# Implicit Mapping at large scale

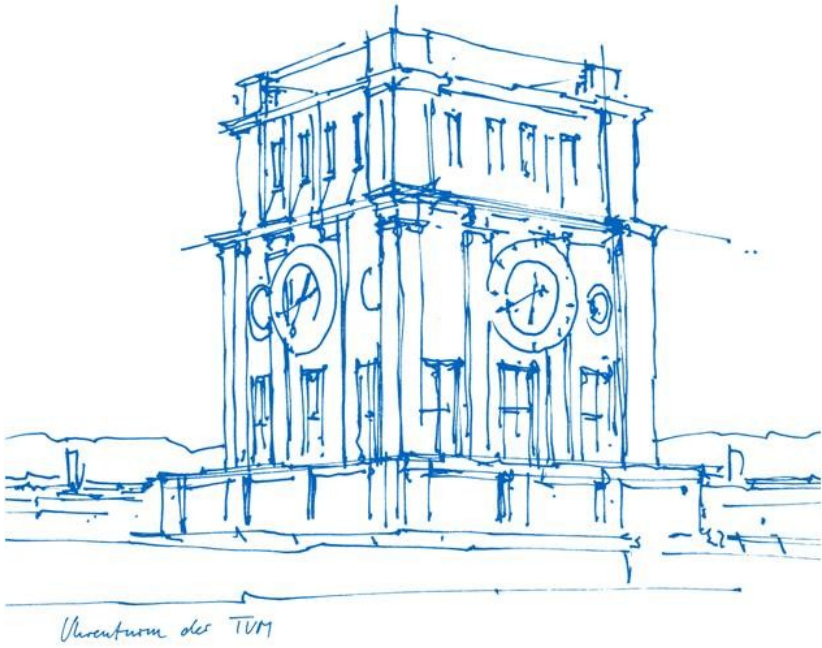
Alican Karaomer

Technische Universität München

TUM School of Computation, Information and Technology

Seminar: Robot Perception & Intelligence

Winter Semester 23/24



# Introduction

- The studies are aimed at solving the problems of current mapping methods such as memory consumption, scalability, sparse data in large environments and speed.

- The two methods bring innovations in different areas. Mega-NERF works on drone imagery, while SHINE-Mapping works on data from LIDAR sensor.

# Introduction

- Shine-Mapping introduces incremental mapping system, memory efficient representations and its own loss function.

- Mega-NERF introduces a reformulation of the NeRF architecture that improves efficiency both in training and rendering.

# Outline

1. SHINE-Mapping
   - Related Works
   - Method Description
   - Experiments and Results
   - Future Work
   - Summary
2. Mega-NERF
   - Related Works
   - Method Description
   - Experiments and Results
   - Future Work
   - Summary

# Related Works – Shine MAPPING

- There are various mapping representations, including surfel-based methods, triangle meshes, histograms, and octree-based occupancy representations. Volumetric integration techniques, like truncated signed distance function (TSDF), enable real-time reconstruction, inspired by Newcombe et al.'s influential work.

- The emergence of neural network-based representations, like NeRF, motivates researchers in mapping and reconstruction. Recent studies explore implicit representations for 3D scene reconstruction, with a noted gap in research for LiDAR data, primarily designed for smaller indoor scenes. For large scale mapping, NICE-SLAM is one of the popular studies in this area, but it also suffers from the forgetting problem due to memory consumption.

# Method Description – SHINE Mapping

**Implicit Neural Map Representation**

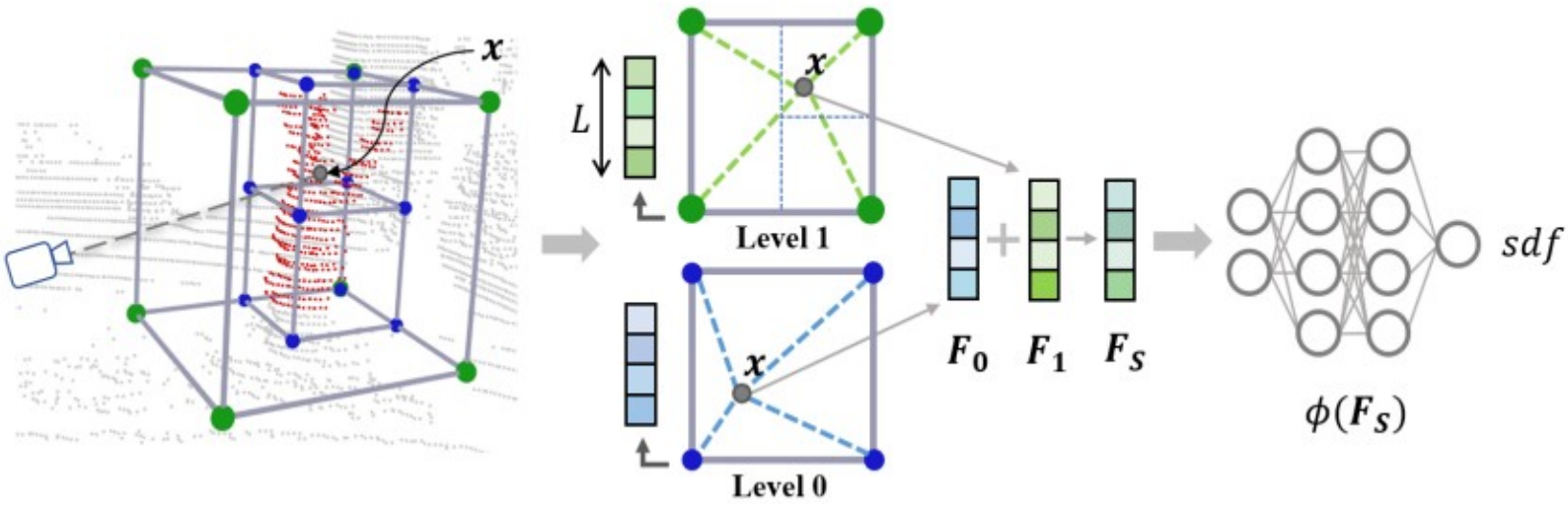



**Image from Haithem Turki et.al 2022**

# Method Description – SHINE Mapping

**Training Pairs and Loss Function**

$$S(x) = 1/(1 + e^{x/\sigma})$$

**σ: control the function's flatness and indicates the magnitude of the measurement noise**

$$o_i = S(f_\theta(\boldsymbol{x}_i))$$

$$L_{\text{bce}} = l_i \cdot \log(o_i) + (1 - l_i) \cdot \log(1 - o_i),$$

$$L_{\text{batch}} = L_{\text{bce}} + \lambda_e \underbrace{\left( \left\| \frac{\partial f_\theta(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i} \right\| - 1 \right)^2}_{\text{Eikonal loss}},$$

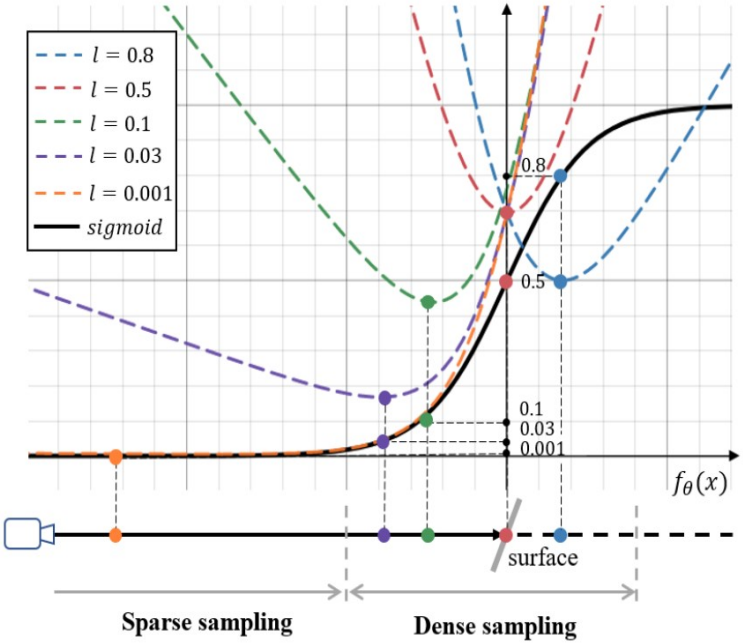



**Image from Haithem Turki et.al 2022**

# Method Description – SHINE Mapping

**Incremental Mapping Without Forgetting**

$$L_r = \sum_{i \in A} \Omega_i (\theta_i^t - \theta_i^*)^2$$

$\theta_i^t$ : **Current Parameters**

$\theta_i^*$ : **Parameter Values converged in training**

$\Omega_i$ : **Importance**

**A : Set of local feature parameters**

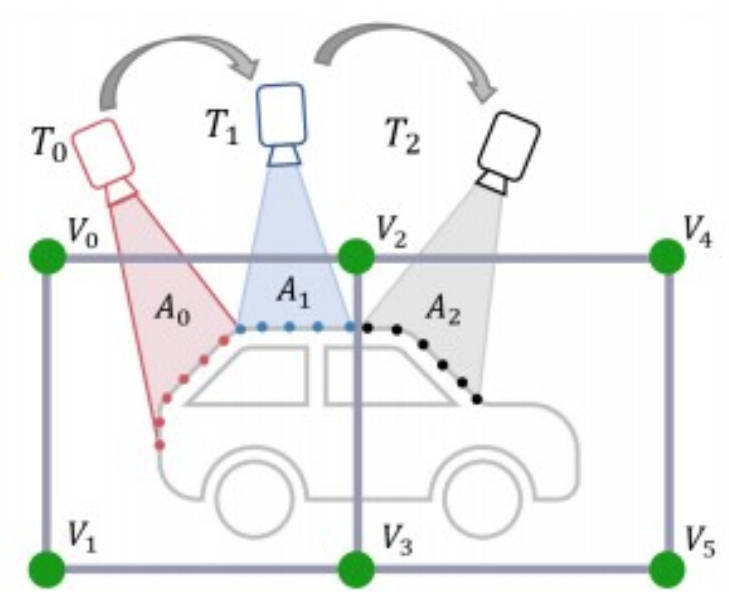



**Image from Haithem Turki et.al 2022**

# Method Description – SHINE Mapping

**Incremental Mapping Without Forgetting**

$$\Omega_i = \min\left(\Omega_i^* + \sum_{k=1}^{N}\left\|\frac{\partial L_{\text{bce}}(\boldsymbol{x}_k, l_k)}{\partial \theta_i}\right\|, \Omega_m\right)$$

**after the convergence of each scan's training**

$\Omega_i^*$: **Previous importance value**

$\Omega_m$: **Limit the weight to prevent gradient explosion**

$\frac{\partial L_{\text{bce}}(\boldsymbol{x}_k, l_k)}{\partial \theta_i}$: **Change of the loss on previous data**

**Image from Haithem Turki et.al 2022**

# Method Description – SHINE Mapping

**Incremental Mapping Without Forgetting**

$$L_{\text{incr}} = L_{\text{bce}} + \lambda_e L_{\text{eikonal}} + \lambda_r L_r$$

**Image from Haithem Turki et.al 2022**

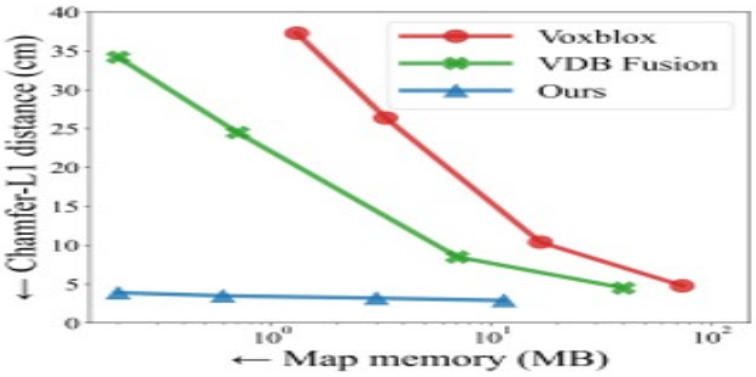
# Experiments and results – SHINE Mapping

TABLE II: Quantitative results of the reconstruction quality on the *MaiCity dataset*. We report the distance error metrics, namely completion, accuracy and Chamfer-L1 in cm. Additionally, we show the completion ratio and F-score in % with a 10 cm error threshold.

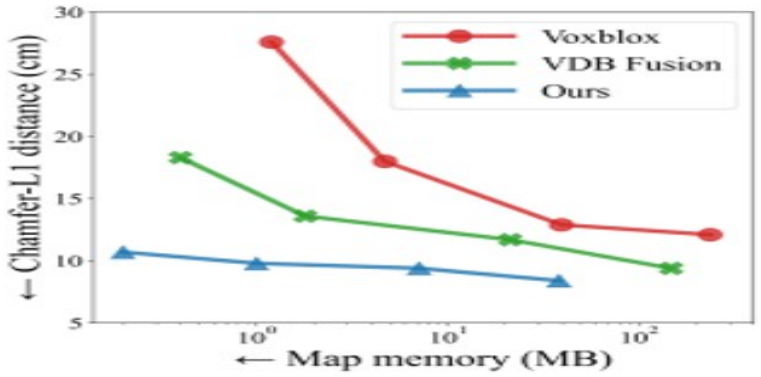
| Method | Comp. ↓ | Acc. ↓ | C-l1 ↓ | Comp.Ratio ↑ | F-score ↑ |
|---|---|---|---|---|---|
| Voxblox | 7.1 | 1.8 | 4.8 | 84.0 | 90.9 |
| VDB Fusion | 6.9 | 1.3 | 4.5 | 90.2 | 94.1 |
| Puma | 32.0 | 1.2 | 16.9 | 78.8 | 87.3 |
| Ours + DR | 3.3 | 1.5 | 3.7 | 94.0 | 90.7 |
| Ours | **3.2** | **1.1** | **2.9** | **95.2** | **95.9** |

TABLE III: Quantitative results of the reconstruction quality on the *Newer College dataset*. We report completion, accuracy and Chamfer-L1 in cm. Additionally, we show the completion ratio and F-score in % calculated with a 20 cm error threshold.

| Method | Comp. ↓ | Acc. ↓ | C-l1 ↓ | Comp.Ratio ↑ | F-score ↑ |
|---|---|---|---|---|---|
| Voxblox | 14.9 | 9.3 | 12.1 | 87.8 | 87.9 |
| VDB Fusion | 12.0 | 6.9 | 9.4 | 91.3 | 92.6 |
| Puma | 15.4 | 7.7 | 11.5 | 89.9 | 91.9 |
| Ours + DR | 11.4 | 11.1 | 11.2 | 92.5 | 86.1 |
| Ours | **10.0** | **6.7** | **8.4** | **93.6** | **93.7** |




(a) MaiCity




(b) Newer College

**Image from Haithem Turki et.al 2022**

# Future Works – SHINE Mapping

**Integration with Sensor Fusion**: Sensor fusion techniques to leverage data from multiple sources, such as cameras, IMUs, and other sensors

**Generalization to Other Sensor Modalities:** Investigating usage of other range sensors like depth cameras or radar

**Adaptation to Dynamic Environments:** Researching methods to adapt SHINE-Mapping to dynamic environments, where the scene undergoes changes over time.

# Summary – SHINE Mapping

SHINE-Mapping presents a novel approach to large-scale 3D SDF mapping using a LIDAR sensor with its own map presentation, loss function and incremental mapping method.

# Related Works – Mega NeRF

- Conventional NeRF rendering falls well below the interactive threshold, but some techniques such as Plenoctree, SNeRG and FastNeRF speed up the process, which Mega-NeRF influence from these techniques.
- Most NeRF-related work targets indoor areas, but NeRF++ handles unbounded environments well, which Mega-NeRF influence these techniques.
- There are several works to speed up model training such as Pixel-NeRF, IBRNet and GRF.
- For large-scale reconstruction, Mega-NeRF is inspired by Agarwal et al's seminal "Building Rome in a Day".

# Method Description – Mega-NERF

## Model Architecture

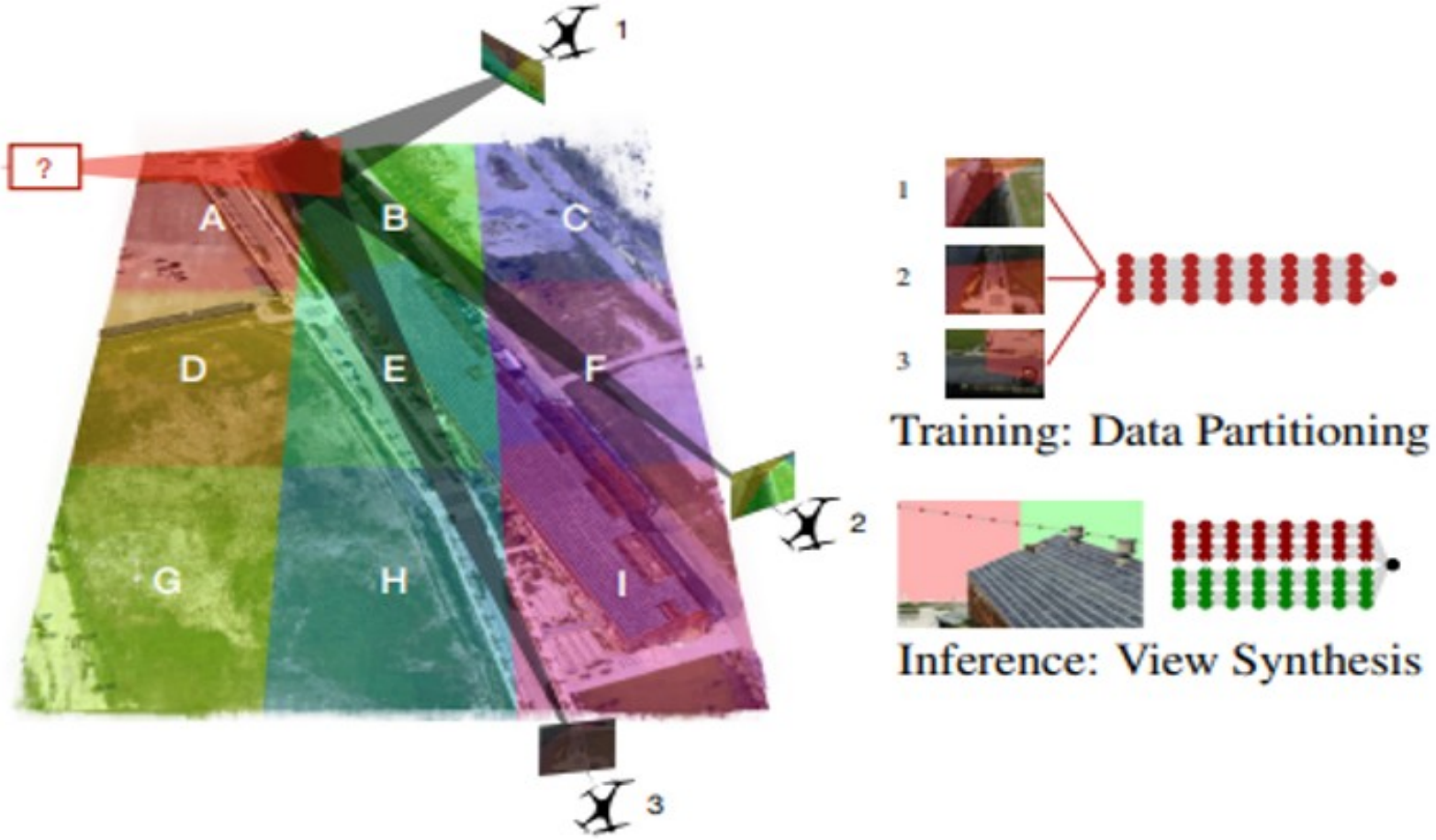



**Image from Xingguang Zhong et.al 2023**

# Method Description – Mega-NERF

## Model Architecture

### Foreground and background decomposition

- They are subdividing the scene into a foreground volume enclosing all camera poses and a background covering the complementary area. Both modeled with separate Mega-NeRFs.

- They are using same ray casting formula and 4D outer volume parameterization as NeRF++ but they are using ellipsoid rather than a sphere to achieve tighter bounds and also use camera altitude measurements to not querying underground regions.

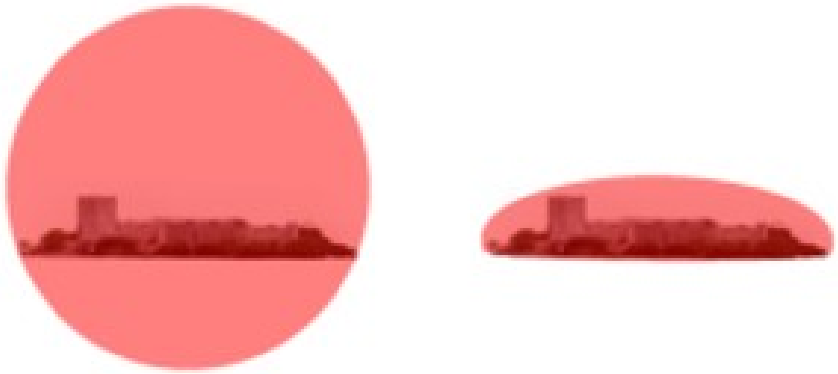


**Image from Xingguang Zhong et.al 2023**

# Method Description – Mega-NERF

## Training

### Spatial Data Parallellism

- As each Mega-NeRF submodule is a self-contained MLP they train them in parallel with no inter communication. They train each submodule with geometry aware pixel-data partitioning, making use of only those pixels whose rays intersect that spatial cell.

### Spatial Data Pruning

- Initial assignment of pixels to spatial cells is based on camera positions, because scene geometry is not known at initialization. Once NeRF gains a coarse understanding of the scene, one could further prune away irrelevant pixels/rays that don’t contribute to a particular NeRF due to an intervening occlude.

# Method Description – Mega-NERF

## Interactive Rendering

## Caching and Temporal Coherence

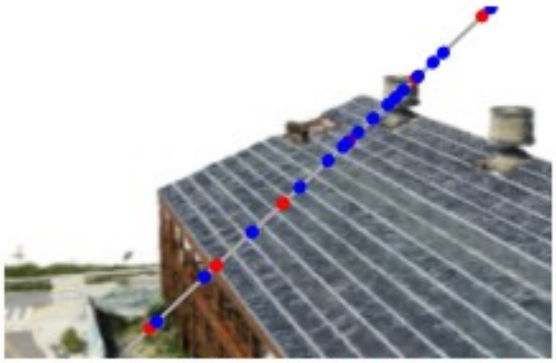
- Mega-NeRF-Dynamic dynamically expands the octree based on the current position of the fly-through. Because of the temporal coherence of camera views, the next-frame rendering can reuse of much of expanded octree and it benefits from the previous refinement and needs to only perform a small amount of incremental work to maintain quality.

# Method Description – Mega-NERF

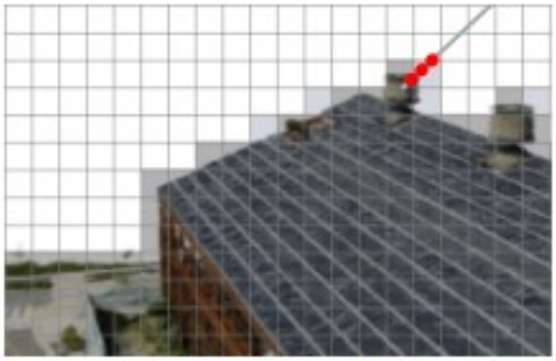
## Interactive Rendering

## Guided Sampling

- Mega-NeRF render rays in a single pass in contrast to NeRF's traditional two-stage hierarchical sampling by using the weights stored in the octree structure



Standard Hierachical Sampling



Guided Sampling

**Image from Xingguang Zhong et.al 2023**

# Experiments and results – Mega NeRF

| | Mill 19 - Building | | | | Mill 19 - Rubble | | | | Quad 6k | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ↑PSNR | ↑SSIM | ↓LPIPS | ↓Time (h) | ↑PSNR | ↑SSIM | ↓LPIPS | ↓Time(h) | ↑PSNR | ↑SSIM | ↓LPIPS | ↓Time(h) |
| NeRF | 19.54 | 0.525 | 0.512 | 59:51 | 21.14 | 0.522 | 0.546 | 60:21 | 16.75 | 0.559 | 0.616 | 62:48 |
| NeRF++ | 19.48 | 0.520 | 0.514 | 89:02 | 20.90 | 0.519 | 0.548 | 90:42 | 16.73 | 0.560 | 0.611 | 90:34 |
| SVS | 12.59 | 0.299 | 0.778 | 38:17 | 13.97 | 0.323 | 0.788 | 37:33 | 11.45 | 0.504 | 0.637 | 29:48 |
| DeepView | 13.28 | 0.295 | 0.751 | 31:20 | 14.47 | 0.310 | 0.734 | 32:11 | 11.34 | 0.471 | 0.708 | 19:51 |
| MVS | 16.45 | 0.451 | 0.545 | 32:29 | 18.59 | 0.478 | 0.532 | 31:42 | 11.81 | 0.425 | **0.594** | **18:55** |
| Mega-NeRF | **20.93** | **0.547** | **0.504** | **29:49** | **24.06** | **0.553** | **0.516** | **30:48** | **18.13** | **0.568** | 0.602 | 39:43 |
| | UrbanScene3D - Residence | | | | UrbanScene3D - Sci-Art | | | | UrbanScene3D - Campus | | | |
| | ↑PSNR | ↑SSIM | ↓LPIPS | ↓Time (h) | ↑PSNR | ↑SSIM | ↓LPIPS | ↓Time(h) | ↑PSNR | ↑SSIM | ↓LPIPS | ↓Time(h) |
| NeRF | 19.01 | 0.593 | 0.488 | 62:40 | 20.70 | 0.727 | 0.418 | 60:15 | 21.83 | 0.521 | 0.630 | 61:56 |
| NeRF++ | 18.99 | 0.586 | 0.493 | 90:48 | 20.83 | 0.755 | 0.393 | 95:00 | 21.81 | 0.520 | 0.630 | 93:50 |
| SVS | 16.55 | 0.388 | 0.704 | 77:15 | 15.05 | 0.493 | 0.716 | 59:58 | 13.45 | 0.356 | 0.773 | 105:01 |
| DeepView | 13.07 | 0.313 | 0.767 | 30:30 | 12.22 | 0.454 | 0.831 | 31:29 | 13.77 | 0.351 | 0.764 | 33:08 |
| MVS | 17.18 | 0.532 | **0.429** | 69:07 | 14.38 | 0.499 | 0.672 | 73:24 | 16.51 | 0.382 | **0.581** | 96:01 |
| Mega-NeRF | **22.08** | **0.628** | 0.489 | **27:20** | **25.60** | **0.770** | **0.390** | **27:39** | **23.42** | **0.537** | 0.618 | **29:03** |

**Image from Xingguang Zhong et.al 2023**

# Experiments and results – Mega NeRF

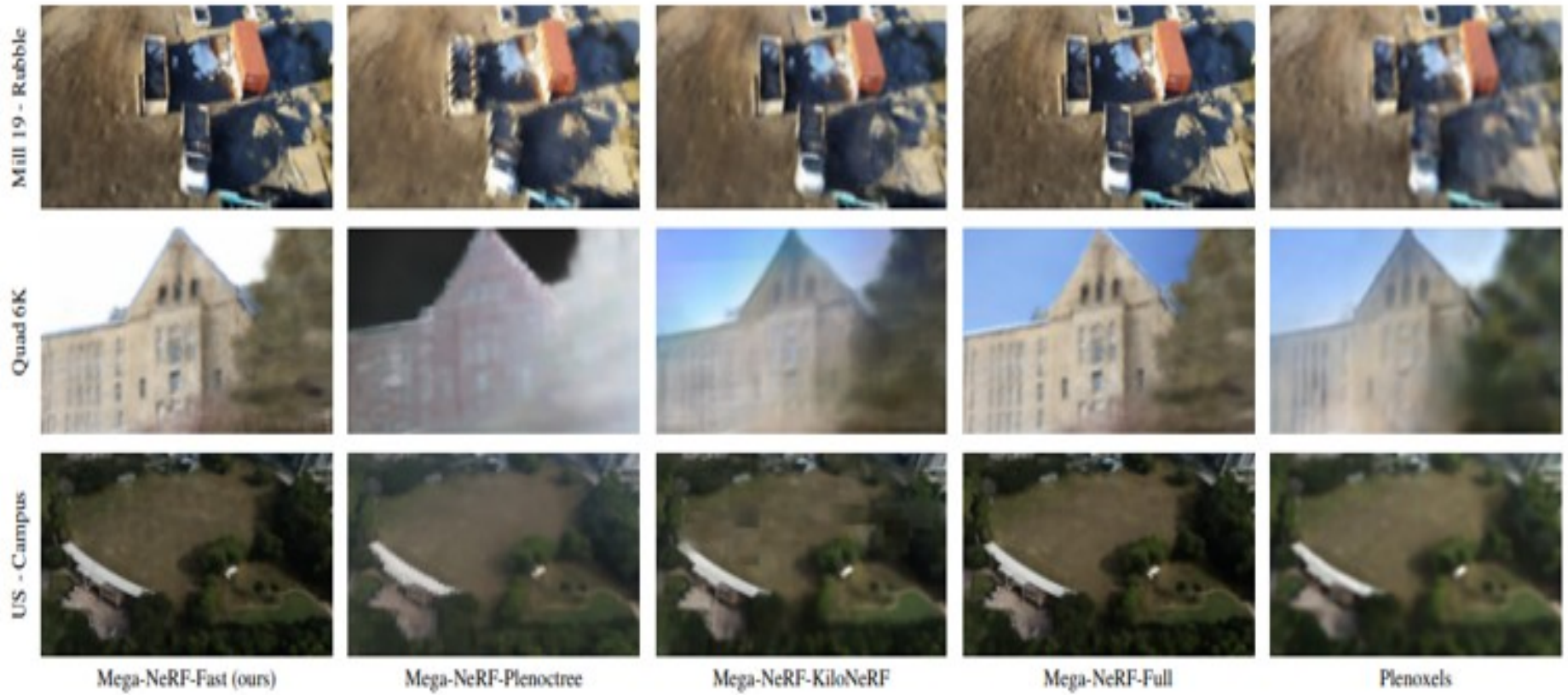



**Image from Xingguang Zhong et.al 2023**

# Future Works – Mega NeRF

**More accurate pose estimation**: Mega-NeRF trained initial models with raw camera poses collected from the drone's GPS and IMU, but they were extremely blurry, then tried PixSFM, which is relatively good, but is working on more accurate pose estimation that could significantly improve model performance or tried another hardware-based solution such as RTK GPS.

**Rendering Speed: :** Mega-NeRF struggle to reach the throughput needed for truly interactive applications. As a future works on this study, focusing the improvement of rendering speed for interactive applications can be a good idea.

**Adaptation to Dynamic Environments:** Researching methods to adapt Mega-NeRF to dynamic environments, where the scene undergoes changes over time.

# Summary – Mega-NERF

Mega-NERF presents a modular approach to building large-scale NeRFs with its own sparse and spatially-aware network structure, parallel trainable NeRF submodules, and improvements for fast rendering.

# Thank You For Listening