Neural Graphics: An Architceture's Perspective

Muhammad Husnain Mubarik, Prof. Rakesh Kumar

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Muhammad Husnain Mubarik

- PhD ECE - UIUC – 5th year
  - Computer Architecture, Hardware Accelerators
  - Hardware for graphics, real-time / energy efficient rendering (HPC and energy efficiency)
  - Hardware for ML/DL
  - Advised by: Rakesh Kumar
- Research Experience
  - Hardware Acceleration of Neural Graphics **(ISCA 2023)**
    - Domain specific hardware design for Neural Radiance Fields
    - Cloud System Research Lab (CSR), Intel Labs, Dec 2021 - May 2021.
    - Graphics Research Organization (GRO), Intel, June 2022 - Present.
  - RASR/LOU-E (Ongoing)
    - Hardware software co-design for Deep Learning based Super Resolution
    - Heterogeneous Platforms Lab (HPL), Intel Labs, May 2021 - Aug 2021
  - DASICS/MASICS (Ongoing)
    - Model/Data-specific Design of Deeply-Embedded Tiny Neural Network Accelerators
  - Encryption in Flexible Electronics **(DATE 2023)**
  - Rethinking Programmable Earable Processors **(ISCA 2022)**
    - Earable Computing – “Powerful” Earbuds!! applications / architecture
  - Architectural Support for Supply Chain Resilience (Ongoing)
  - Enabling Strong Encryption On Flexible Devices (Ongoing)
  - Printed Machine Learning Classifiers **(MICRO 2020)** IEEE Micro Top Picks - Honorable Mention 2021
  - Printed Microprocessors **(ISCA 2020)**

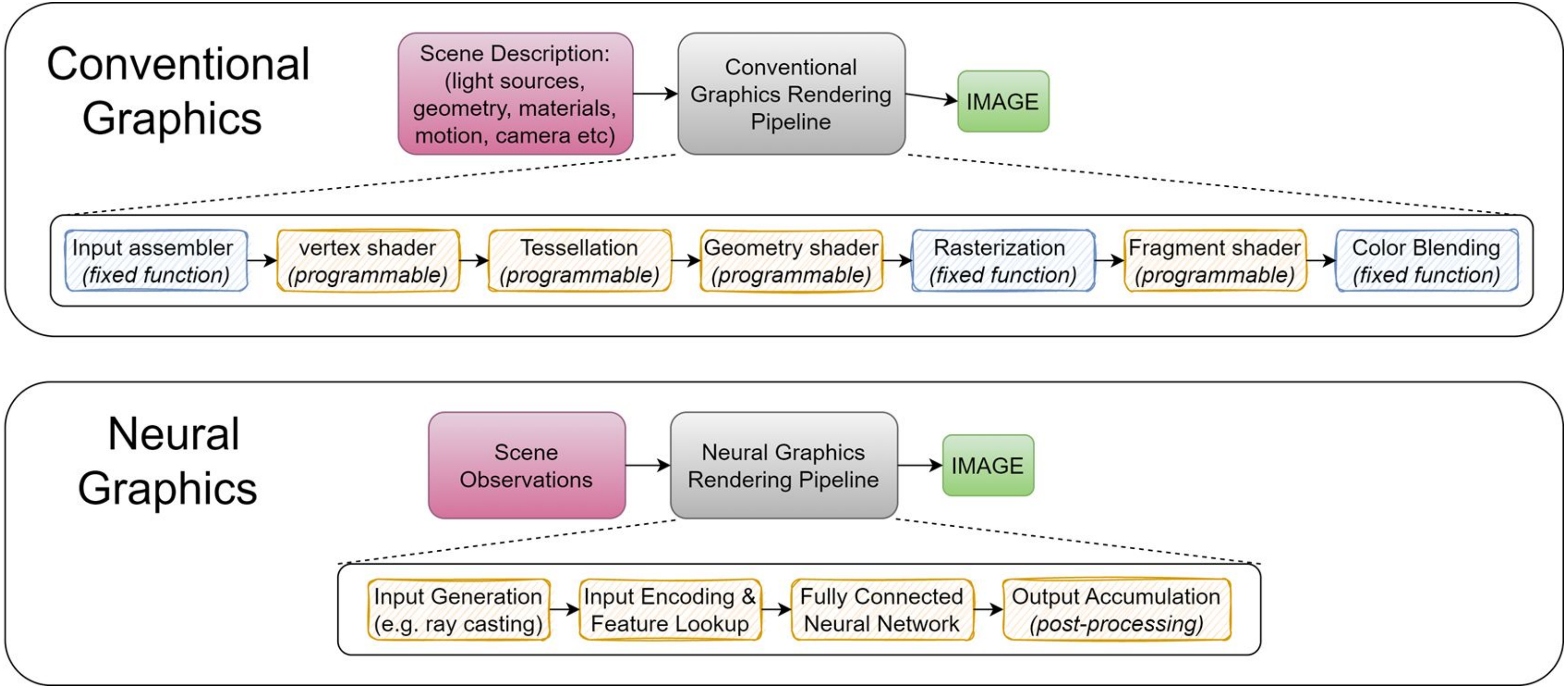
Contents

- About Me
- Conventional Computer Graphics VS Neural Graphics (NG)
- An overview of NG
- State of the art in NG: HW/SW optimizations
- Motivation to accelerate NG in hardware
- NGPC: An accelerator for NG
- Conclusion
- Discussion / Questions

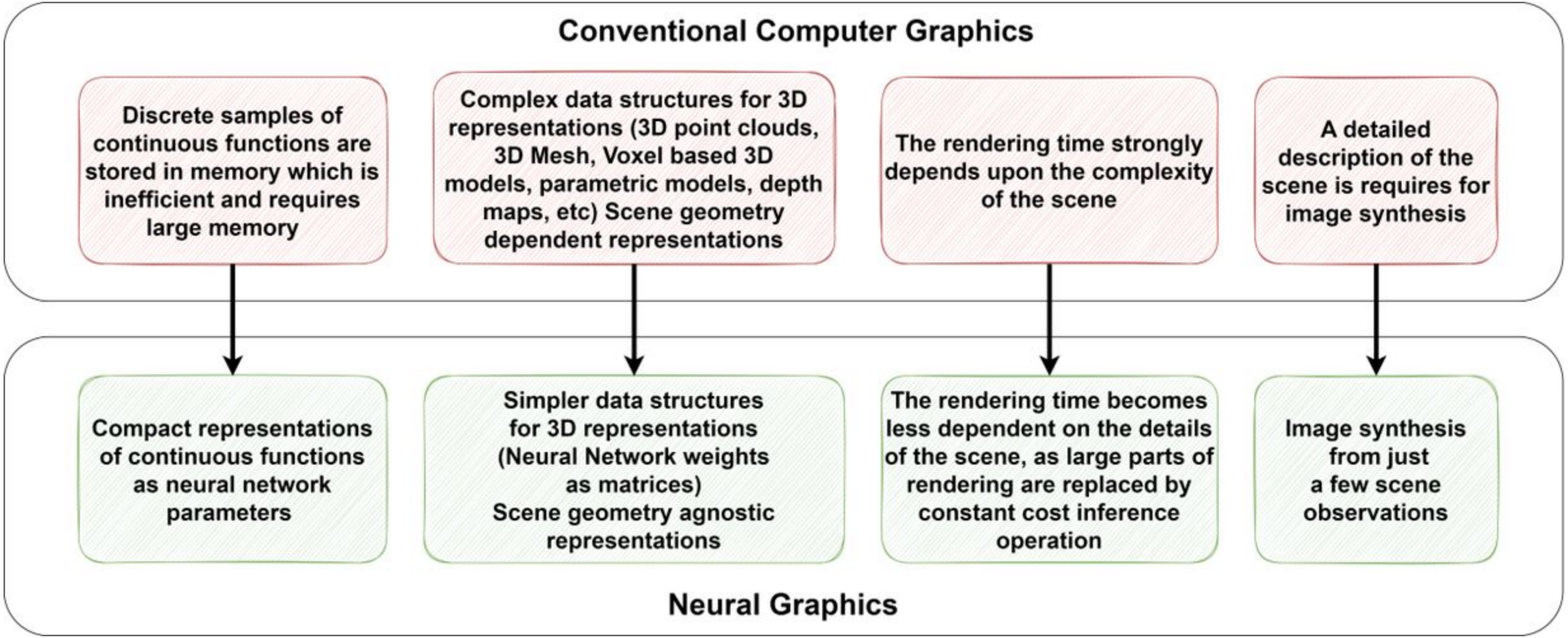
Conventional Computer Graphics VS Neural Graphics 1/3

- Goal: Synthesize photo-realistic and controllable imagery.
- Challenges: Rendering and inverse rendering algorithms are computationally demanding.
- Can neural networks be used to approximate algorithms used in classical computer graphics?
- Neural graphics: Approximating entire or parts of computer graphics using neural networks.
- Benefits: Compact representation, Simpler data structures, Deterministic rendering time, observations to image synthesis.

Conventional Computer Graphics VS Neural Graphics 2/3



Conventional Computer Graphics VS Neural Graphics 3/3



Representing Scenes as Neural Radiance Fields

➔ Neural networks learn scene representations

➔ Query the network to get color and densities

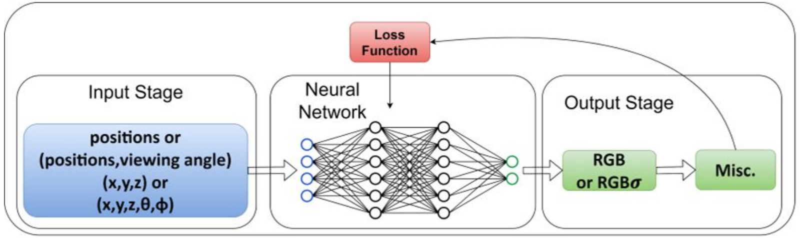
➔ Accumulate color and densities using volumetric rendering

➔ (position, view direction) - (color, volume density)

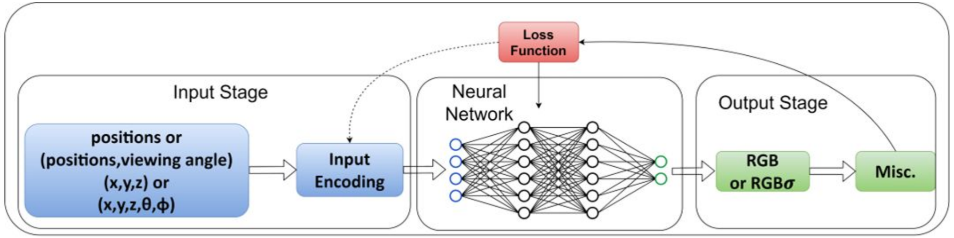
Gist of Neural Graphics

(x,y,z,θ,ϕ) → MLP with scene representation learned in weights → RGB$\boldsymbol{\sigma}$

Structure of a Typical NG Application



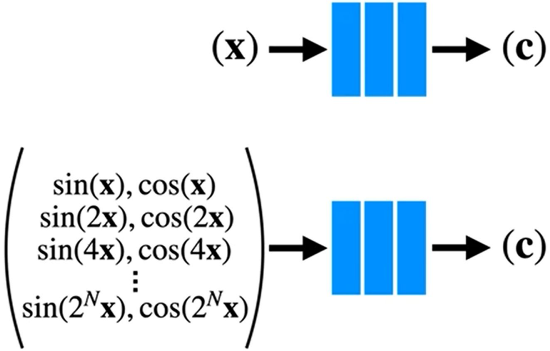
a) Structure of a typical neural graphics application



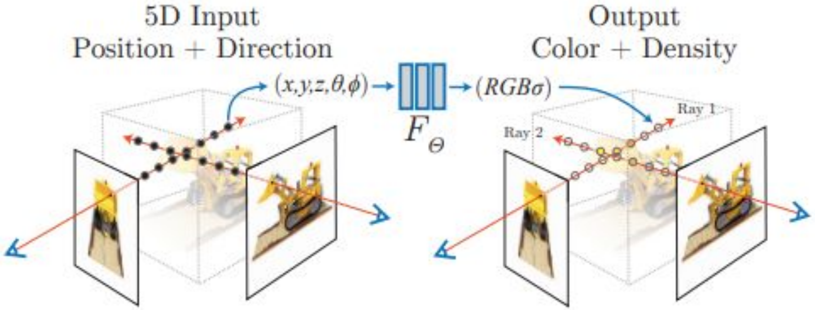
b) Neural graphics application with input encoding - Loss function may or may not update encoding parameters

How does NG Work (images)?

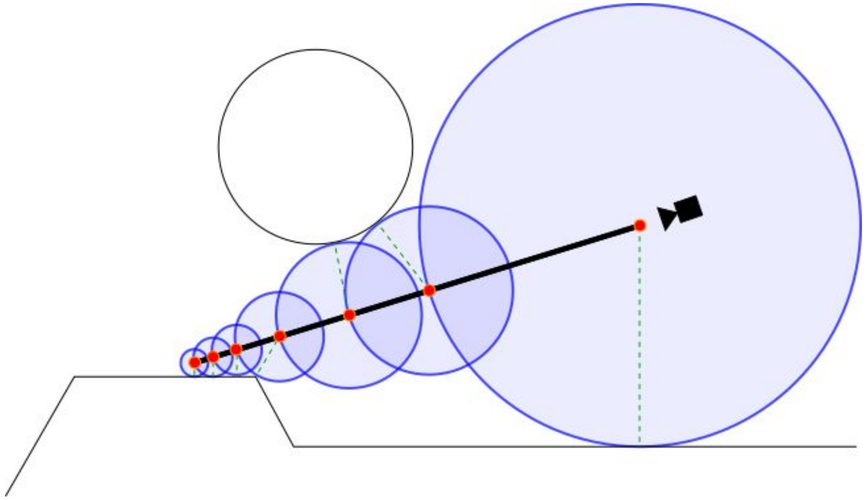
- Ray generation and sampling
  - Representing the scene as a continuous 5D function
    - Can not capture the high frequency details
    - Blurry output frames
  - Positional Encoding
- MLP queries
  - Neural Network replaces large N-d array
  - 100s of times for each pixel
- Volumetric rendering

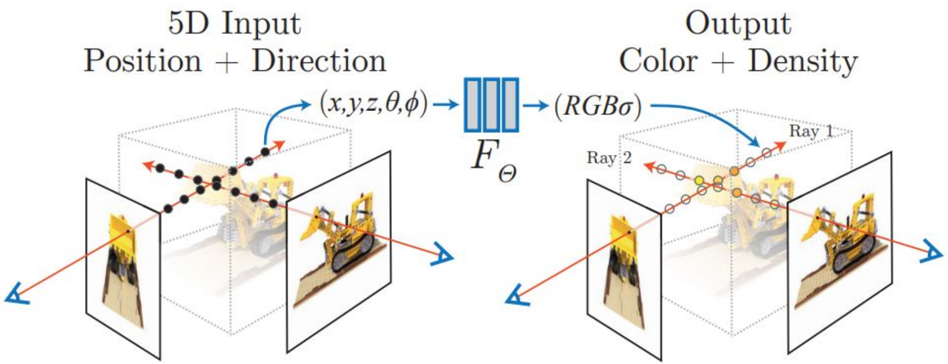


$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t),\mathbf{d})dt\,, \text{ where } T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right).$$



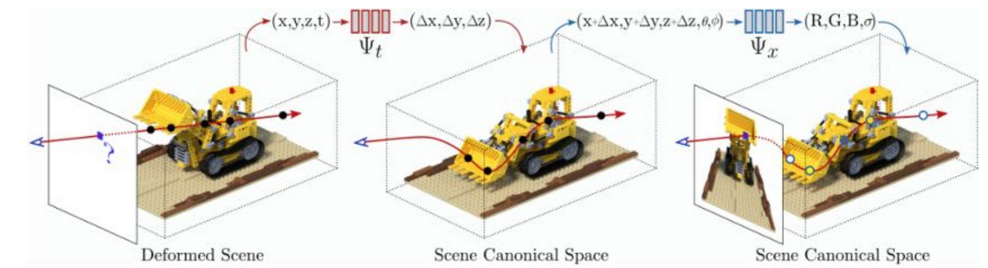
Sampling analogous to ray-marching

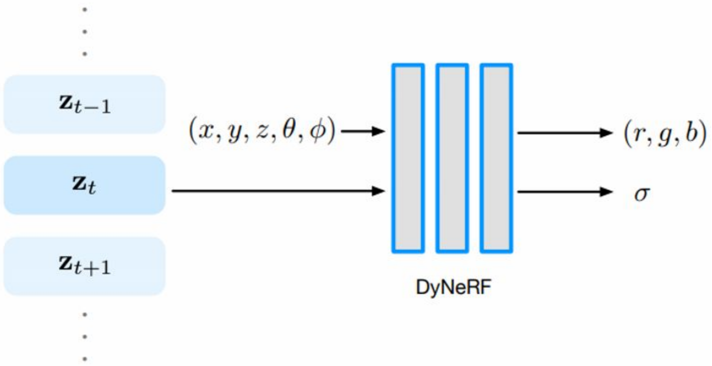




How does NG Work (videos)?

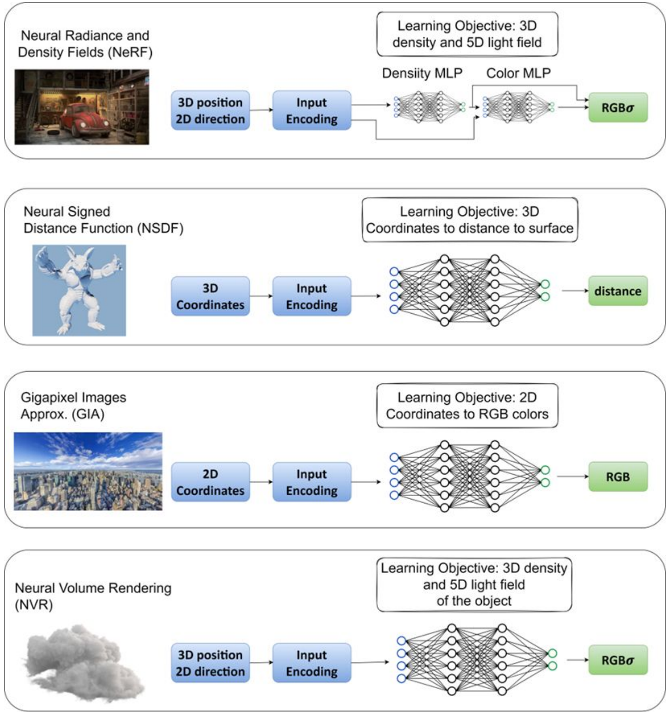
- Deformation based approaches
  - Canonical representation of the network
- Modulation based approaches
  - Learned latent codes
  - Network embeddings
- Research questions to ask!
  - Can compression be used to
    - Accelerate the inference by skipping some work?
    - How much can the memory footprint be reduced without a significant dent on visual fidelity?
    - Speedup vs memory vs visual fidelity tradeoff.





Representative NG Applications/Benchmarks

- Neural radiance and density fields (NeRF)
- Neural signed distance functions (NSDF)
- Gigapixel image approximation (GIA)
- Neural volume rendering (NVR)



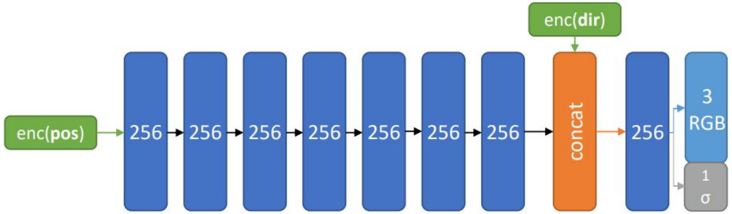
NG Applications

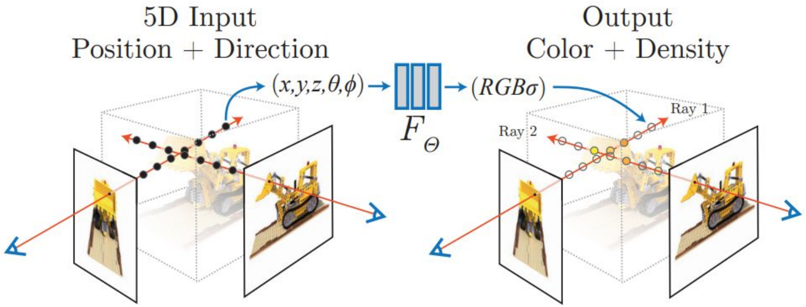
- Neural radiance and density fields (NeRF): The MLP learns the 3D density and 5D light field of a given scene from image observations and corresponding perspective transforms
- Novel view synthesis from a few photos
  - **Rendering:** Capable of rendering extremely high resolution images!
  - **Data Compression:** 3D Geometry structures ~2MB Network
- Virtual tourism on VR headsets
  - Realestate, Tourism etc
- Educational purposes
  - Students looking at NeRF rendered organs (medical), machine parts (mechanical), building structures (civil) etc
- Gaming
  - A combination of classical rendering and NeRF
- Gigapixel image: The MLP learns the mapping from 2D coordinates to RGB colors of a high-resolution image.
- Neural signed distance functions (SDF): The MLP learns the mapping from 3D coordinates to the distance to a surface.
- Neural radiance caching (NRC): The MLP learns the 5D light field of a given scene from a Monte Carlo path tracer.

Algorithmic Optimizations

TITLE	CITED BY	YEAR
NeRF: Representing scenes as neural radiance fields for view synthesis B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, R Ng arXiv preprint arXiv:2003.08934	1223	2020

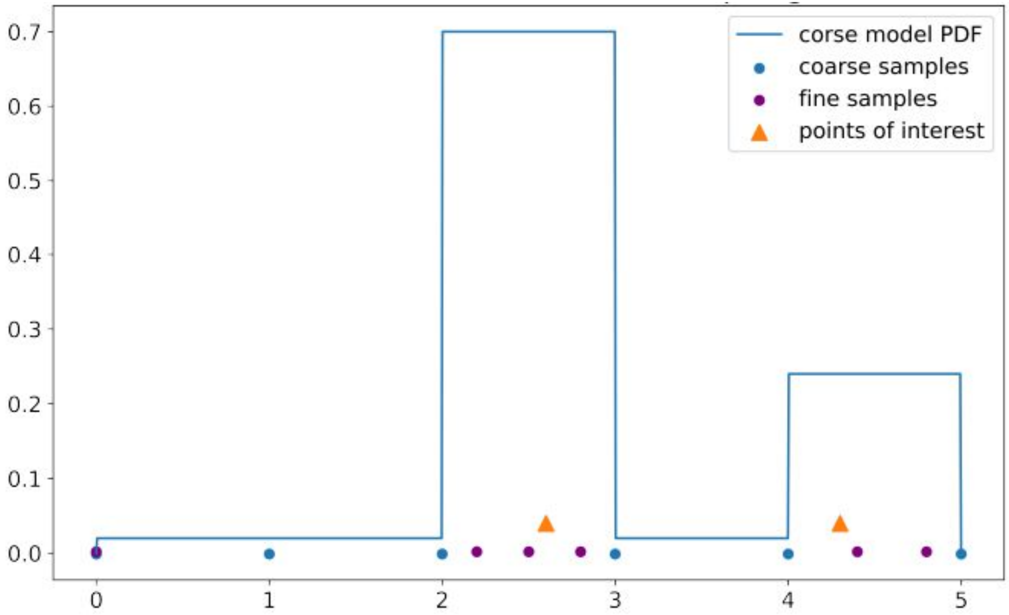
- Problems with NG
  - Inference cost of MLP :
    - 8 layers, 256 hidden neurons each
  - 100s of millions of MLP queries
    - 128 - 356 samples for each pixel (2k resolution)?
- Algorithmic solutions
  - Reduce the number of queries
    - Auxiliary geometric structures (voxels, trees etc)
    - Depth prediction (NNs to predict important samples)
    - Goal: Early Ray Termination (ERT), Empty Space Skipping (ESS).
  - Reduce the size of MLP
    - Learn parts of scene in tiny MLPs then query unique (smaller) MLP for subset of rays.
    - Learn neural network embeddings to generate inputs for MLPs.





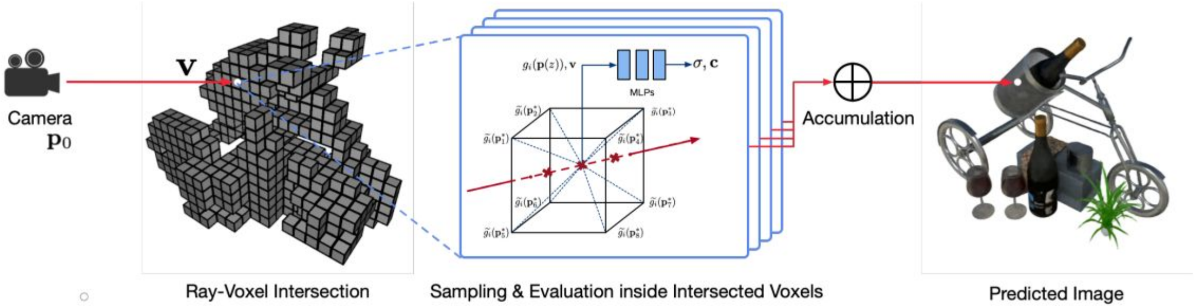
Hierarchical Sampling - coarse/fine grained queries.

- Coarse grained MLP
  - Uniform sampling
- Fine grained MLP
  - Non-Uniform sampling
- Number of samples/ray
  - 128 - 356



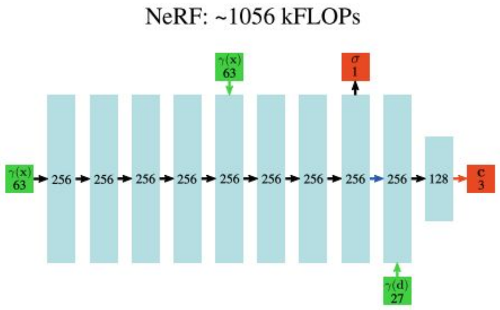
Classical data structures + Neural representations

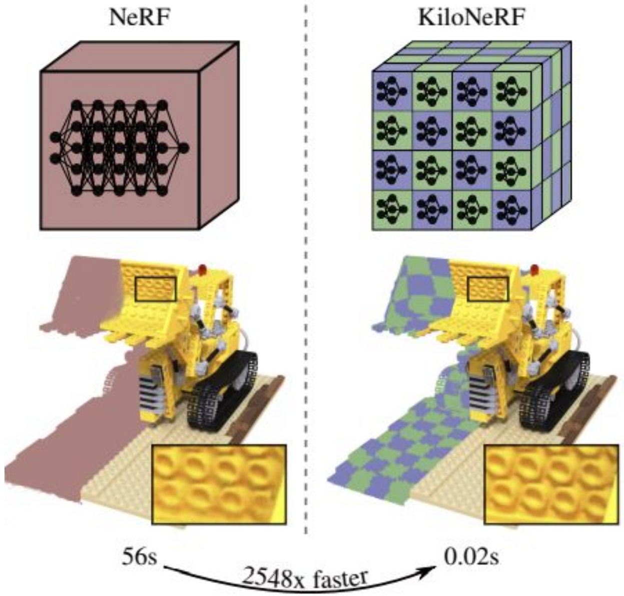
- Neural Sparse Voxel Fields
  - Skip empty space using sparse voxel grid
  - + Efficient sampling, better quality, **~10x speedup**
  - - prior knowledge of the geometry of the scene, complicated training
  - - bigger memory footprint
  - MLP query is still required for every sample



Smaller MLPs

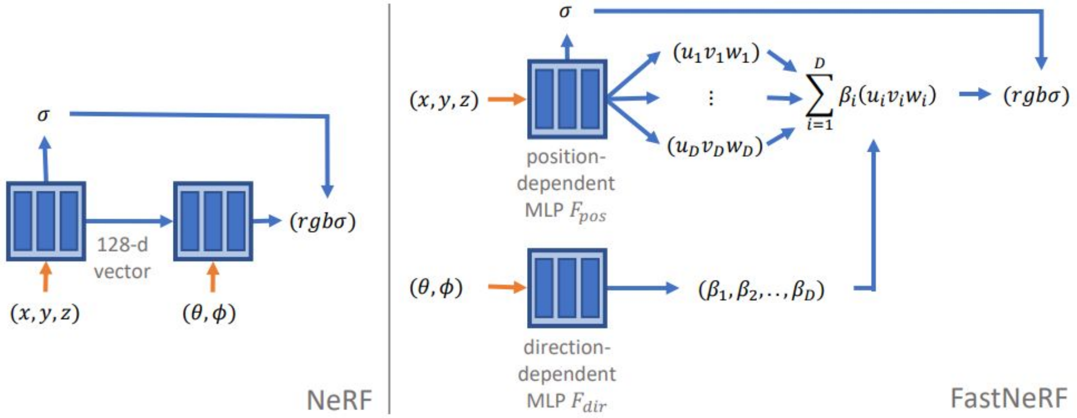
- kiloNeRF
  - + ~ 3 OOM speedup (20 msec) – RTX2080
  - + Smaller model + less samples with EST+ERT trees
  - - 100MB instead of 2MB
  - - Bounded scenes





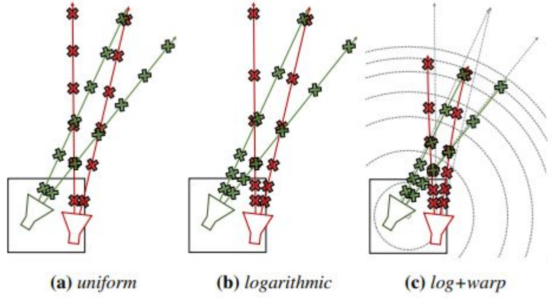
Caching – memoization of NeRF

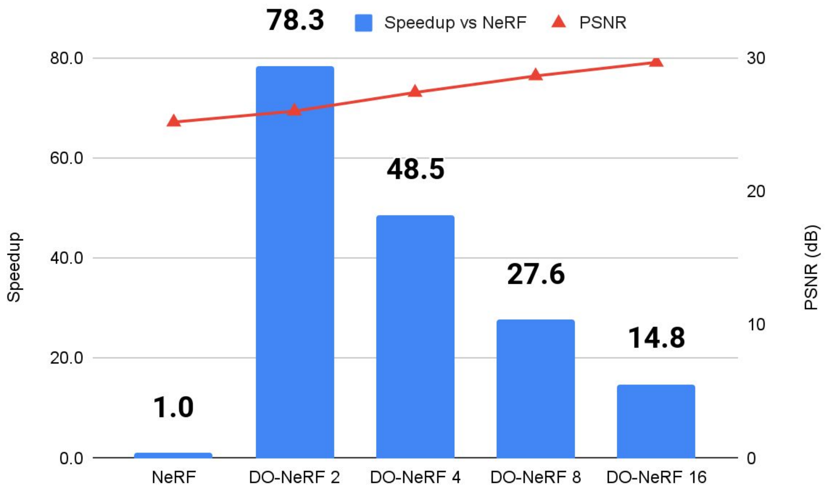
- FastNeRF
  - + ~3 OOM speedup (<10 msec rendering time)
  - - 0.34-10 GB cache – not scalable – increases with resolution
- Fast NeRF is memory bottlenecked instead of compute



NNs for depth estimation

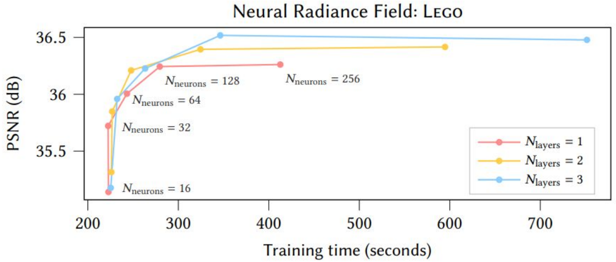
- DONeRF
  - Coarse grained MLP replaced with Depth Oracle Network
  - Use a ground truth depth texture to place samples during training
    - What is the best quality-speed tradeoff that can possibly be reached?
  - Skip empty space using depth prediction – depth oracle network
  - Sampling placement strategy - log + warp
  - Oracle net solves the classification task
  - DO MLP: One query for each ray; 8 layers, 256 nodes/layer
  - NeRF MLP: One query for each sample
  - 2-16 MLP queries are still required for every pixel

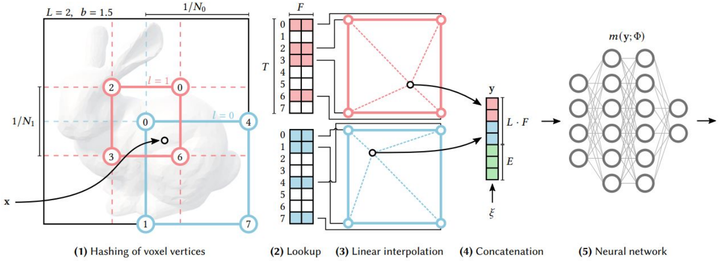




Instant NGP - multi resolution hash encoding

- Positional enc. - multi-res. hash enc.
- Trainable encoding parameters
  - Multi-res. voxel vertices
  - 20X fewer parameters vs dense voxel grids
  - Predictable mem. layout of hash tables - good caching
- 3 - 25 samples per ray
- Linear interpolation to find nearest vertices
- ~1 OOM smaller MLP
  - 1 to 3 layers, 16 to 256 nodes / layer
- Memory: ~200kB to 100MB; Speedup ~ 100s msec
- Potentially, much more suited for in-memory, near-memory architecture.





Does NG Need HW Support?

Extended Reality Systems Have Strict PPA Requirements

Metric	Varjo VR-3 [19]	Ideal VR [17], [20]	Microsoft HoloLens 2	Ideal AR [17], [20], [21]
Resolution (MPixels)	15.7	200	4.4 [22]	200
Field-of-view (Degrees)	115	Full: 165×175 Stereo: 120×135	52 diagonal [23], [24]	Full: 165×175 Stereo: 120×135
Refresh rate (Hz)	90	90 – 144	120 [25]	90 – 144
Motion-to-photon latency (ms)	< 20	< 20	< 9 [26]	< 5
Power (W)	N/A	1 – 2	> 7 [27]–[29]	0.1 – 0.2
Silicon area (mm^2)	N/A	100 – 200	> 173 [27], [30]	< 100
Weight (grams)	944	100 – 200	566 [22]	10s

Component	Parameter	Range	Tuned	Deadline
Camera (VIO)	Frame rate Resolution Exposure	15 – 100 Hz VGA – 2K 0.2 – 20 ms	15 Hz VGA 1 ms	66.7 ms – –
IMU (Integrator)	Frame rate	≤ 800 Hz	500 Hz	2 ms
Display (Visual pipeline + Application)	Frame rate Resolution Field-of-view	30 – 144 Hz ≤ 2K ≤ 180°	120 Hz 2K 90°	8.33 ms – –
Audio (Encoding + Playback)	Frame rate Block size	48 – 96 Hz 256 – 1024	48 Hz 1024	20.8 ms –

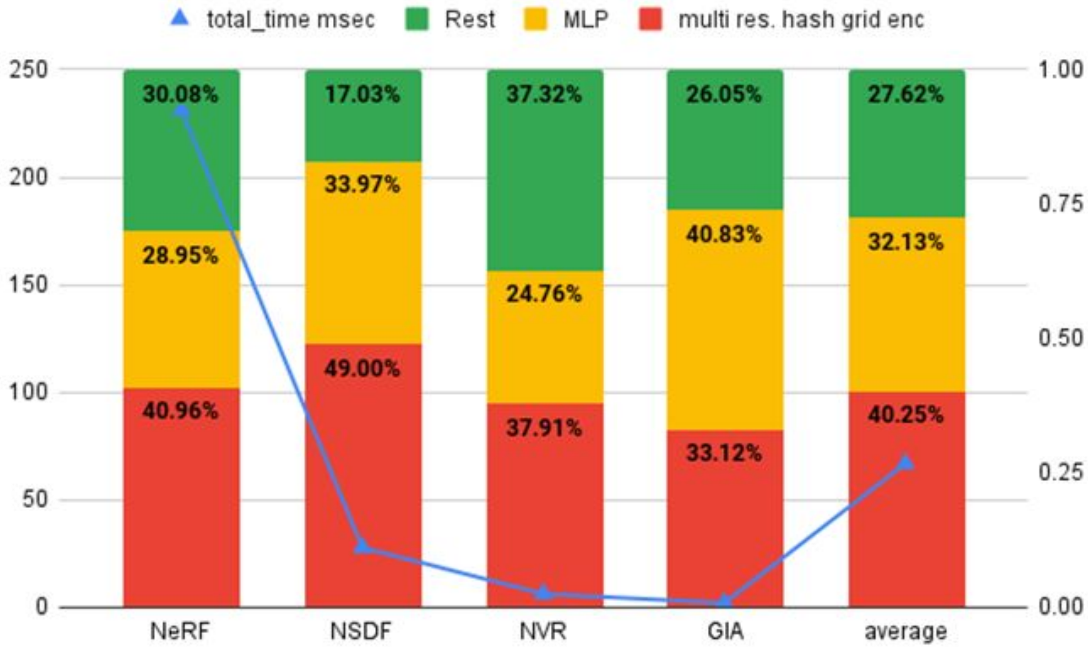
Approximate	Current	Desired
Res (Mpixels)	4	200
Power (W)	10	0.1
Weight (g)	500	10
...	...	...

Table taken from the illixr project.

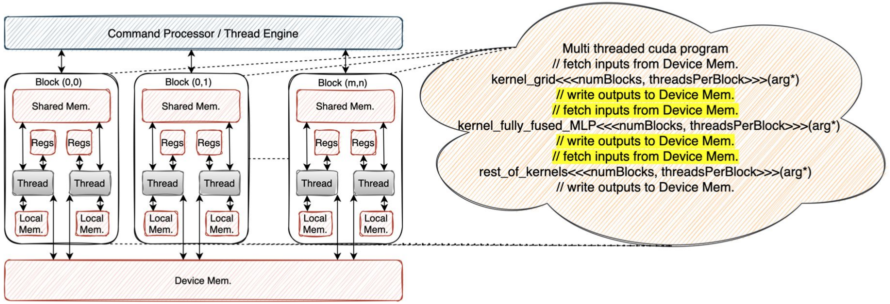
Illixr is an open source extended reality prototyping and evaluation tool

Many different deadlines need to be met to ensure a high-quality user experience!

Performance on RTX 3090

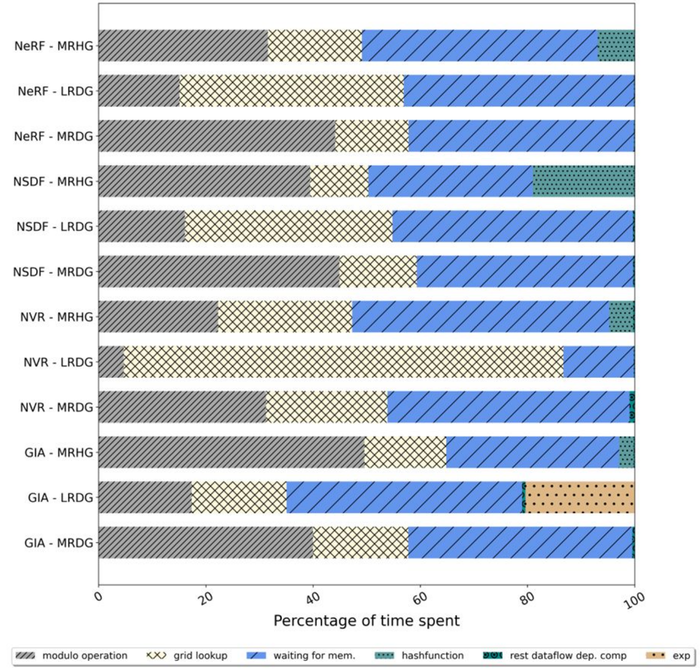


Neural Graphics on RTX3090

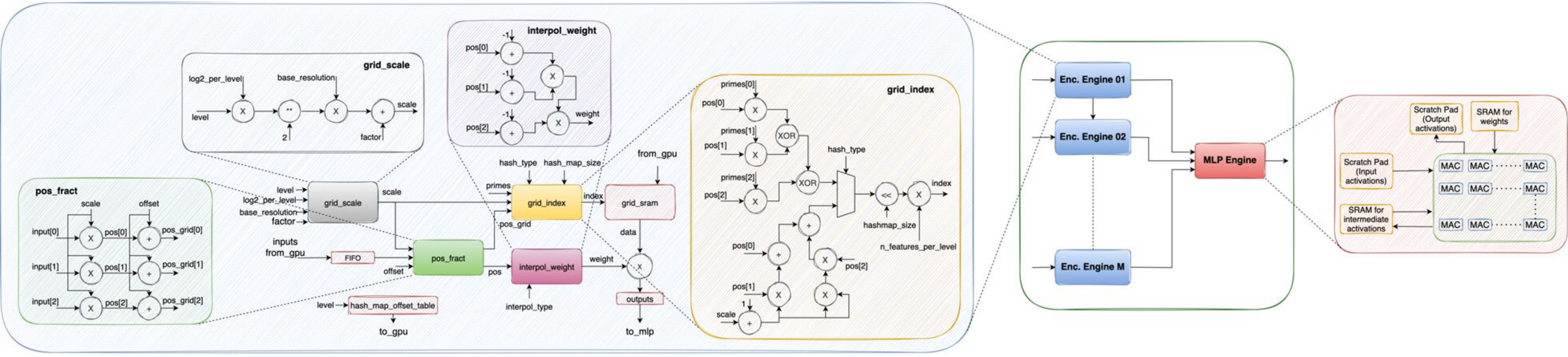


App.-Kernel	Grid Size/Block Size	Comp. Util. per kernel call	Mem. Util. per kernel call	Kernel Calls	Comp. Util. avg. across application	Mem. Util. avg. across application
NeRF multi res. hashgrid	(3853;16;1)/(512;1;1)	61.73	72.85	59	40.63	72.02
NeRF MLP	(3853;16;1)/(512;1;1)	34.3	65.2	118	33.36	63.07
NSDF multi res. hashgrid	(1823;16;1)/(512;1;1)	73.08	43.54	256	15.97	30.8
NSDF MLP	(1823;16;1)/(512;1;1)	38.13	71.74	256	9.76	18.28
NVR multi res. hashgrid	(403;16;1)/(512;1;1)	52.5	59.03	48	18.67	30.36
NVR MLP	(403;16;1)/(512;1;1)	36.51	67.01	48	11.51	21.05
GIA multi res. hashgrid	(4050;16;1)/(512;1;1)	82.87	62.23	1	82.87	62.23
GIA MLP	(4050;16;1)/(512;1;1)	39.1	72.22	1	39.1	72.22
NeRF multi res. densegrid	(3966;8;1)/(512;1;1)	71.39	91.81	45	57.37	72.31
NeRF MLP	(3966;8;1)/(512;1;1)	39.53	68.4	90	34.51	62.31
NSDF multi res. densegrid	(1823;8;1)/(512;1;1)	76.1	48.25	244	18.38	21.28
NSDF MLP	(1823;8;1)/(512;1;1)	41.66	73.49	244	11.06	19.41
NVR multi res. densegrid	(403;8;1)/(512;1;1)	57.38	56.8	48	17.41	22.43
NVR MLP	(403;8;1)/(512;1;1)	39.83	67.67	48	12.17	20.59
GIA multi res. densegrid	(4050;8;1)/(512;1;1)	78.53	65.83	1	78.53	65.83
GIA MLP	(4050;8;1)/(512;1;1)	42.89	73.07	1	42.89	73.07
NeRF low res. densegrid	(3980;2;1)/(512;1;1)	53.83	49.74	43	31.17	59.57
NeRF MLP	(3980;2;1)/(512;1;1)	39.41	68.17	86	35.5	64.1
NSDF low res. densegrid	(1823;2;1)/(512;1;1)	55.88	45.52	260	7.21	20.07
NSDF MLP	(1823;2;1)/(512;1;1)	41.37	72.98	260	10.34	18.14
NVR low res. densegrid	(403;2;1)/(512;1;1)	22.71	69.16	48	6.29	22.71
NVR MLP	(403;2;1)/(512;1;1)	39.2	66.58	48	12.11	20.48
GIA low res. densegrid	(4050;2;1)/(512;1;1)	66.15	59.12	1	66.15	59.12
GIA MLP	(4050;2;1)/(512;1;1)	42.87	73.02	1	42.87	73.02

Waiting for Long Scoreboard to Resolve Global Mem. req.



Neural Fields Processor

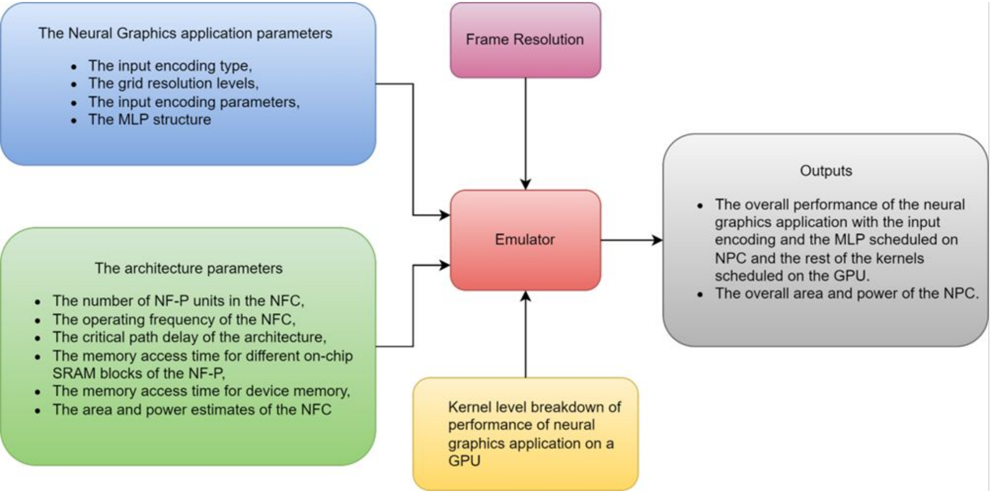


a) Encoding Engine

c) Neural Fields Processor (NFP)

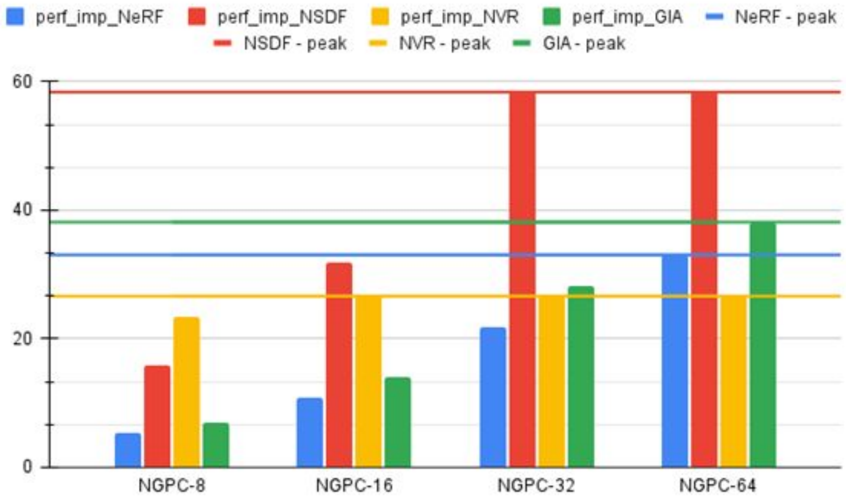
b) MLP Engine

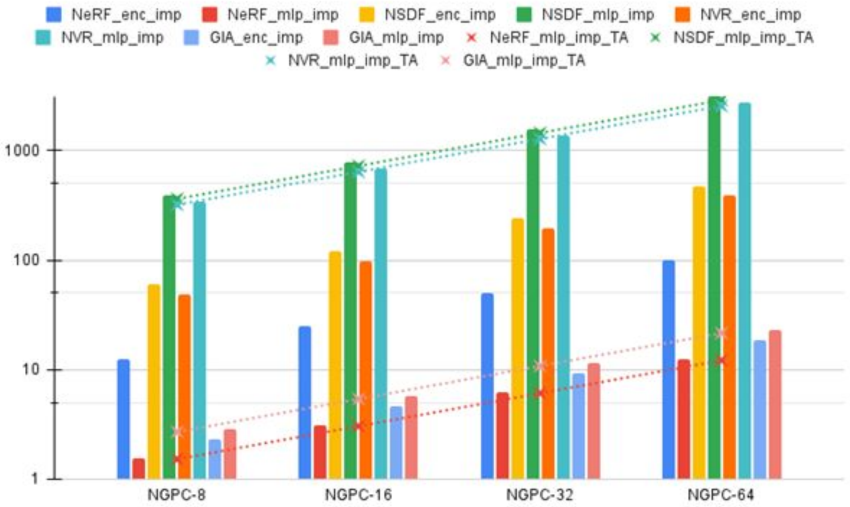
Evaluation



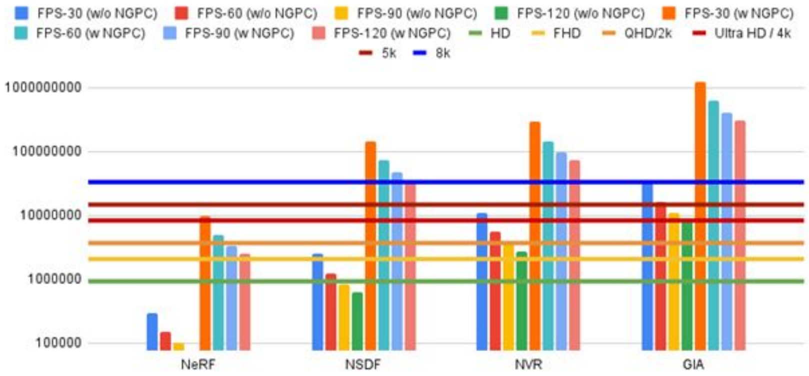
App.	Input BW (GB/s)	Output BW (GB/s)	Totoal BW (GB/s)	Access time (ms)
NeRF	69.523	46.349	231.743	4.126
NSDF	34.761	34.761	69.523	1.238
GIA	34.761	34.761	69.523	1.238
NVR	34.761	34.761	69.523	1.238

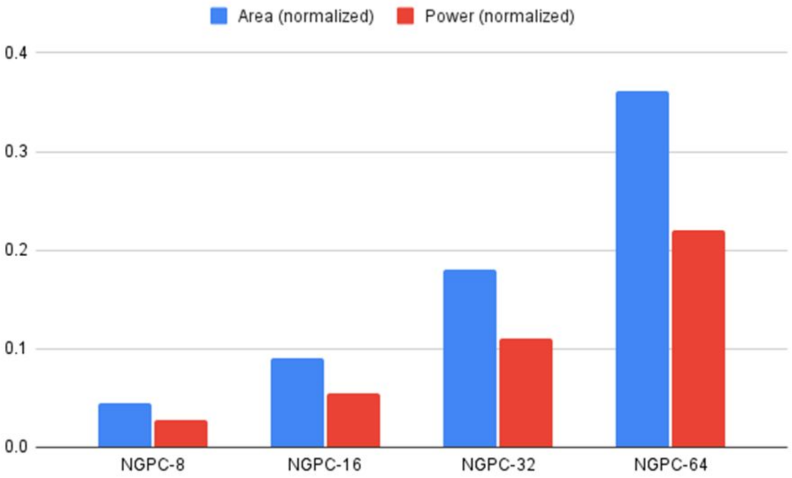
Estimated Performance Improvements





Estimated FPS improvements





Conclusion

- “If not NeRF, some form of Neural Rendering is here to stay” – Anton Kaplan
- XR has stringent PPA requirements
  - Latency, Power, Energy
  - Power gap is ~2OOMX
  - Performance gap for unbounded scenes is ~1OOM - 2OOM
- Rendering high quality images is difficult even on high end systems
- NG is a promising recent alternative to classical rendering methods
- We proposed “a solution” to accelerate NG in HW
  - Configurable enough to run a wide class of NG algorithms
  - Scalable architecture
    - Integrated on edge, desktop and/or embedded devices depending upon the use-case/application
    - Further SW/HW optimizations are required to minimize power and energy footprints for HMDs.

Discussion / Questions!?