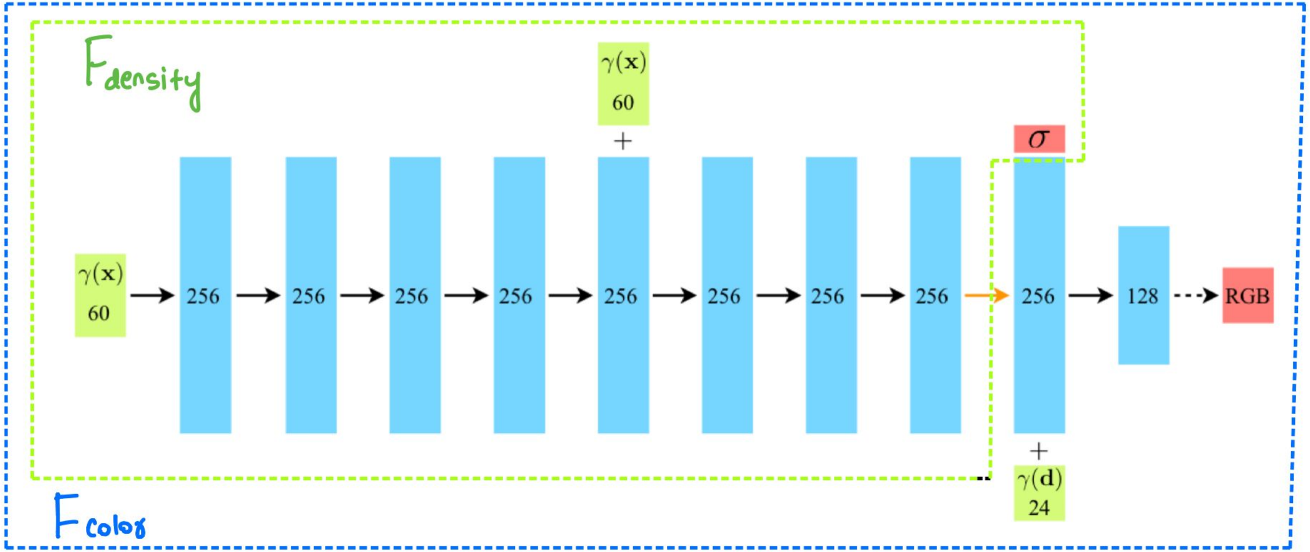


$$t_i \sim U\left[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)\right]$$

1. Figure credit: Ben Mildenhall

NeRF - Deep Dive

- (b) Feed those locations into an MLP to produce a color and volume density
    - MLP to approximate the functions F_{color} and $F_{density}$

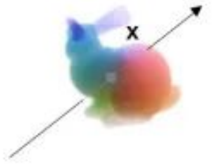


$$d = (d_x, d_y, d_z) = \text{viewpoint}(\theta, \phi)$$
$$x = (x, y, z)$$

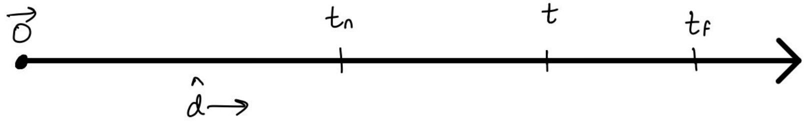
1. Mildenhall et al. “Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020
2. https://www.apoorvesinghal.com/blog/nerf/

NeRF - Deep Dive

- **(c) Use volume rendering techniques to composite these values into an image**
  - **Blending the colours along a ray using Differentiable Volumetric Rendering Function:**



$$\gamma(t) = \vec{o} + t\hat{d}$$



  - **Differentiable Volumetric Rendering Equation:**

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt\,, \text{ where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$$

Transmittance Volume Density Radiance/Color

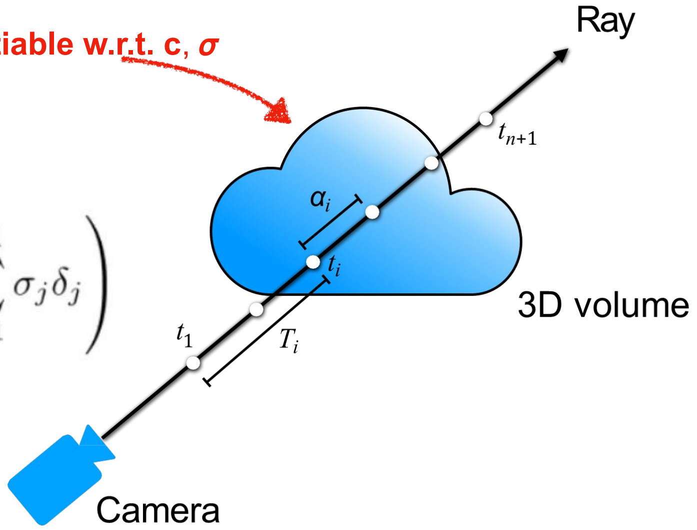
1. Mildenhall et al. "Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020

- **(c) Use volume rendering techniques to composite these values into an image**
  - **Differentiable Volumetric Rendering Equation:**

$$C(\mathbf{r}) = \int_{t_n}^{t_f} \underbrace{T(t)}_{\text{Transmittance}}\underbrace{\sigma(\mathbf{r}(t))}_{\text{Volume Density}}\underbrace{\mathbf{c}(\mathbf{r}(t), \mathbf{d})}_{\text{Radiance/Color}}dt\,, \quad \text{where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$$

Transmittance Volume Density Radiance/Color

differentiable w.r.t. c, σ



  - **Integral as a sum:**

[Approx. for sampled points along the ray]

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i \underbrace{(1 - \exp(-\sigma_i \delta_i))}_{\text{Opacity}}\mathbf{c}_i\,, \quad \text{where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$$

Opacity

Contribution Weight

$$C(r) \approx \sum_{i=1}^{N} T_i \alpha_i c_i$$

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

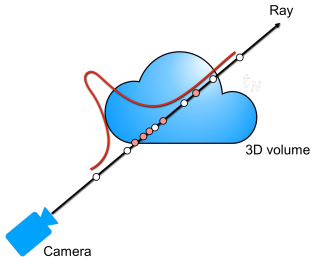
NeRF - Deep Dive

- **(d) Optimize our scene representation by minimizing the residual between synthesized and ground truth observed images**
  - **Stratified sampling:**

$$t_i \sim U\left[t_n + \frac{i-1}{N}(t_f - t_n),\, t_n + \frac{i}{N}(t_f - t_n)\right]$$

  - **Hierarchical sampling:**

$$\hat{C}_c(\mathbf{r}) = \sum_{i=1}^{N_c} w_i c_i \,, \qquad w_i = T_i(1 - \exp(-\sigma_i \delta_i))$$



    - first coarse pass, we will sample a small number of points N_c
    - Normalize weights $\hat{w}_i = w_i / \sum_{j=0}^{N_c} w_j$, produce a piecewise-constant PDF along the ray
    - second fine pass, we will use the regions where the *good* points lie to sample more points N_f
    - compute the final rendered colour of the ray using all samples $N_c + N_f$

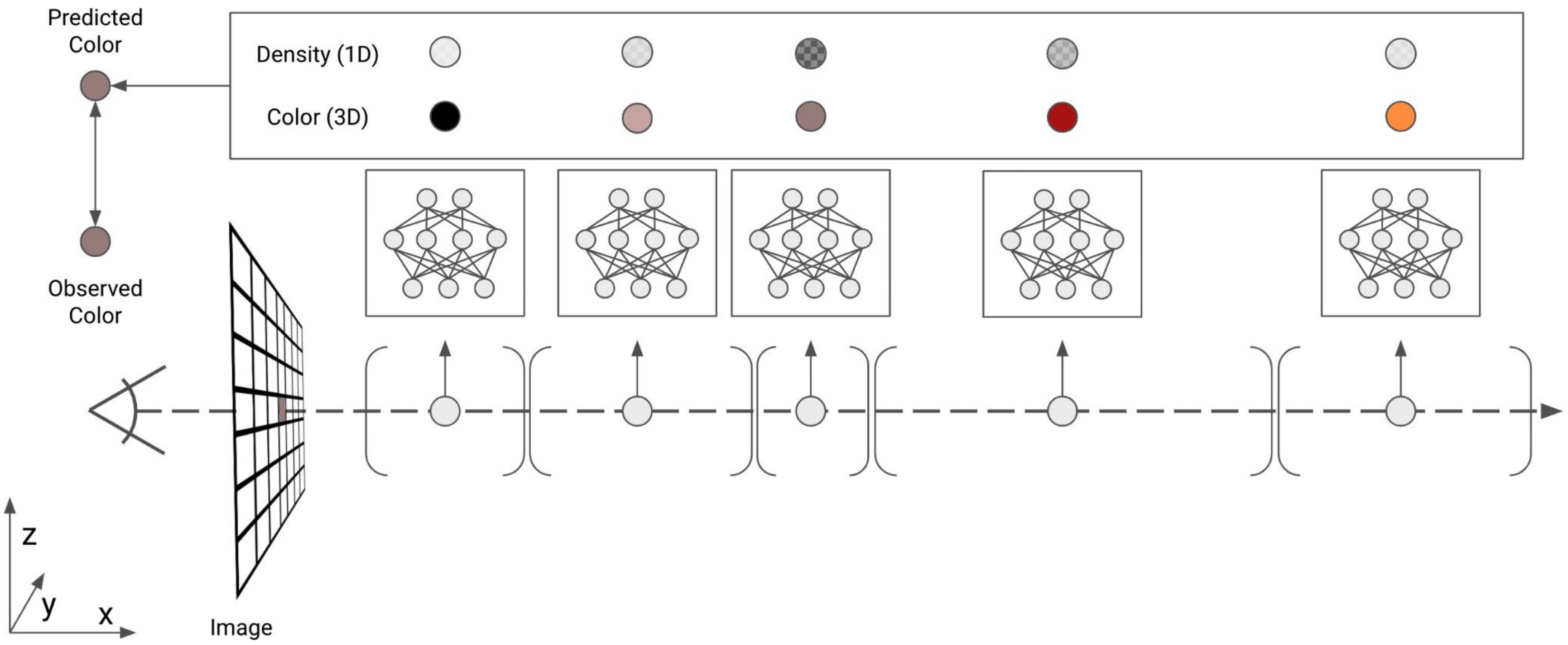
  - **Loss Function:**

$$\mathcal{L} = \sum_{\mathbf{r} \in \mathcal{R}} \left[\left\| \hat{C}_c(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right]$$

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

NeRF - Deep Dive

- **(d) Optimize our scene representation by minimizing the residual between synthesized and ground truth observed images**



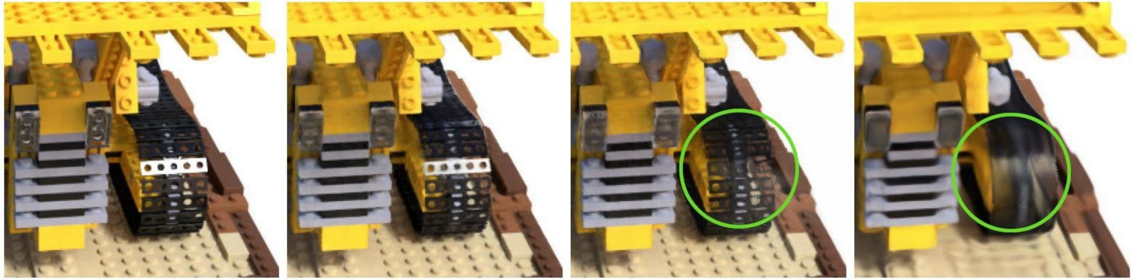
1. Slide credit - Aria Lee

NeRF - Deep Dive

- **Position encoding plays vital role in high-fidelity visual representation:**

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

  - For any scalar p the function $\gamma(p)$ converts it into a $2L$-dimensional representation.
- Having network directly process input coordinate **fails to represent high-frequency details** in color and geometry
- Positional encoding **maps low-dimensional coordinates to high-dimensional space using high frequency functions**



Ground Truth | Complete Model | No View Dependence | No Positional Encoding

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

NeRF - Deep Dive

- **Performance**
  - PSNR (Peak Signal to Noise Ratio)
  - SSIM (Structural Similarity Index Measure)
  - LPIPS (Learned Perceptual Image Patch Similarity)

	Input	#Im.	L	(N_c, N_f)	PSNR↑	SSIM↑	LPIPS↓
1) No PE, VD, H	xyz	100	-	(256, -)	26.67	0.906	0.136
2) No Pos. Encoding	$xyz\theta\phi$	100	-	(64, 128)	28.77	0.924	0.108
3) No View Dependence	xyz	100	10	(64, 128)	27.66	0.925	0.117
4) No Hierarchical	$xyz\theta\phi$	100	10	(256, -)	30.06	0.938	0.109
5) Far Fewer Images	$xyz\theta\phi$	25	10	(64, 128)	27.78	0.925	0.107
6) Fewer Images	$xyz\theta\phi$	50	10	(64, 128)	29.79	0.940	0.096
7) Fewer Frequencies	$xyz\theta\phi$	100	5	(64, 128)	30.59	0.944	0.088
8) More Frequencies	$xyz\theta\phi$	100	15	(64, 128)	30.81	0.946	0.096
9) Complete Model	$xyz\theta\phi$	100	10	(64, 128)	**31.01**	**0.947**	**0.081**

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

NeRF - Deep Dive

- **Limitations**
    - **8-layer MLP for each sample points**

        Number of sample points for rendering [480, 640] resolution image = 480 x 640 x (64 + 192) ≅ 79M
    - **Redundant computation**

        No caching of the intermediate result of the sample point

        Backprop gradients to every layer of MLP for every sample point
    - **Requires up to one day to train a single scene**

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

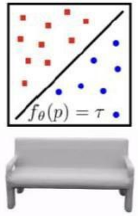
Explicit Representation based Novel View Synthesis Methods

- Blending classical data structures and neural approaches
- Store additional trainable parameters in an auxiliary data structures (such as grids and trees)
- Requires larger memory footprint but smaller computational cost because
  - Number of parameters are high
  - Number of FLOPs and memory access required for the update during training is not high

Explicit Representation based Novel View Synthesis Methods

- **Diachronic Analysis of the Improvements in Encoding Methods**

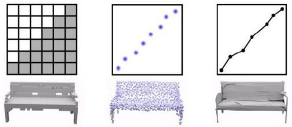
Implicit Representation



$$(x, y) \rightarrow f_\theta(x, y) \rightarrow RGB$$

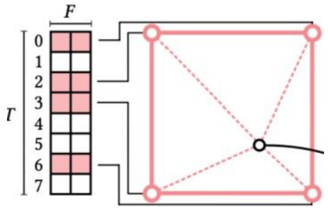
- **(+) Smooth and expressive, low memory footprint**
- (-) NeRF requires **rendering of too many sample points**
- (-) NeRF requires redundant computation, **thus resulting in up to one day of training**

Explicit Representation



- **Uses explicit data structure (= Feature grid-based method)**
- Dense or sparse voxel grid, octree, Plenoxels, etc
- Cache intermediate features of MLP (aiming that **retrieving features from explicit data structure is cheap** - O(1) for voxel grid)
- Enable fast training or rendering

Instant NGP



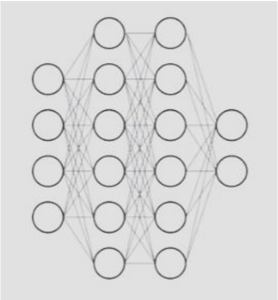
- **Overcomes drawbacks of feature grid-based methods**
  - **Require increased memory footprint** and high resolution to achieve good quality
  - **Involve complicated training procedures** (ex. structural modifications like pruning), limiting its performance on GPU where control flow chasing is expensive

1. Muller et al. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding", SIGGRAPH 2022
2. Slide credit - Aria Lee

Instant NGP

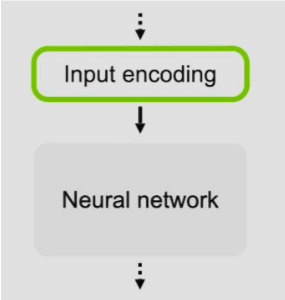
- **Pillars of Instant Neural Graphics**

Small Neural Network



- Fully fused implementation
- Task-specific GPU implementation
- 5-10x fewer steps than TensorFlow

Good Input Encoding
Hybrid Data Structure



- Multiresolution hash encoding
- High approximation power
- Better speed-vs-quality tradeoff

1. Slide credit - Thomas Muller

Instant NGP

- **Architecture - Neural Graphics Primitives**

1) Define L-numbered D-dimension grids (each grid a single level with N resolution.

2) Store each levels to T- sized feature vectors of F-dimensions (TxF table)

★ quality-performance tradeoff (as T grows, memory grows linearly but quality grows sub- linearly)

3) Map input coordinate x to voxels

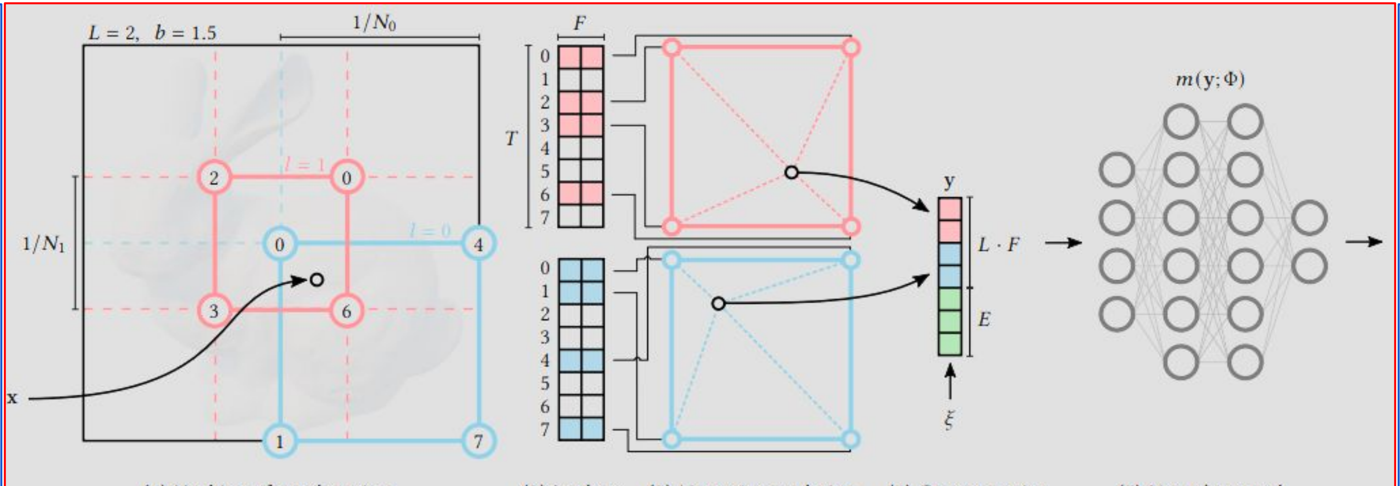


Illustration of the multiresolution hash encoding in **2D**. **(1)** for a given input coordinate x, we find the surrounding voxels at L resolution levels and assign indices to their corners by hashing their integer coordinates. **(2)** for all resulting corner indices, we look up the corresponding F-dimensional feature vectors from the hash tables θ_l and **(3)** linearly interpolate them according to the relative position of x within the respective l-th voxel. **(4)** we concatenate the result of each level, as well as auxiliary inputs $\xi \in \mathbb{R}^E$, producing the encoded MLP input $y \in \mathbb{R}^{LF+E}$, which **(5)** is evaluated last. To train the encoding, loss gradients are backpropagated through the MLP **(5)**, the concatenation **(4)**, the linear interpolation **(3)**, and then accumulated in the looked-up feature vectors.

4) Map voxel vertices to feature vectors (1:1 mapping if V≤T, else use spatial hash function

5) Conduct D-dimension linear interpolation (in order to guarantee continuity)

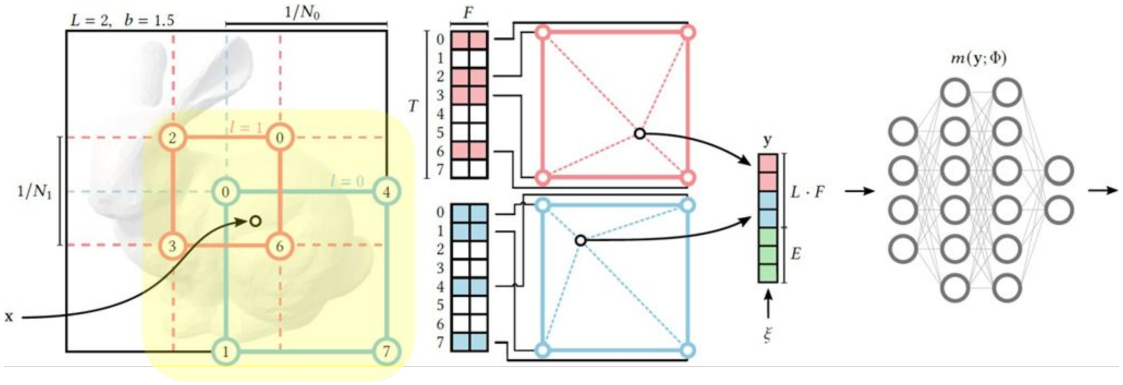
6) Concatenate each level vectors and auxiliary Input (with encoded view directions)

7) Feed to MLP

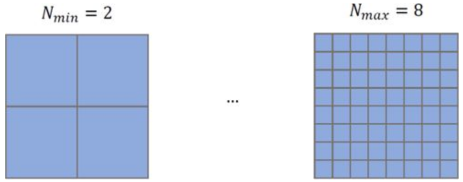
★ Resolutions of each levels are calculated by geometric progression (given N_min and N_max)

1. Muller et al. “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding”, SIGGRAPH 2022

- **Architecture - Neural Graphics Primitives (1)**



- For a given input coordinate x…
  - We find the surrounding voxels at L resolution levels and assign indices to their corners by hashing their integer coordinates
  - L is decided depending on the given task type (Gigapixel: 2 & NeRF: 3)
  - Calculating Resolution

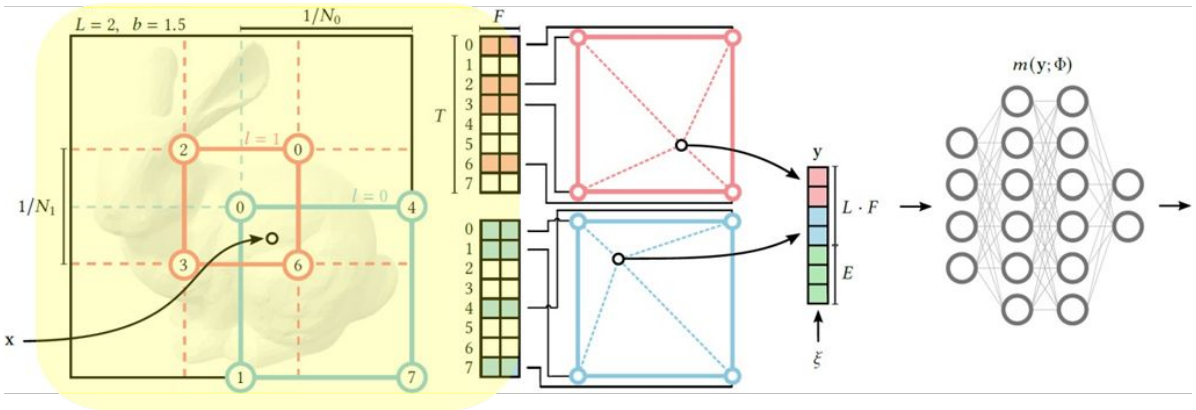


- $N_l \coloneqq \lfloor Nmin \cdot b^l \rfloor$
- $b \coloneqq \exp(\ln N_{max} - N_{min})$

1. Muller et al. “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding”, SIGGRAPH 2022
2. Slide credit - Aria Lee

Instant NGP

- **Architecture - Neural Graphics Primitives (2)**



- **For all resulting corner indices, we look up the corresponding F-dimensional feature vectors from the hash tables θ_l**
  - The input coordinate x is scaled by that level's grid resolution before rounding down and up

    $\lfloor x_l \rfloor := \lfloor x \cdot N_l \rfloor$ and $\lceil x_l \rceil := \lceil x \cdot N_l \rceil$
  - **Each corner is mapped to an entry in the level's respective feature vector array with a fixed size of at most T**
    - For coarse levels where a dense grid requires fewer than T parameters (V < T), this mapping is 1 : 1
    - For finer levels (V > T), we use a hash function to index into the array
    - There is NO explicit collision handling
- **Linearly interpolate them according to the relative position of x within the respective l-th voxel else blocky appearance**

1. Muller et al. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding", SIGGRAPH 2022
2. Slide credit - Aria Lee

Instant NGP

- **Architecture - Neural Graphics Primitives (3) - Hash Function and Collision Handling**

$$h(\mathbf{x}) = \left(\bigoplus_{i=1}^{d} x_i \pi_i \right) \mod T$$

- d for dimension,
- π for unique prime numbers ($\pi_1 := 1, \pi_2 := 2{,}654{,}435{,}761$),
- T for the size of the hash table,
- and $\oplus$ for the bit-wise XOR operation

- **Implicit (Automatic) Collision Handling Process**
  - When samples collide in this way, their gradients average
  - The gradients of the more important samples dominate the collision average, reflecting the needs of the higher-weighted point

1. Muller et al. “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding”, SIGGRAPH 2022

Instant NGP

- **Performance**

	Mic	Ficus	Chair	Hotdog	Materials	Drums	Ship	Lego	avg.
Ours: Hash (1 s)	26.09	21.30	21.55	21.63	22.07	17.76	20.38	18.83	21.202
Ours: Hash (5 s)	32.60	30.35	30.77	33.42	26.60	23.84	26.38	30.13	29.261
Ours: Hash (15 s)	34.76	32.26	32.95	35.56	28.25	25.23	28.56	33.68	31.407
Ours: Hash (1 min)	35.92	33.05	34.34	36.78	29.33	25.82	30.20	35.63	32.635
Ours: Hash (5 min)	36.22	33.51	35.00	37.40	29.78	26.02	31.10	36.39	33.176
mip-NeRF (~hours)	36.51	33.29	35.14	37.48	30.71	25.48	30.41	35.70	33.090
NSVF (~hours)	34.27	31.23	33.19	37.14	32.68	25.18	27.93	32.29	31.739
NeRF (~hours)	32.91	30.13	33.00	36.18	29.62	25.01	28.65	32.54	31.005
Ours: Frequency (5 min)	31.89	28.74	31.02	34.86	28.93	24.18	28.06	32.77	30.056
Ours: Frequency (1 min)	26.62	24.72	28.51	32.61	26.36	21.33	24.32	28.88	26.669

1. Muller et al. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding", SIGGRAPH 2022

Instant NGP

- Positives
  - Extremely fast convergence
  - Fast inference
    - All operations are fully parallelized
    - No control flow is involved throughout the process
    - Uses shallow MLP
    - Moderate memory footprint
  - No task-specific data structure
    - Same structure applies to four different tasks No complicated training procedure
  - No complicated training procedure

- Limitations
  - Still limited to static scenes
  - Specular surfaces

1. Slide credit - Aria Lee

Thank You !

Questions Please !