Propose to Efficiently Integrate NeRF to SLAM System (Draft)

Jian Zhou

June 22, 2023

1. Introduction

NeRF as an emerging rendering tech can reconstruct the environment in photolike realistic performance attract considerable attention recently. According to the original NeRF paper [1], NeRF, despite high-performance, still has some disadvantages on training time, scalability and mapping unbounded environment. Much research has been done to improve the performance of NeRF in the last three years. For instance, Mip-NeRF [29] and its following work Mip-NeR 360 [16] map the unbounded scenes to a bounded coordinate space that make NeRF be able to model the outdoor environment with high performance. And Point-NeRF uses a combination of implicit NeRF and explicit Point Cloud to accelerate the training process. More details are shown in session 2 literature review. However, these pure NeRF based approaches are not able to handle the mismatch of the input photos and their poses. They count on a perfect poses acquisition method to train their data as this is easy when using simulated datasets. In the real world, perfect poses acquisition is not possible. So, introducing a NeRF to a system that can help it learn the real environment with poses estimation and optimization is necessary.

Under that consideration, many groups have integrated NeRF to SLAM to explore new possibilities. The photo-realistic rendering provided by NeRF cannot be achieved by traditional SLAM methods based on voxel grids or other Point Cloud methods, and such rendering can generate a vast number of additional applications. As a 3D reconstruction method, NeRF naturally finds its way into the backend of SLAM to explore its potential. This is the driving force behind the development of NeRF+SLAM.

After reviewing most of the popular approaches, I have a general understanding of the various directions of development in NeRF. So, I propose the following work on NeRF+SLAM. It combines many state-of-the-art NeRF methods and uses fully explicit or hybrid approaches to create a NeRF map. The technical details are as follows. Firstly, it serves as a state-of-the-art compilation, and secondly, it explores the limitations of fully explicit representation in SLAM. The goal is to accelerate real-time FPS as much as possible.

- I will construct an extendable blocked framework of SLAM.
- Pure Explicit Representing, without MLP like DVGO [11] or Hybrid way like Nice-SLAM [15]. This requires an accelerated computing program in the language of cuda.
- Using Depth to initialize voxel grid, finding surface and their corresponding voxel.
- Trilinear interpolation for continuous representation and high frequency features.
- Dynamic Octree
- Using Hash Encoding to reduce retrieval time [6].
- Cone-like rendering to anti-alias [29].
- Interacting of mapping and tracking.
- Using unbounded scene [16].

This is an updated version of the proposed NeRF+SLAM system. It is updated because Co-SLAM [35] (published at CVPR 2023) has already improved the training and rendering speed of NeRF based SLAM to around 12 Hz, using nearly the same method that I initially proposed.

In the reminder of this proposal, I will first conduct a literature review from the perspectives of pure NeRF improvement attempts in the field and then review the development of NeRF+SLAM approaches. Secondly, I will list my objectives and proposed methodology to achieve that.

2. Literature Review

In basic NeRF, the rendering requires 30 seconds for each frame and the training requires almost one day to converge [1]. The basic NeRF focuses on demonstrating the photorealistic performance of NeRF indicating its great potential on 3D scene reconstruction but didn't optimize the training process and rendering procedures. What's more, scalability is another problem that NeRF needs to solve to be a better mapping technology.

This session will be organized alone the development directions of NeRF's optimizing paths. To make clear the paper's focus and show the development order in time, I use the following regularized expression **[NeRF 2020 [1] Training time],** where **Training time, inference time, scalability, NeRF+SLAM** and **Others** are marked with corresponding color. The marked label says that the paper has prominent contribution in this particular direction but not indicating it just has this specific development.

2.1 Rendering time optimization

This direction of optimization focusses on reducing the time of a well-trained NeRF MLP to render a required scene. Here are some parallel works on reducing rendering time from different approaches. Many papers have some small but efficient tricks to reducing time, but here we will focus on their main ideas.

Neural Sparse Voxel Field (NSFV) 2020 [3] **Inference time,** firstly introduce Voxel into NeRF, which divided the entire space into voxels. Voxels are an extension of concept pixel. It treats a space as a set of uniform 3D grids. Rather than utilizing a single implicit function to model the whole space, the NSVF approach uses a collection of implicit fields, each confined to a specific voxel, all arranged within a sparse voxel octree.
Overall, it stores valid samples in voxels and will skip all the empty voxels when sampling.

KiloNeRF 2021 [7] **inference time :** The trained NeRF is broken down into thousands of smaller NeRFs. This doesn't speed up the training time, it actually might require even more time for training. During the splitting process, these smaller NeRFs are used to learn the well-trained NeRF. It means these small NeRFs are a regression to big NeRF. Logically, there should be some information loss, but it seems the loss is not significant. This method is not likely to be used in SLAM in an efficient way.
Overall, it stores MLP into a set of small sub-MLPs and querying small MLPs is more efficient.

DONeRF 2021[4] inference time: DONeRF substitutes the MLP-based raymarching technique used by basic NeRF [1] with a concise local sampling method that only focuses on crucial samples around surfaces. DONeRF is composed of two networks and is implemented through a five-stage process.
Overall, it skips empty sample points by depth prediction.

FastNeRF 2021[2] inference time: In both NSVF [3] and DONeRF [4], an MLP (Multilayer Perceptron) is still needed for each sample. A cache is used to store the trained NeRF MLP, and a separate network is employed to accomplish this. This separate network further employs a mathematical factorization technique to reduce the number of times the MLP is called upon during inference, thus minimizing the inference time. But so, it requires high memory. 512*512 resolution requires about 4GB, although its rendering rate can reach 200fps.

Overall, it uses a mathematical factorization technique to reduce the number of times of calling MLP to reduce time.

SNeRG 2021 [5] **inference time**: This method stores the diffuse color, density, and feature vectors into a sparse voxel grid. By simply searching for color and density within the voxel grid and performing a series of operations, fast rendering can be achieved, significantly reducing the number of queries to the MLP.

PlenOctrees 2021[8] **Inference time**: Like the method used in SNeRG, this approach also stores information in voxels. However, it utilizes spherical harmonics to represent color and volumetric density.

2.2 Training time optimization

Instant-NPG 2022 [6] **Training time**: Performing hash encoding for each voxel. Its main contribution lies in optimizing the retrieval steps during the training process. Previously, accessing a voxel required multiple retrieval comparisons. With hash encoding, it only requires a single step to locate the voxel. Since the useful voxels only occupy about 2.8%, even if there are hash collisions, it doesn't matter much. Only two layers with 64 bits are used. CUDA is incredibly fast! Further research is needed. Check out Tiny-cuda-nn

Plenoxels 2021 [9] **Training time**: Not using MLP. This tells us the reason for the photorealistic performance of NeRF originates from its volume rendering, not MLP. Trained on Titan, reduced from one day to 11 minutes. The idea is to store spherical harmonics in voxels and represent the scene using a voxel grid. This is extremely important and requires a detailed understanding.

JAXNeRF[10] **Training time** : a framework of multi-GPU computing.

DVGO [11] **Training time**: Voxel representation, posterior activation strategy, coarse-to-fine approach, training completed in 11 minutes. The results are similar to PlenOctree. However, it is not suitable for unbounded and forward-facing scenes. Its posterior activation strategy has been adopted by other well-known NeRF algorithms.

DVGOV2 [34] Training time: Pytorch-vesion DVGO. They extend DVGO to support Forward-facing and Unbounded Inward-facing capturing.

In summary, many papers focus on optimizing specific details. The central idea for reducing inference time is to trade space for time. The key thought for reducing training time is to replace high-frequency expressions with low-frequency ones, while minimizing information loss as much as possible. Notably, Plenoxel suggests that an MLP is not always necessary.

2.3 Scalability

Vox-fusion [21] Scalability: The map is stored in an octree structure, which can be expanded. The voxels in the octree store Signed Distance Function (SDF) values but do not include voxel densities. However, this approach sacrifices some capabilities in terms of synthesizing new viewpoints.

Block-NeRF [18] Scalability. Scalable wide-angle synthesis involves using multiple MLPs placed at different intersections, ensuring the MLPs' receptive fields overlap. When rendering an image at a specific location, the approach involves identifying which blocks are relevant, rendering them individually, and then synthesizing the final image based on visibility. The challenge lies in handling different lighting conditions.

NeRF++【17】Scalability： Partitioning the space into foreground and background, where the foreground is represented using classic NeRF and the background is represented using reverse spherical parameterization, placing the background on a sphere. The foreground and background are stored separately.

Mega-NeRF [19] Scalability: Dividing the scene into spatial cells and dynamically expanding an octree. Separate sets of NeRF models are used to represent the foreground and background respectively.

2.4 NeRF+SLAM

IMAP [12] **NeRF+SLAM**： Real-time, only used a single MLP (Multi-Layer Perceptron), with a memory bank to prevent forgetting. I don't think implicit MLP is suitable for SLAM (Simultaneous Localization and Mapping. It uses 4 hidden layers with 256 perceptron in each layer. Each sample just 200 points. Bounded scene.

NeRF-SLAM[13] **NeRF+SLAM**： Real-time. It utilizes traditional visual odometry as the front-end for Droid-SLAM, with Instant-NGP serving as the back end. NeRF has not been leveraged for tracking purposes.

Orbeez-Slam[14] **NeRF+SLAM**: It is a simple combination of Orb-slam2 and NeRF cannot operate in real-time.

Nice-SLAM (Dec 2021) [15] **NeRF+SLAM**: Pre-allocated voxels are used to represent the scene using three voxel grids: coarse, mid, and fine. The tracking performance is good. Not scalable.

Nicer-SLAM (Feb 2023) [24] **NeRF+SLAM**： It cannot be applied in real-time. It is important to focus on identifying the specific issues and determining which computations are consuming computational resources. Further investigation is needed. I am still confused about its tracking strategy. **This paper is crucial for me to think about tracking.**

BARF (Apr 2021) [25]: Using low-frequency information alignment for Bundle Adjustment optimization.

Volumetric Bundle Adjustment [26]: Doing BA with Octree and voxels.

2.5 other optimization

Depth-supervised NeRF (July 2021) [30] Quality**:** The paper discusses the role of depth in NeRF (Neural Radiance Fields) and how it greatly improves the results. However, due to the limitations of depth acquisition methods solely based on computer vision, its effectiveness is restricted. Nevertheless, it inspires us to incorporate depth in neural networks.

MERF (Feb 2023) [31] Memory**:** The paper focuses on memory optimization, where it applies nonlinear mapping after interpolation and summation, inspired by DVGO [11]. It introduces an unbounded segmented contraction mapping function. As for the meaning of 'ray-AABB,' I'm not entirely sure either. It may require further reading and understanding of the paper to provide an accurate explanation.

Mip-nerf [29] Anti-aliasing: In traditional ray tracing, rays extend from the origin along specific directions and intersect with objects in the scene to gather information. By using cone-shaped rays, a broader sampling can be performed in a particular direction to obtain more scene information.

Mip-nerf 360 [16] Bound: Mapping an unbounded scene to a bounded scene.

Point-NeRF 2022[22] Point Cloud: Storing feature vectors in point clouds.

2.6 Review for CVPR 2023

vMAP (Feb 2023) [33] NeRF+SLAM**:** A combination of NeRF and object segment. This work creates MLP for each object in the scene. I don’t think it is a good pipeline to segment scenes while creating a map. It is coupled!

CO-SLAM (Apr 2023) [35] NeRF+SLAM: This paper is in collision with my original Proposed work. Hash grid (From Instant-ngp) + voxel explicit representation (From DVGO) to accelerate training as a real-time SLAM. It also achieves global BA by a certain approach.
It replaces several common nouns with many other nouns. Among them, 'Parametric embeddings' is used to refer to 'Voxel-based,' and 'coordinate-based network' refers to NeRF's MLP. 'Hash-grid' is used to refer to the hash encoding in Instant-NGP. I felt very confused while reading this paper.

What’s more, KiloNeRF (referred in this paper) is not a training time speed up work but a render accelerating work. Perhaps 'coordinate-based' and 'parametric embeddings' come from broader concepts, and perhaps the negative emotions arise from the clash between my own ideas and the article's content. Subjectively, I don't like this paper.

ESLAM (Nov 2022) [32] TSDF+SLAM: This paper uses TSDF as the method of 3D reconstruction. It also uses a "multi-scale axis-aligned feature planes" instead of voxel, which I don't understand at the moment and need to know more about.

3. Proposed work

My original proposal was to construct a real-time fully explicit or hybrid NeRF-SLAM system using all the accelerating SOTA approaches like Plenoxels, DVGO and hash encoding (Instant-NGP). But since Co-SLAM (Published at CVPR 2023) has already completed this work, I now turn this work into a base of future work. Also, to mitigate the effect of another collision that may happen in the future, I propose to create some branches for my project. The structure of my project is shown in figure x.

In my updated plan, I will first construct a Base SLAM system (Original Proposal), which uses hybrid or fully explicit NeRF and apply several tricks (such as hash-encoding, importance-sampling, sparse voxels, octrees) to make the training and rendering time to be minimal while keeping a good performance. Then, develop the system from one of the directions in figure 1. Their priority ranks are shown in table 1. Details of them are shown in session 3.1 and 3.2.

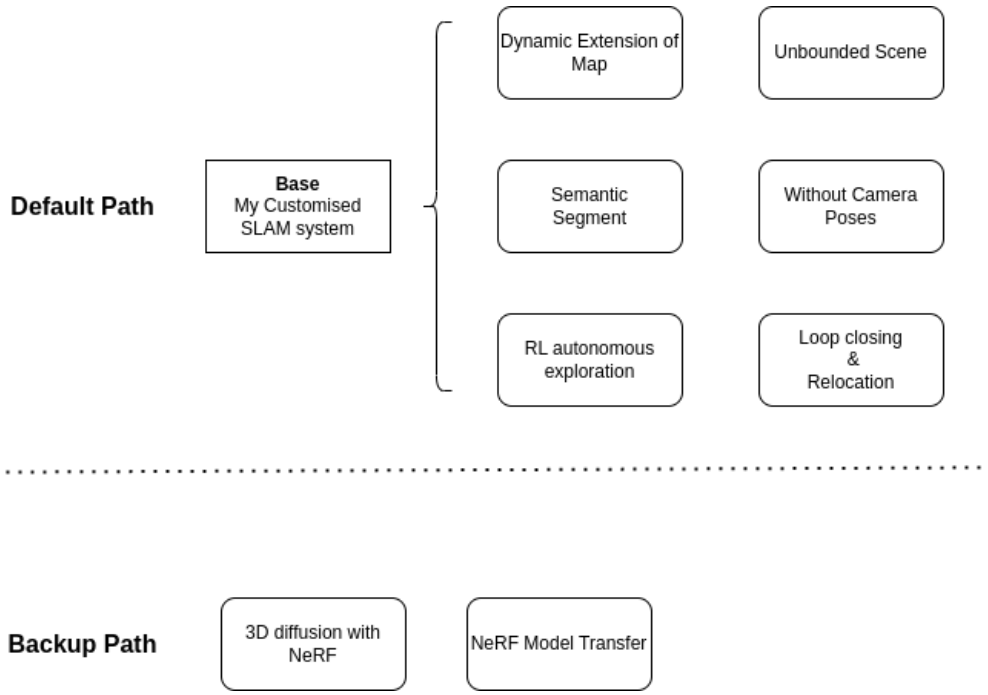

Figure 1: Possible Research Directions

Table 1: Priority ranks of future work	
Dynamic Extension of Map	1
RL autonomous exploration	2
Without Camera Initial Poses	3
Unbounded scene	4
Semantic segment	5
Loop closing & Relocation	6
3D diffusion with NeRF	7
NeRF model transfer	8

3.1 Proposed work on the NeRF based SLAM.

The current state of NeRF-based SLAM faces challenges in mapping, particularly mapping and regarding outdoor environments without boundaries and the expansion of dynamic maps. Most existing NeRF-based SLAM methods rely on pre-defined voxels and lack the ability to handle large-scale, boundaryless environments. But, since some solid works of dynamic maps and unbounded scenes have been done in pure NeRF reconstruction, this direction may require more efforts on finding better solutions to do that.

In terms of tracking, a significant question arises on how to use Bundle Adjustment for pose optimization when using voxel acceleration. Another challenge lies in performing pure NeRF-based SLAM without knowing the camera poses. Approaches such as iMAP, NiceSLAM, ESLAM, and co-slam seem to have ignored this problem and to assume camera poses are known.

Furthermore, few NeRF-based SLAM methods appear to consider loop closure and location, indicating potential areas for exploration and improvement.

3.2 Proposed work on NeRF itself.

The speech of Prof. Dahua Lin when sharing their OmniObject3D dataset inspired me that 3D model generation may be possible to be achieved by using a combination of diffusion and NeRF model. So, I list 3D diffusion with NeRF as one of the possible research directions.

While Point-NeRF [22] has already been developed and indicated that volume density when storing in point cloud can also has good performance, NeRF model transfer between maps are now possible. So, I add it to the possible research lists.

3.3 Base work (Original Proposal)

Define some tasks:

Table 2: Tasks on Base work		
Task	Content	Expect time
Preparation 1	Understanding source code of Plenoxels, Instant-ngp and DVGO. This needs to learn some CUDA programming or just use “tiny-cuda-nn”.	1 week
Preparation 2	Understanding how to test with commonly used datasets	1 day
Pre-experiment 1	Implementing Plenoxels + Hash encoding to find out how fast it can be	3 days
Pre-experiment 2	Implementing DVGO + Hash encoding to find out whether endpoint color MLP will significantly affect the training speed.	3 days
Main 1	Construct My Proposed SLAM system with all the compatible SOTA tricks Testing it	2 weeks
Main 2	Explore how to manage voxels in an Octree to achieve dynamic mapping Testing it	2 weeks
Experiment 1	Testing my Base with commonly used Dataset and compare it with other related work especially CO-SLAM [35].	2 weeks
Archive	Modify the code to make it as modular as possible for future use	1 week

PS: Known problems of this Research Proposal draft

1. **Need Milestones and Gantt**
2. **Citations are not in the form of their published venues.**
3. **The format of citations in text is not consistent.**
4. **Need a general summary of Literature review.**

4. Reference

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, ‘NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis’. arXiv, Aug. 03, 2020. Accessed: Jun. 10, 2023. [Online]. Available: http://arxiv.org/abs/2003.08934

[2] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, ‘FastNeRF: High-Fidelity Neural Rendering at 200FPS’, in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada: IEEE, Oct. 2021, pp. 14326–14335. doi: 10.1109/ICCV48922.2021.01408.

[3] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, ‘Neural Sparse Voxel Fields’. arXiv, Jan. 06, 2021. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2007.11571

[4] T. Neff et al., 'DONeRF: Towards Real‐Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks', Computer Graphics Forum, vol. 40, no. 4, pp. 45–59, Jul. 2021, doi: 10.1111/cgf.14340.

[5] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, ‘Baking Neural Radiance Fields for Real-Time View Synthesis’, in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada: IEEE, Oct. 2021, pp. 5855–5864. doi: 10.1109/ICCV48922.2021.00582.

[6] T. Müller, A. Evans, C. Schied, and A. Keller, ‘Instant neural graphics primitives with a multiresolution hash encoding’, ACM Trans. Graph., vol. 41, no. 4, pp. 1–15, Jul. 2022, doi: 10.1145/3528223.3530127.

[7] C. Reiser, S. Peng, Y. Liao, and A. Geiger, ‘KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs’, in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada: IEEE, Oct. 2021, pp. 14315–14325. doi: 10.1109/ICCV48922.2021.01407.

[8] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, ‘PlenOctrees for Real-time Rendering of Neural Radiance Fields’. arXiv, Aug. 17, 2021. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2103.14024

[9] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, ‘Plenoxels: Radiance Fields without Neural Networks’, in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA: IEEE, Jun. 2022, pp. 5491–5500. doi: 10.1109/CVPR52688.2022.00542.

[10] B. Deng, J. T. Barron, and P. P. Srinivasan, JaxNeRF: an efficient JAX implementation of NeRF. 2020.

[11] C. Sun, M. Sun, and H.-T. Chen, ‘Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction’, in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA: IEEE, Jun. 2022, pp. 5449–5459. doi: 10.1109/CVPR52688.2022.00538.

[12] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, ‘iMAP: Implicit Mapping and Positioning in Real-Time’, in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada: IEEE, Oct. 2021, pp. 6209–6218. doi: 10.1109/ICCV48922.2021.00617.

[13] A. Rosinol, J. J. Leonard, and L. Carlone, ‘NeRF-SLAM: Real-Time Dense Monocular SLAM with Neural Radiance Fields’. arXiv, Oct. 24, 2022. Accessed: Jun. 15, 2023. [Online]. Available: http://arxiv.org/abs/2210.13641

[14] C.-M. Chung et al., ‘Orbeez-SLAM: A Real-time Monocular Visual SLAM with ORB Features and NeRF-realized Mapping’. arXiv, Jan. 31, 2023. Accessed: Jun. 12, 2023. [Online]. Available: http://arxiv.org/abs/2209.13274

[15] Z. Zhu et al., ‘NICE-SLAM: Neural Implicit Scalable Encoding for SLAM’, in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA: IEEE, Jun. 2022, pp. 12776–12786. doi: 10.1109/CVPR52688.2022.01245.

[16] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, ‘Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields’, in 2022 IEEE/CVF Conference on Computer

Vision and Pattern Recognition (CVPR), New Orleans, LA, USA: IEEE, Jun. 2022, pp. 5460–5469. doi: 10.1109/CVPR52688.2022.00539.

[17] K. Zhang, G. Riegler, N. Snavely, and V. Koltun, 'NeRF++: Analyzing and Improving Neural Radiance Fields'. arXiv, Oct. 21, 2020. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2010.07492

[18] M. Tancik et al., 'Block-NeRF: Scalable Large Scene Neural View Synthesis'. arXiv, Feb. 10, 2022. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2202.05263

[19] H. Turki, D. Ramanan, and M. Satyanarayanan, 'Mega-NeRF: Scalable Construction of Large-Scale NeRFs for Virtual Fly- Throughs', in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA: IEEE, Jun. 2022, pp. 12912–12921. doi: 10.1109/CVPR52688.2022.01258.

[20] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, 'Neural Sparse Voxel Fields'. arXiv, Jan. 06, 2021. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2007.11571

[21] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, 'Vox-Fusion: Dense Tracking and Mapping with Voxel-based Neural Implicit Representation', in 2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Singapore, Singapore: IEEE, Oct. 2022, pp. 499–507. doi: 10.1109/ISMAR55827.2022.00066.

[22] Q. Xu et al., 'Point-NeRF: Point-based Neural Radiance Fields'. arXiv, Mar. 15, 2023. Accessed: Jun. 15, 2023. [Online]. Available: http://arxiv.org/abs/2201.08845

[23] Z. Li, L. Li, Z. Ma, P. Zhang, J. Chen, and J. Zhu, 'READ: Large-Scale Neural Scene Rendering for Autonomous Driving'. arXiv, May 11, 2022. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2205.05509

[24] Z. Zhu et al., 'NICER-SLAM: Neural Implicit Scene Encoding for RGB SLAM'. arXiv, Feb. 07, 2023. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2302.03594

[25] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, ‘BARF: Bundle-Adjusting Neural Radiance Fields’. arXiv, Aug. 19, 2021. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2104.06405

[26] R. Clark, ‘Volumetric Bundle Adjustment for Online Photorealistic Scene Capture’, in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA: IEEE, Jun. 2022, pp. 6114–6122. doi: 10.1109/CVPR52688.2022.00603.

[27] Y. Yuan and A. Nuchter, ‘An Algorithm for the SE(3)-Transformation on Neural Implicit Maps for Remapping Functions’, IEEE Robot. Autom. Lett., vol. 7, no. 3, pp. 7763–7770, Jul. 2022, doi: 10.1109/LRA.2022.3185383.

[28] A. Karnewar, T. Ritschel, O. Wang, and N. Mitra, ‘ReLU Fields: The Little Non-linearity That Could’, in Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings, Vancouver BC Canada: ACM, Aug. 2022, pp. 1–9. doi: 10.1145/3528233.3530707.

[29] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, ‘Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields’, in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada: IEEE, Oct. 2021, pp. 5835–5844. doi: 10.1109/ICCV48922.2021.00580.

[30] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, ‘Depth-supervised NeRF: Fewer Views and Faster Training for Free’. arXiv, Apr. 29, 2022. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2107.02791

[31] C. Reiser et al., ‘MERF: Memory-Efficient Radiance Fields for Real-time View Synthesis in Unbounded Scenes’. arXiv, Feb. 23, 2023. Accessed: Jun. 20, 2023. [Online]. Available: http://arxiv.org/abs/2302.12249

[32] M. M. Johari, C. Carta, and F. Fleuret, ‘ESLAM: Efficient Dense SLAM System Based on Hybrid Representation of Signed Distance Fields’. arXiv, Apr. 03, 2023. Accessed: Jun. 20, 2023. [Online]. Available: http://arxiv.org/abs/2211.11704

[33] X. Kong, S. Liu, M. Taher, and A. J. Davison, ‘vMAP: Vectorised Object Mapping for Neural Field SLAM’. arXiv, Mar. 13, 2023. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2302.01838

[34] C. Sun, M. Sun, and H.-T. Chen, ‘Improved Direct Voxel Grid Optimization for Radiance Fields Reconstruction’. arXiv, Jul. 01, 2022. Accessed: Jun. 22, 2023. [Online]. Available: http://arxiv.org/abs/2206.05085

[35] H. Wang, J. Wang, and L. Agapito, ‘Co-SLAM: Joint Coordinate and Sparse Parametric Encodings for Neural Real-Time SLAM’.