SW-NeRF: Scale-Aware Neural Radiance Fields for Metric-Accurate Scene Understanding

Haiwen Xia* **Yuxin Shou*** **Hang Dai***
School of EECS
Peking University
https://github.com/daihangpku/SW-NeRF

Abstract

Neural Radiance Fields (NeRF) have revolutionized 3D scene reconstruction, yet existing methods often lack absolute scale information and comprehensive dynamic scene modeling capabilities. However, Scale perception is crucial for robotic tasks such as manipulation, navigation, and scene understanding. Thus, we present SW-NeRF (Scale-Aware NeRF), a unified framework that bridges this gap, enabling robots to perceive and reconstruct scenes with metric accuracy. Our framework makes three key contributions: (1) A scale recovery mechanism using ArUco markers for accurate 3D reconstructions; (2) A unified implementation combining NeRF for static scenes, D-NeRF for dynamic deformations, and T-NeRF for time-dependent changes, suitable for various robotic applications; (3)Multi-Resolution NeRF, an extension of D-NeRF to learn the decomposition of a image; (4)An auxiliary loss function to D-NeRF; (5) Efficient performance with minimal computational overhead. Extensive experiments show that SW-NeRF achieves precise metric measurements while preserving the high-fidelity reconstruction quality of the original NeRF methods.

1 Introduction

Scale perception plays a fundamental role in robotic applications, particularly in tasks requiring precise interaction with the physical world, such as grasping an object or navigating through space. While recent advances in Neural Radiance Fields (NeRF) have demonstrated remarkable capabilities in high-fidelity 3D scene reconstruction, they typically operate in arbitrary units, lacking the essential metric scale needed for real-world interaction.

Traditional approaches to 3D reconstruction in robotics often rely on depth sensors or stereo vision systems, which can provide metric information but may struggle with scene completeness and view consistency. While NeRF addresses these limitations by learning a continuous volumetric scene representation, it sacrifices absolute scale information in favor of relative spatial relationships. This limitation becomes particularly problematic in robotic applications, where the difference between a 10cm and a 15cm object can determine the success or failure of a manipulation task.

Several attempts have been made to incorporate scale information into neural scene representations. Some methods use depth sensors as auxiliary inputs, while others rely on multi-view geometry constraints. However, these approaches either require additional specialized hardware or introduce computational complexity that may not be suitable for real-time robotic applications.

To address these challenges, we present SW-NeRF (Scale-Aware Neural Radiance Fields), a unified framework that enables metric-accurate 3D reconstruction while preserving NeRF's advantages. Our key insight is to leverage ArUco markers, widely used in robotics for pose estimation, as reliable scale references during scene reconstruction.

*equally contributed

The main contributions of this paper can be summarized as follows:

- We propose a novel scale-aware NeRF framework using ArUco markers that recovers absolute scale for precise 3D reconstructions in robotics applications.
- We develop a unified implementation that integrates NeRF, D-NeRF, and T-NeRF for static, dynamic, and time-varying scenes while maintaining scale consistency.
- We extend D-NeRF by Multi-Resolution NeRF to explore different representations and TV loss to enhance its performance.
- We present a comprehensive evaluation in real-world static and dynamic scenes, demonstrating its effectiveness in tasks requiring accurate scale perception.

2 Related Work

Our approach lies at the intersection of multiple fields. In the following, we review related work.

ArUco Markers. In recent years, ArUco markers have gained significant attention in computer vision tasks, especially for pose estimation and scene reconstruction. This is due to their distinct visual patterns, making them easily detectable even in challenging environments. Past research has focused on improving marker robustness in occluded conditions. S. Garrido-Jurado *et al.* have made significant contributions to this problem. Their work introduces an approach for generating ArUco markers that maintain high reliability even when parts of the markers are occluded (1). This approach is particularly relevant to our work, as we aim to use ArUco markers for scale recovery during 3D scene reconstruction. Unlike traditional methods that focus solely on pose estimation, our framework leverages the metric accuracy of ArUco markers to ensure precise 3D reconstruction in both static and dynamic environments.

Robotics Applications in 3D Reconstruction. Robotic systems increasingly rely on accurate 3D reconstructions for tasks such as navigation, manipulation, and scene understanding. Traditional approaches often use LiDAR or stereo cameras to obtain metric information (2). However, these methods face limitations in terms of scene completeness. Neural representations like NeRF offer improved scene fidelity but lack metric accuracy. SW-NeRF bridges this gap, enabling robots to leverage the strengths of NeRF while obtaining reliable scale information for real-world interaction.

3 Methods

In NeRF literature, we represent a continuous scene as a 5D vector-valued function whose input is a 3D location $\mathbf{x} = (x, y, z)$ and 2D viewing direction (θ, ϕ), and whose output is an emitted color $\mathbf{c} = (r, g, b)$ and volume density σ. To effectively capture high-frequency details in the scene, we employ positional encoding γ that lifts the input coordinates into a higher dimensional space before processing them through the network:

$$F_\Theta : \gamma(\mathbf{x}, \theta, \phi) \rightarrow (\mathbf{c}, \sigma)$$

Base on this classical NeRF formulation, our framework utilizes ArUco markers as scale references and modify the neural representation to incorporate temporal information and deformation fields. The resulting architecture maintains the advantages of neural radiance fields while providing metric accuracy crucial for real-world applications.

3.1 Positional Encoding

In the standard NeRF model, a single 2D image is used to train the neural network to model the underlying 3D scene. To map spatial coordinates to a higher-dimensional space, we employ positional encoding as the input of the neural network.

Given a 3D input point (x, y, z), the positional encoding function is defined as:

$$\gamma(p) = \big(\sin(2^0\pi p), \cos(2^0\pi p), \sin(2^1\pi p), \cos(2^1\pi p), \ldots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)\big)$$

where p represents the spatial coordinate, and L is the number of frequency bands.

For fitting a single image, we use a coordinate-based MLP. The input is the 2D pixel coordinate (x, y), and the output is the corresponding RGB color. This forms a supervised learning problem, which we can train using gradient descent.

3.2 Volume Rendering

To render the 3D scene from any given viewpoint, we employ volume rendering techniques to accumulate color and density values along each camera ray. For a camera ray **r**(t) = **o** + t**d** originating from camera center **o** with direction **d**, we calculate the expected color **C**(**r**) using:

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

In practice, we approximate this continuous integral using numerical quadrature with a stratified sampling approach. We partition the ray into N evenly-spaced bins and sample one point per bin:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))\mathbf{c}_i$$

where $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j\delta_j)$ is the discrete accumulated transmittance, δ_i is the distance between adjacent samples, and σ_i and $\mathbf{c}_i$ are the density and color predicted by the network at sample point i. To improve rendering quality while maintaining computational efficiency, we employ a hierarchical sampling strategy. We first sample N_c points using a coarse network to identify regions of high density, then sample an additional N_f points concentrated in these regions using a fine network. This hierarchical approach allows us to allocate more samples to regions that contribute most significantly to the final rendered image.

3.3 Modeling dynamic scenes

We propose two main methods for modeling dynamic scenes:

- **T-DeRF:** The dynamic scene is represented by a 6D input consisting of the query 3D location, viewing direction, and time, denoted as $(x, y, z, t, \theta, \phi)$, where (x, y, z) is the 3D point, (θ, ϕ) is the viewing direction, and t is the time variable.
- **Canonical Network and Deformation Network:** We model the dynamic scene using two neural network modules, Ψ_t and Ψ_x. The Canonical Network, an MLP $\Psi_x(x, d) \rightarrow (c, \sigma)$, is trained to encode the scene in its canonical form. Given a 3D point x and a view direction d, it outputs the emitted color c and volume density σ. The Deformation Network, consisting of another MLP $\Psi_t(x, t) \rightarrow \Delta x$, predicts a deformation field that defines the transformation between the scene at time t and the scene in its canonical configuration.

3.4 Multi-Resolution NeRF

Multi-Resolution NeRF predicts the final image by combining the outputs of multiple networks. Specifically, the final output image is the sum of the images rendered by the networks $\Phi_1, \Phi_2, \ldots, \Phi_t$. This approach attempts to explicitly make the network learn a decomposition rather than position encoding.

In Multi-Resolution NeRF, each network Φ_i generates an image with dimensions $\left(\frac{H}{2^{i-1}}, \frac{W}{2^{i-1}}\right)$. The camera parameters for Φ_i are slightly modified from the original, with the focal length adjusted to $\frac{\text{focal}}{2^{i-1}}$. Given the same view directions, the networks $\Phi_1, \ldots, \Phi_n$ collaboratively produce a pyramid of images. We utilize the same combination technique employed in reconstructing an image from its Laplacian pyramid, as described below:

1. **Initialize**: Start with the topmost layer of the Laplacian Pyramid, L_{N-1}.
2. **Iterative Reconstruction**:
   - For each layer i from $N - 2$ down to 0:
     (a) **Upsample** the current reconstruction R_{i+1} to match the dimensions of L_i.
     (b) **Add** the Laplacian layer L_i to the upsampled image: R_i = Upsample(R_{i+1}) + L_i

3. **Finalize**: The reconstructed image R_0 is obtained after processing all layers.

To make these networks converge faster and learn different information, the training process first minimizes the mean square error between L_i and the output of Φ_i, training the models respectively. After several iterations, the loss becomes the MSE between Ground Truth and the combined image.

The Total Variation (TV) loss is used to enforce temporal consistency in image reconstruction tasks.It penaltizes the abrupt changes in the deformation network in D-NeRF. In each iteration, when reconstructing images in time t,TV loss samples the adjacent timestamp t' from $\mathcal{U}(t_1, t_2)$, then calculate the deformation $\Delta_{t,sample}$ and $\Delta_{t',sample}$ in each sample. The final TV loss is represented as:

$$\mathcal{L}_{\text{TV}} = \lambda_{\text{TV}} \sum_k \left(\Delta_{t,k} - \Delta_{\text{t}',k}\right)^2$$

where k stands for the number of samples.

3.5 Mesh Extraction

We present a method for extracting explicit surface meshes from trained NeRF models through density field sampling and marching cubes. Our approach addresses both the geometric and appearance aspects of the reconstruction process. The core of our method involves sampling the neural radiance field across multiple viewpoints to obtain a robust density estimation. Specifically, we generate a set of uniformly distributed viewing directions on a unit sphere using the golden spiral algorithm. This multi-view sampling strategy helps mitigate potential view-dependent artifacts in the density field. For spatial sampling, we discretize the bounded volume into a regular grid with resolution Rş. At each grid point, we perform N view-directional queries through the NeRF network to obtain density and color values. These queries are processed in batches to optimize computational efficiency. Once the density field is obtained, we employ the marching cubes algorithm with a density threshold to extract the isosurface.The vertex positions $\mathbf{v}_j$ and faces $\mathbf{f}_k$ are generated by:

$$\mathcal{M} = \text{MarchingCubes}(\{\sigma(\mathbf{x}) | \mathbf{x} \in \text{Grid}(R)\}, \tau)$$

The color of each vertex in the resulting mesh is determined by interpolating from the nearest grid points in our sampled color field. This approach preserves the appearance information captured by the NeRF model while providing an explicit geometric representation. Our method provides controllable parameters - the grid resolution R and density threshold - that allow for trade-offs between reconstruction accuracy and computational cost. The resulting textured mesh enables direct compatibility with traditional graphics pipelines while maintaining the high-quality appearance modeling characteristic of NeRF representations.

3.6 Scale Calibration and Transformation

To address the scale ambiguity in NeRF reconstruction and align the mesh with real-world coordinates, we propose a calibration method using ArUco markers. The process consists of three main steps: First, we detect ArUco markers across multiple captured images and extract their corner positions in 2D image space. For each corner point, we compute the ray direction $\mathbf{d}_i$ from the camera center through the image point after accounting for lens distortion:

$$\begin{aligned}
\mathbf{p}_n &= [(x - c_x)/f_x, (y - c_y)/f_y]^\top \\
\mathbf{p}_u &= \text{Undistort}(\mathbf{p}_n, k_1, k_2, p_1, p_2) \\
\mathbf{d}_i &= \mathbf{R}[\mathbf{p}_u^\top, 1]^\top
\end{aligned}$$

where (f_x, f_y) are focal lengths, (c_x, c_y) is the principal point, and (k_1, k_2, p_1, p_2) are distortion coefficients. Second, we triangulate the 3D position of each marker corner by minimizing the distance between rays from different views:

$$\mathbf{X}_j = \underset{\mathbf{X}}{\text{argmin}} \sum_i d(\mathbf{X}, \mathbf{C}_i, \mathbf{d}_i)$$

where $d(\mathbf{X}, \mathbf{C}_i, \mathbf{d}_i)$ computes the distance from point $\mathbf{X}$ to ray originating from camera center $\mathbf{C}_i$ in direction $\mathbf{d}_i$. Finally, given the known physical size of the ArUco marker s, we compute the scale factor α as:

$$\alpha = \frac{s}{\frac{1}{4}\sum_{i=1}^{4} \|\mathbf{X}_i - \mathbf{X}_{i+1}\|}$$

Additionally, we calculate a transformation matrix $\mathbf{T}$ that aligns the marker's normal with the z-axis. The final mesh vertices are transformed as:

$$\mathbf{v}_i' = \alpha \mathbf{T} \mathbf{v}_i$$

This calibration approach enables accurate scale recovery and alignment of the reconstructed mesh with real-world coordinates, which is crucial for applications requiring metric measurements.

4 Experiments

4.1 Dataset

Our experiments were conducted on two categories of datasets: (1) standard benchmark datasets commonly used in NeRF literature, and (2) custom datasets captured and processed using COLMAP and Segment Anything. For benchmark evaluation, we utilized the synthetic NeRF dataset, which includes the widely-used Lego and BouncingBalls scenes along with other synthetic objects. These datasets provide relatively precisely calibrated camera poses and ground truth depth maps, making them ideal for quantitative evaluation. For real-world validation, we collected our own dataset using a handheld camera, processing the images through COLMAP to obtain camera poses and sparse reconstructions. This dataset is object-centered with ArUco markers, enabling evaluation of our scale recovery mechanism.

4.2 Training

Due to NeRF's inherent limitations in generalization and robustness, we implemented different training strategies for synthetic and custom datasets. All experiments were conducted on an NVIDIA RTX 4060 Ti GPU. For synthetic datasets like the Lego scene, we used a standard configuration with hyperparameters optimized for known camera poses:

- **Sampling Strategy:** $N_{\text{samples}} = 64$ coarse samples, $N_{\text{importance}} = 128$ fine samples
- **Progressive Training:** `precrop_iters`=500, `precrop_frac`=0.5
- **View Parameters:** `use_viewdirs`=True, `white_bkgd`=True
- **Learning Rate:** `lrate`=5e-4

For custom datasets processed through COLMAP, we adopted a more flexible approach to address the challenges of imperfect camera pose estimation. We selectively filtered out images with unreliable COLMAP pose estimates and dynamically adjusted training parameters based on pose confidence metrics. Key modifications included varying sampling densities, adaptive learning rates, and progressive view incorporation. This adaptive strategy proved crucial for achieving stable convergence while maintaining high reconstruction quality despite the inherent uncertainties in COLMAP-estimated poses.

4.3 Results

4.3.1 2D Positional Encoding

We experimented with different values of L (dimension of positional encoding) and the number of layers. The results show that increasing L and network depth consistently reduces MSE and improves PSNR. The optimal performance was achieved with $L = 10$ and 10 layers, obtaining an MSE of 0.0012 and a PSNR of 29.5673. The visualizations(Table 1, Table 2, Figure 1) below further illustrate that increasing L allows the model to represent more complex image details, as seen in the improvement in visual quality. This shows the importance of higher positional encoding dimensions and deeper architectures in image reconstruction.

4.3.2 Vanilla NeRF

We first evaluated our implementation of Vanilla NeRF on the standard synthetic datasets to establish baseline performance. The experiments were conducted on both the synthetic Lego dataset and our custom COLMAP-processed real-world scenes. Visualizations are shown in Figure 2).
On the synthetic `Lego` test dataset, our implementation achieved:

L / Layer Num	6	8	10
0	18.8391	19.1501	19.3000
6	26.5685	27.4766	27.9530
8	27.9611	28.9512	29.4493
10	28.0572	29.0213	**29.5673**

Table 1: PSNR of Different L and Number of Layers

L / Layer Num	6	8	10
0	0.0132	0.0123	0.0119
6	0.0023	0.0019	0.0017
8	0.0018	0.0014	0.0013
10	0.0017	0.0014	**0.0012**

Table 2: MSE Loss of Different L and Number of Layers

Baseline Best Configuration

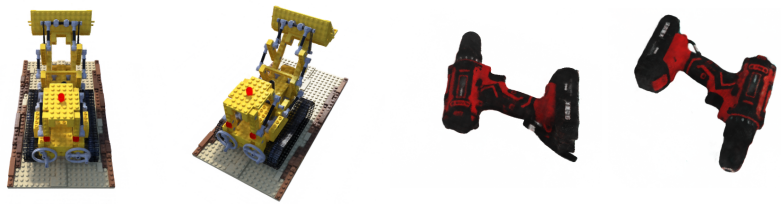
Figure 1: Comparison of reconstructions for different configurations. Left: Baseline configuration ($L = 0$, layer num=10). Right: Best-performing configuration ($L = 10$, layer num=10).

PSNR: 31.0680 dB **SSIM**: 0.9617 **LPIPS**: 0.0194

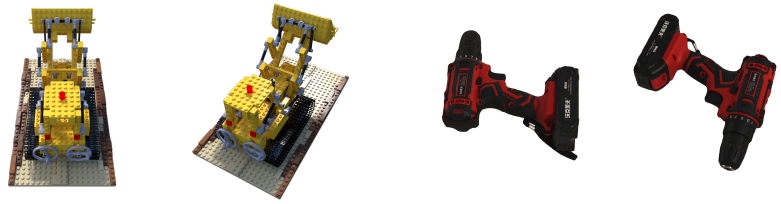
These results are comparable to those reported in the original NeRF paper, validating our implementation. For novel view synthesis, our model demonstrated strong performance in capturing fine geometric details and complex view-dependent effects, particularly evident in the intricate patterns of the Lego model. For custom scenes `Drill` processed through COLMAP, we observed still competitive metrics:

PSNR: 32.6213 dB **SSIM**: 0.9757 **LPIPS**: 0.0159

The slight degradation in performance can be attributed to the inherent noise in COLMAP-estimated camera poses and the complexity of real-world lighting conditions. Nevertheless, the model successfully reconstructed complex scenes with faithful geometry and appearance across novel viewpoints.



Estimated Images (NeRF Reconstruction)

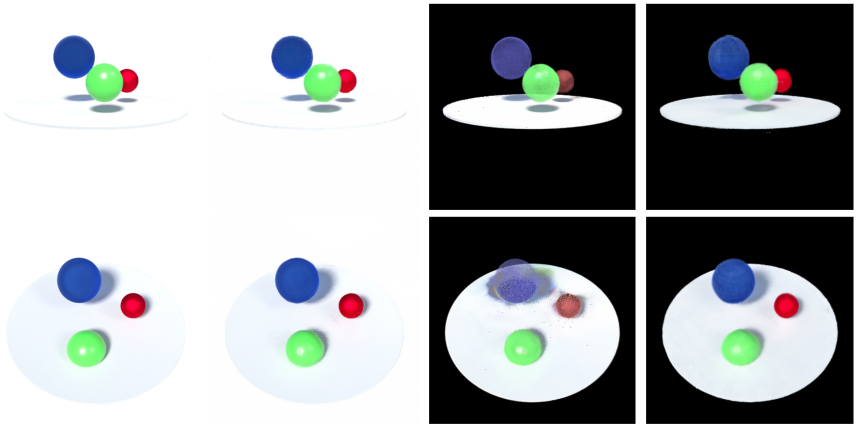


Ground Truth

Figure 2: Comparison of NeRF reconstructions with ground truth images on the `lego` dataset.

4.3.3 T-NeRF and D-NeRF

We evaluated the performance of T-NeRF and D-NeRF on the `BouncingBalls` dataset. The model was trained on the dataset and tested on the same sequence to assess its ability to reconstruct dynamic and deformable objects accurately. As shown in Figure 3 and Table 3, D-NeRF consistently outperforms T-NeRF in capturing the dynamic motion and deformation of objects within the scene. Among the T-NeRF variations, T-NeRF without viewing direction exhibits better performance compared to T-NeRF with viewing direction. Specifically, D-NeRF demonstrates the best performance across all metrics, achieving the highest PSNR (36.02) and SSIM (0.984), alongside a low MSE (0.00025) and LPIPS (0.115). T-NeRF without viewing direction improves upon T-NeRF with viewing direction in all metrics, indicating better structural fidelity.



Ground Truth D-NeRF T-NeRF (with view) T-NeRF (no view)

Figure 3: Comparison of reconstruction results from Ground Truth, D-NeRF, T-NeRF with viewing direction, and T-NeRF without viewing direction on two frames.

Method	MSE	PSNR	SSIM	LPIPS
D-NeRF	0.00025	36.02	0.984	0.115
T-NeRF (with view)	0.0036	24.91	0.936	0.122
T-NeRF (no view)	0.0019	27.58	0.967	0.115

Table 3: Comparison of D-NeRF, T-NeRF with viewing direction, and T-NeRF without viewing direction across metrics.

4.3.4 D-NeRF Extensions

We evaluated the performance of D-NeRF with TV loss on the `BouncingBalls` dataset and Multi-Resolution NeRF on `Mutant`. In comparison with original D-NeRF, TV loss allows the model to generate a smoother video when rendering a fixed camera pose and changing the time. 35000 iterations of TV-D-NeRF has similar results as 100000 iterations of original D-NeRF. On the other hand, Multi-Resolution NeRF performs badly. Despite the lack of implementation time and training time, it mainly suffer from the fact that Laplacians are mainly zero, shown in Figure 4. The current PSNR of Multi-Res is 11.01.

4.3.5 Mesh Extract and Transform

Our mesh extraction and transformation pipeline successfully converts the implicit neural radiance fields into explicit textured meshes while maintaining proper scale and alignment. For mesh extraction, we sample the NeRF model with a resolution of 128ş grid points and employ marching cubes with a density threshold = 30. Using the ArUco marker-based calibration method shown in Figure 5, we achieve accurate scale recovery with an average edge length error of less than 5% . If the

Figure 4: the first layer of the laplacian of mutant

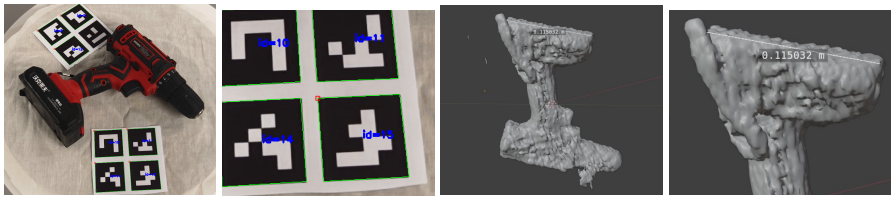


Figure 5: ArUco marker detection results and Mesh extraction and transformation results.

dataset is better calibrated, the result is bound to be more accurate. As shown in Figure 5, The drill's base is measured 0.1150 m in Blender and its physical length is about 12 cm, which means that the transformed mesh aligns well with real-world measurements, enabling direct integration with existing 3D modeling pipelines and physical prototyping applications.

5 Conclusion

This paper presents SW-NeRF, a scale-aware neural radiance field framework that successfully bridges the gap between high-fidelity neural scene reconstruction and metric-accurate applications. Through extensive experiments, we have demonstrated that our approach effectively combines the advantages of NeRF-based reconstruction with absolute scale information, achieving both visual quality and metric accuracy. Morover our implementation unifies multiple NeRF variants (classical NeRF, D-NeRF, and T-NeRF) and establishes a pipline to maintain consistent physical scale information with the help of aruco using Vanilla NeRF. However, several limitations remain to be addressed. First, the accuracy of our approach on custom datasets is constrained by the quality of COLMAP camera pose estimation, which can introduce errors in both reconstruction and scale recovery. Second, the implicit nature of neural radiance fields presents challenges in terms of robustness and interpretability, particularly in scenarios requiring guaranteed behavior for safety-critical robotic applications. Looking forward, several promising directions for future work emerge:
Real-time Performance: Investigating neural architecture optimization and acceleration techniques to enable real-time inference for dynamic robotic environments.
Multi-sensor Integration: Extending the framework to incorporate additional sensor modalities such as LiDAR or depth cameras to enhance reconstruction accuracy and scale estimation.
We believe that addressing these challenges while building upon the strengths of our current approach will lead to more practical and reliable neural scene reconstruction systems for robotics applications.

References

[1] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[2] P. Li, X. Chen, and S. Shen, "Stereo R-CNN Based 3D Object Detection for Autonomous Driving," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 7636–7644, doi: 10.1109/CVPR.2019.00783.