Depth-Enhanced Neural Radiance Fields for Autonomous Driving Scenes with LiDAR-Grounded Supervision

Dharm Mehta
Northeastern University
Boston, MA, USA
mehta.dhar@northeastern.edu

Manivannan Senthil Kumar
Northeastern University
Boston, MA, USA
senthilkumar.m@northeastern.edu

Jaimin Sorathiya
Northeastern University
Boston, MA, USA
sorathiya.j@northeastern.edu

Ket Bhikadiya
Northeastern University
Boston, MA, USA
bhikadiya.k@northeastern.edu

Abstract

This report enhances Neural Radiance Fields (NeRF) for large-scale autonomous driving environments using multimodal data from the NuScenes dataset. We integrate LiDAR-derived depth constraints as geometric priors into the NeRF training pipeline through a dedicated depth loss term. The model employs sinusoidal positional encoding (10 bands for 3D coordinates, 4 for viewing direction) and a comprehensive loss function combining photometric reconstruction, depth supervision, and distortion regularization, optimized via Cosine Annealing. Experiments on a single NuScenes scene achieved PSNR and SSIM scores of 13.52 and 0.59 respectively, revealing significant challenges in applying standard NeRF to unbounded outdoor scenarios with dynamic elements. Our findings highlight the need for careful analysis of design choices and hyperparameter sensitivity to improve robustness in autonomous driving scene reconstruction.

1. Introduction

The accurate perception and reconstruction of the environment are fundamental challenges in the development of autonomous driving systems. Traditional methods often rely on sensor fusion techniques, but capturing the intricate, view-dependent appearance and fine geometric detail of real-world scenes remains computationally intensive and complex. Neural Radiance Fields (NeRF) have emerged as a powerful paradigm for novel view synthesis, representing a scene as a continuous function that maps 3D location ($\mathbf{x}$) and viewing direction ($\mathbf{d}$) to color (RGB) and volume density (σ).

However, applying the original NeRF framework to unbounded, dynamic, and large-scale outdoor environments like those encountered in autonomous driving presents significant difficulties. These outdoor scenes exhibit a vast range of depths and contain moving objects (e.g., vehicles and pedestrians), which standard NeRF struggles to model effectively. To address these issues, our project focuses on enhancing the representation capability of NeRF by incorporating explicit geometric constraints from high-fidelity sensor measurements.

1.1. Project Focus

Our work is based on an autonomous driving dataset that provides synchronized multimodal data, including high-resolution camera images and accurate range measurements from LiDAR. This integration is crucial for addressing the limitations of NeRF in large outdoor scenes.

The core contribution of this work is the integration of LiDAR-derived depth as a geometric prior to stabilize and constrain the optimization of the neural network. Specifically, we utilize the projected LiDAR measurements as ground truth depth to enforce a depth loss term during training, penalizing the model when its expected depth differs from the measured depth (ground truth). This depth-enhanced approach is designed to produce a more geometrically accurate radiance field, which is essential for reliable 3D scene reconstruction in autonomous vehicle contexts.

2. Dataset and Preprocessing

For this project, we utilized the NuScenes dataset, a prominent large-scale public resource specifically designed for research on autonomous driving. NuScenes is well-suited for our depth-enhanced NeRF approach due to its comprehen-

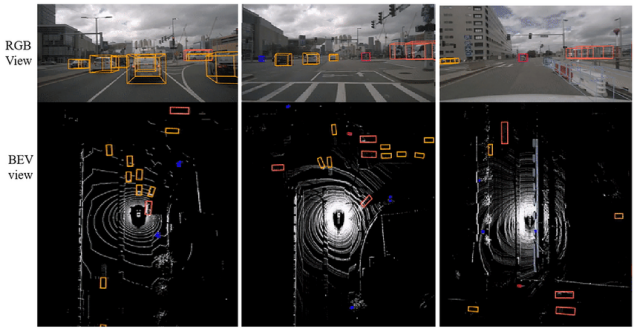

Figure 1. A real sample from the nuScenes autonomous driving dataset. Outdoor environments introduce challenges such as large spatial extent, moving vehicles, illumination changes, and sparse LiDAR depth — motivating the need for depth-supervised NeRF.

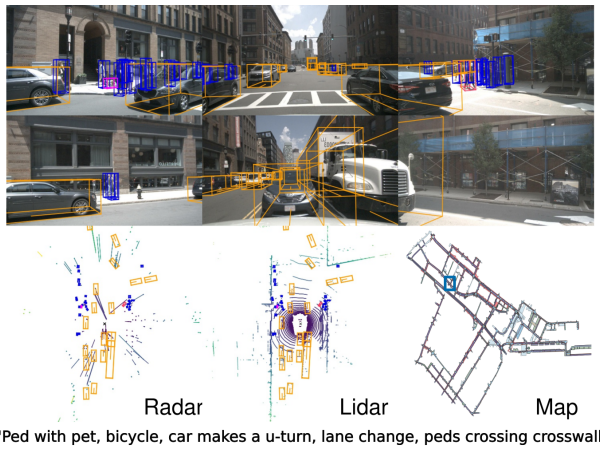

Figure 2. Example input data used in our experiments. We extract three forward-facing camera views (Front, Front-Left, Front-Right) from each scene and project LiDAR points into the image plane to obtain depth supervision. This setup reflects our chosen subset of 15 scenes and 180 frames, enabling a focused analysis of depth-supervised NeRF performance.

sive multimodal sensor configuration and the diversity of its capture environments.

2.1. Dataset Characteristics

The NuScenes dataset provides a rich collection of driving scenarios, encompassing 1,000 scenes, each approximately 20 seconds long. The data were captured in the distinct urban environments of Boston and Singapore, providing high variability in urban geometry, weather conditions, and traffic density.

Crucially, the data are synchronized across a full sensor suite, which is essential for deriving the geometric priors required by our method. This sensor setup includes the following:

- **LiDAR:** One 32-beam LiDAR unit operating at 20Hz, providing highly accurate, sparse 3D point cloud measurements. This is the primary source for our depth supervision.
- **Cameras:** Three cameras offering a 180° view for visual context and RGB input.
- **RADAR:** Five long-range RADAR units.

Our experimental work focused specifically on the NuScenes Mini subset to facilitate efficient prototyping and training.

2.2. Scene Selection

To study the effect of depth supervision on NeRF reconstruction quality, we manually selected a subset of 15 scenes from nuScenes-mini with diverse layouts, including intersections, straight roads, and turns. Restricting the dataset allows us to examine failure cases, training dynamics, and depth-alignment issues in greater depth rather than prioritizing large-scale coverage.

2.3. Data Processing Pipeline

To convert the raw NuScenes data into a format suitable for the NeRF training framework, a dedicated preprocessing pipeline was implemented, focusing on generating clean, accurate depth ground truth. The primary stages of this pipeline were:

1. **Data Input and Selection:** Selection of specific keyframes and corresponding sensor data from the NuScenes Mini subset.
2. **Geometric Transformation:** Application of intrinsic and extrinsic calibration parameters to align all sensor measurements within a unified coordinate system (e.g., world or ego-vehicle frame).
3. **Depth Ground Truth Generation:** This critical step involves three sub-components to create the necessary depth constraints:
   - *Projection:* LiDAR point clouds are accurately projected onto the 2D image plane of the selected camera views.
   - *Filtering:* Outlier points and measurements falling outside a predefined range (near and far bounds) are filtered out to ensure the robustness of the depth prior.
   - *Dynamic Object Masking:* To mitigate the disruptive effect of moving objects on the static scene representation assumption of NeRF, points belonging to dynamic entities (e.g., moving vehicles and pedestrians) are identified and masked from the depth ground truth map.
4. **Final Training Dataset Generation:** The processed RGB images, alongside their corresponding sparse, filtered, and masked depth ground truth maps, are packaged for input to the NeRF training procedure.

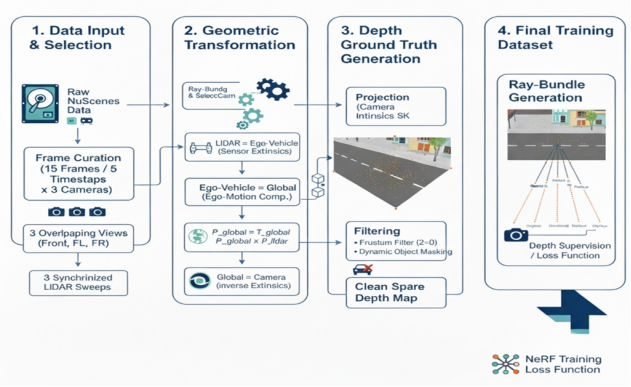

Figure 3. Overview of the data processing pipeline for generating depth-supervised training data from the NuScenes dataset. The pipeline transforms raw multimodal sensor readings into aligned RGB images and corresponding sparse, clean depth maps suitable for Neural Radiance Field training.

3. Model Architecture

The model architecture is designed to reconstruct a continuous 3D radiance field in outdoor driving environments under the extremely sparse sensing conditions characteristic of the nuScenes dataset [1]. Each timestamp provides observations from only three front-facing RGB cameras with narrow baselines and a single LiDAR sweep of limited spatial density.

To operate reliably under these constraints, the scene is represented as a hierarchical neural radiance field consisting of two multilayer perceptrons—a coarse network and a fine network—trained jointly through differentiable volumetric rendering [2]. The architecture incorporates stratified sampling with perturbation to ensure broad depth coverage, hierarchical importance sampling to concentrate evaluations near probable surfaces [3, 4], and a LiDAR-aware sampling strategy that injects explicit geometric priors from the available LiDAR sweep into the ray depth distribution [9]. Together, these components adapt classical NeRF methodology to sparse-sensor autonomous-driving data and enable stable optimization, improved depth reasoning, and significantly enhanced reconstruction fidelity in settings where traditional NeRF pipelines typically fail.

3.1. Radiance Field Parameterization

The scene is modeled as a continuous volumetric function $F_\Theta : (\mathbf{x}, \mathbf{d}) \to (\sigma, \mathbf{c})$, where $\mathbf{x} \in \mathbb{R}^3$ denotes a 3D location, $\mathbf{d} \in \mathbb{S}^2$ a viewing direction, $\sigma \in \mathbb{R}^+$ the predicted volume density, and $\mathbf{c} \in [0, 1]^3$ the emitted color. This formulation follows the Neural Radiance Field representation, which encodes geometry through σ and view-dependent appearance through $\mathbf{c}$ using differentiable volume rendering [2]. To enable high-frequency function approximation and to reduce

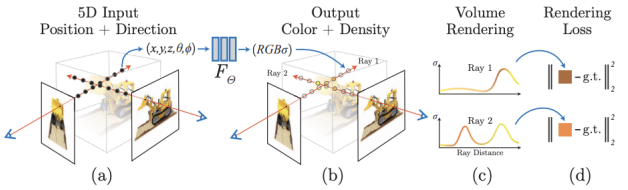

Figure 4. Overview of the NeRF rendering pipeline. **(a)** A camera ray is sampled at multiple 3D locations, each represented using its 5D input (position + direction). **(b)** The radiance field F_Θ predicts view-dependent color and volumetric density at each point. **(c)** Densities are accumulated along the ray using the differentiable volume rendering equation. **(d)** The rendered color is supervised with an ℓ_2 photometric loss.

aliasing across sampling scales, Fourier-style positional encodings are applied to both spatial coordinates and viewing directions, consistent with multiscale analyses introduced in Mip-NeRF [3].

The radiance field is instantiated as a pair of multilayer perceptrons—a coarse network and a fine network—that share the same functional mapping but operate on different point samples along each ray. The coarse network processes stratified and perturbed samples to produce an approximate density distribution, which is then used to construct a probability field that guides hierarchical importance sampling for the fine network. This refinement stage sharpens surface localization and recovers higher-frequency detail through more targeted evaluation of the radiance field, a strategy shown to be effective in sparse-view reconstruction [2, 4].

In outdoor driving scenarios where camera baselines are narrow and depth ambiguity is severe, regularizing the radiance field toward geometrically plausible solutions is essential for stable convergence [4]. Street-view radiance-field parameterizations that incorporate LiDAR cues and scene-specific priors further improve robustness under limited parallax; S-NeRF is a representative example that enhances pose handling and geometry supervision by leveraging sparse LiDAR measurements [8].

3.2. Positional Encoding

Neural networks that operate directly on raw 3D coordinates tend to learn overly smooth functions and fail to capture high-frequency geometric or appearance variations. To address this limitation, the radiance field employs Fourier-style positional encodings, first introduced in NeRF [2] and later extended to multiscale anti-aliased formulations in Mip-NeRF [3]. These encodings lift input coordinates into a higher-dimensional space whose sinusoidal basis functions expose a broad range of spatial and angular frequencies to the MLP.

Given a scalar coordinate $u \in \mathbb{R}$ and L frequency bands,

the encoding is defined as

$$\gamma(u) = \left[u,\, \sin(2^0\pi u),\, \cos(2^0\pi u),\, \ldots,\, \cos(2^{L-1}\pi u)\right] \tag{1}$$

which enables the network to approximate sharp transitions and high-frequency detail. For a 3D location $\mathbf{x} = (x, y, z)^\top$, the full positional encoding concatenates the encoded components:

$$\gamma_x(\mathbf{x}) = \left[\gamma(x),\, \gamma(y),\, \gamma(z)\right] \tag{2}$$

and the same construction is applied to the normalized viewing direction $\mathbf{d} = (d_x, d_y, d_z)^\top \in \mathbb{S}^2$:

$$\gamma_d(\mathbf{d}) = \left[\gamma(d_x),\, \gamma(d_y),\, \gamma(d_z)\right] \tag{3}$$

These encodings provide the MLP with a richer representational basis that supports the modeling of fine-scale geometry, view-dependent appearance, and sharp radiance-field variations. Moreover, the formulation remains compatible with multiscale antialiasing strategies for continuous radiance fields [3].

3.3. Stratified Sampling and Perturbation

Given a calibrated camera and a pixel location, a primary ray is cast from the camera center into the scene and parameterized over a depth interval $[t_n, t_f]$ corresponding to near and far bounds. To evaluate the radiance field along this ray, a discrete set of sample points is first generated using stratified sampling [2]. The interval $[t_n, t_f]$ is divided into N equal bins, and a nominal sample position in the ith bin is defined as

$$t_i = t_n + \frac{i}{N}(t_f - t_n), \qquad i = 0, \ldots, N-1 \tag{4}$$

To prevent aliasing artifacts and avoid overfitting to a fixed deterministic discretization, each bin is perturbed with uniform random noise. The perturbed depth sample is given by

$$\tilde{t}_i = t_i + \epsilon_i, \qquad \epsilon_i \sim \mathcal{U}\left(-\frac{\Delta t}{2}, \frac{\Delta t}{2}\right) \tag{5}$$

where $\Delta t = (t_f - t_n)/N$ denotes the bin width. The corresponding 3D point along the ray is computed as

$$\mathbf{x}_i = \mathbf{o} + \tilde{t}_i\, \mathbf{d} \tag{6}$$

with $\mathbf{o}$ the camera origin and $\mathbf{d}$ the unit ray direction.

This stratified sampling with perturbation produces a stochastic but unbiased approximation to the volume rendering integral and increases depth coverage along each ray, which is especially important in sparse-view settings. The same perturbation strategy is employed for both the initial sampling used by the coarse network and the hierarchical resampling used by the fine network. Furthermore, the stochastic sampling procedure remains compatible with the multiscale anti-aliasing treatments proposed for continuous radiance fields in Mip-NeRF [3].

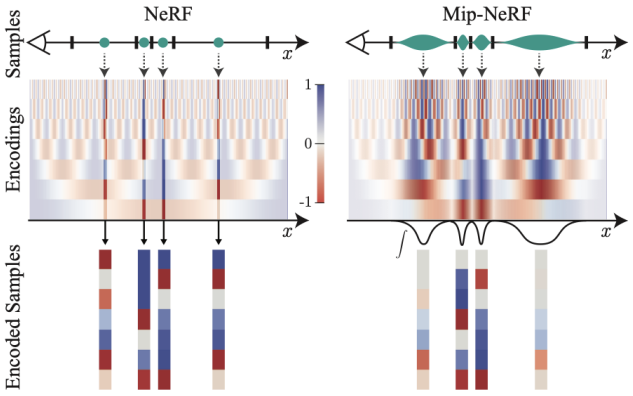

Figure 5. Comparison of positional encoding strategies in NeRF and Mip-NeRF. NeRF applies fixed-frequency sinusoidal encodings to individual point samples, while Mip-NeRF encodes scale-aware conical regions to mitigate aliasing and better represent multiscale structures [3].

3.4. Network Architecture and Configuration

The radiance field is instantiated using two feed-forward neural networks with identical architecture, referred to as the coarse and fine networks, following the standard NeRF-style MLP design adapted to sparse-view street scenes [2, 4, 8]. Each network receives as input the concatenation of the positional encoding of a 3D location $\gamma_x(\mathbf{x})$ and the directional encoding of the viewing direction $\gamma_d(\mathbf{d})$, and maps this high-dimensional representation to a predicted volumetric density and RGB color. Formally, each network realizes the mapping

$$F_\Theta^{(\cdot)} : \left(\gamma_x(\mathbf{x}),\, \gamma_d(\mathbf{d})\right) \mapsto \left(\sigma,\, \mathbf{c}\right) \tag{7}$$

where the superscript $(\cdot)$ denotes either the coarse or the fine branch.

Each MLP consists of L_{mlp} fully connected layers with H hidden units and ReLU activations, along with a skip connection that concatenates the input positional encoding with the feature representation at an intermediate depth, consistent with the original NeRF formulation [2]. In the implementation adopted here, $L_{\text{mlp}} = 8$ and $H = 256$, with the skip connection introduced after the fourth layer. The network is decomposed into two output pathways: a density head that predicts σ from intermediate features dependent solely on $\gamma_x(\mathbf{x})$, and a color head that receives a compact bottleneck representation augmented with the directional encoding $\gamma_d(\mathbf{d})$. This architectural separation enables structured modeling of view-independent geometry and view-dependent appearance, a design shown to be effective in radiance-field reconstruction [2, 3].

The coarse and fine networks share the same architectural structure and hyperparameters but are evaluated on different sampling distributions along each ray. The coarse

branch processes stratified and perturbed samples, providing an initial estimate of the density distribution. This estimate is used to construct a probability distribution that guides hierarchical importance sampling for the fine branch, which is evaluated on the union of coarse samples and additional importance samples. Such a hierarchical design concentrates computational effort near likely surfaces and significantly improves reconstruction quality in sparse-view regimes [4].

4. Training and Optimization Framework

This section describes the complete training pipeline for depth-enhanced NeRF reconstruction on the nuScenes dataset, including hierarchical sampling, LiDAR supervision, distortion regularization, and optimization dynamics.

4.1. Ray Sampling and Hierarchical Rendering

Training operates using stochastic ray sampling. At each iteration, a mini-batch of $N_r = 2048$ rays is selected uniformly from the training images. For each camera ray parameterized as $r(t) = o + td$, the NeRF represents a continuous radiance field [7]:

$$F_\theta(x, d) = (\sigma(x), c(x, d)) \tag{8}$$

where σ denotes density and c the view-dependent color. Given sampled depths $\{t_i\}_{i=1}^{M}$, volumetric rendering is computed as:

$$T_i = \exp\left(-\sum_{j<i} \sigma_j \Delta t_j\right), \quad w_i = T_i \left(1 - e^{-\sigma_i \Delta t_i}\right) \tag{9}$$

$$\hat{C}(r) = \sum_{i=1}^{M} w_i c_i, \qquad \hat{D}(r) = \sum_{i=1}^{M} w_i t_i \tag{10}$$

To reduce variance from uniform sampling, two proposal networks P_1 and P_2 predict coarse densities along each ray. These guide a three-stage hierarchical sampler:

$$128 \rightarrow 64 \rightarrow 48,$$

corresponding to proposal-level sampling followed by fine-level NeRF sampling. This hierarchy concentrates samples in regions with high density gradients, improving stability on outdoor scenes with wide baselines and low-texture areas.

4.2. Loss Formulation and Depth-Guided Supervision

The overall training objective combines photometric reconstruction, sparse LiDAR depth supervision, and geometric distortion regularization:

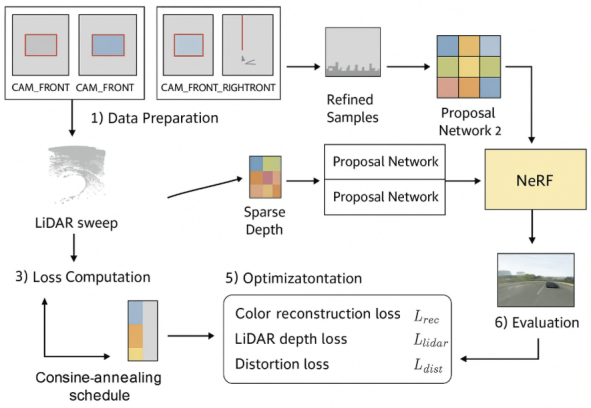

Figure 6. Overall training pipeline for the proposed depth-enhanced NeRF system.

$$\mathcal{L} = \mathcal{L}_{rgb} + \lambda_d \mathcal{L}_{depth} + \lambda_{dist} \mathcal{L}_{dist} \tag{11}$$

The photometric term supervises rendered and observed RGB color:

$$\mathcal{L}_{rgb} = \frac{1}{N_r} \sum_{i=1}^{N_r} \left\| \hat{C}(r_i) - C(r_i) \right\|_2^2. \tag{12}$$

LiDAR depth is incorporated by projecting LiDAR points into each camera frame. For rays with valid depth, the loss is:

$$\mathcal{L}_{depth} = \frac{1}{N_d} \sum_{i=1}^{N_d} m_i \left| \hat{D}(r_i) - D(r_i) \right| \tag{13}$$

where m_i denotes the binary mask of valid LiDAR projections. This term resolves geometric ambiguities arising in outdoor scenes with large depth variation.[6]

To suppress floaters and multi-modal density distributions, we incorporate the Mip-NeRF 360 distortion loss [5]:

$$\mathcal{L}_{dist} = \sum_i \sum_j w_i w_j |t_i - t_j| \tag{14}$$

which penalizes incoherent sample distributions and stabilizes the learned density field. Following prior work, we use $\lambda_d = 1.0$ and $\lambda_{dist} = 0.002$.

4.3. Optimization Strategy and Training Dynamics

We train the model using Adam with $(\beta_1, \beta_2) = (0.9, 0.999)$. The learning rate is scheduled using cosine annealing:

$$\eta_t = \eta_{min} + \tfrac{1}{2}(\eta_{max} - \eta_{min}) \left(1 + \cos\left(\pi \tfrac{t}{T}\right)\right) \tag{15}$$

with $\eta_{max} = 10^{-3}$, $\eta_{min} = 10^{-5}$, and $T = 10{,}000$. This schedule accelerates early convergence and ensures stable late-stage refinement, particularly when depth supervision is sparse.

In practice:

- $\mathcal{L}_{rgb}$ decreases rapidly due to dense gradients.
- $\mathcal{L}_{depth}$ converges slowly due to sparse LiDAR coverage.
- $\mathcal{L}_{dist}$ becomes dominant mid-training, suppressing floaters.

A notable aspect of NeRF is the discrepancy between train and test ray counts. While training processes only 2048 rays per iteration, evaluation requires full-frame rendering (typically 350k–500k rays), resulting in significantly higher computational cost at test time.

Figure 7. Training curves for distortion losses.

5. Results

5.1. Model Rendering Outputs

To gain insights into the evolution of the NeRF model throughout its training process, we implemented a strategy of periodically saving checkpoints. At each of these stages, we rendered images using consistent camera poses from the nuScenes dataset. This approach allowed us to create a series of visual comparisons that clearly showcase the improvements in reconstruction quality as the neural network progressively learns to interpret and recreate the intricate details of the scene.

Figure 8. Model output after 1000 iterations

At the early checkpoint, the renderings are extremely blurry and lack clear structure. Large regions such as road, sky, and buildings appear as broad color blobs, and object boundaries are almost invisible. The model has only learned very coarse color statistics and has not yet formed a stable geometry of the scene.

At the mid-training checkpoint, the outputs become clearer. You can see the layout of the scene better: roads, buildings, and vehicles have more defined shapes, and the order of depth looks more consistent. However, small details like lane markings, traffic signs, and thin poles still appear fuzzy. Some areas also show ghosting or "melting" effects, especially around moving objects.

Figure 9. Model output after 9500 iterations

At the final checkpoint, the renderings show much greater consistency across different views. Object boundaries appear sharper, textures on nearby structures are clearer, and the background geometry is more accurately reconstructed. The benefits of LiDAR depth supervision are evident in the enhanced depth alignment and the reduction of floating artifacts. However, even at this stage, the results are not flawless; some details remain missing, and a few small artifacts are still present. Nonetheless, a comparison across the various checkpoints clearly demonstrates that the model's quality improves steadily with each increase in training iterations.

5.2. Quantitative Evaluation Results

5.2.1. PSNR (Pixel-wise Reconstruction Fidelity)

Peak Signal-to-Noise Ratio (PSNR) is used to quantify the pixel-wise reconstruction accuracy between the ground-truth image I and the rendered image $\hat{I}$ produced by the model.[11]

First, the Mean Squared Error (MSE) between the two images is computed as

$$\text{MSE} = \frac{1}{N} \sum_{p=1}^{N} \Big(I(p) - \hat{I}(p) \Big)^2 \tag{16}$$

where p indexes all pixel positions and color channels, and N is the total number of pixel values (height $\times$ width $\times 3$).

PSNR is then defined as

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{MAX^2}{\text{MSE}} \right) \tag{17}$$

where MAX denotes the maximum possible pixel intensity. Since all images in this project are normalized to the

range $[0, 1]$, we have $MAX = 1$, which simplifies the expression to

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{1}{\text{MSE}} \right) \tag{18}$$

For each training checkpoint, PSNR is computed for all test images and then averaged across the test set. The plotted PSNR curve therefore represents the mean reconstruction quality as a function of training iteration.

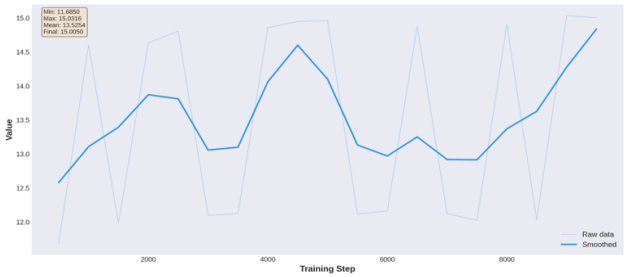

Figure 10. PSNR Evaluation Metrics Plot

The PSNR values start low due to blurred and noisy outputs during early training. As training progresses, PSNR increases, indicating reduced reconstruction error and improved image quality. Toward the end, the curve levels off, showing diminishing returns from further iterations. The final checkpoint achieves the highest PSNR, reflecting improved reconstruction, but still shows some residual artifacts.

5.2.2. SSIM(Structural Similarity Index)

The Structural Similarity Index (SSIM) is used to measure the similarity between a ground-truth image I and a rendered image $\hat{I}$ in terms of luminance, contrast, and structural information[12]

For two corresponding local patches x and y extracted from I and $\hat{I}$, SSIM is defined as

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{19}$$

where μ_x and μ_y are the mean intensities, σ_x^2 and σ_y^2 are the variances, σ_{xy} is the covariance between the patches, and C_1 and C_2 are small constants added for numerical stability.[12]

SSIM is designed to correlate more closely with human perception than simple pixel error, since it explicitly models structural information. Its values lie in the range [0,1], with values closer to 1 indicating higher structural similarity.

The SSIM plot begins with low values, reflecting early renderings without clear structure. As training advances, SSIM steadily increases, indicating improved reproduction of edges and shapes. Growth slows at later checkpoints,

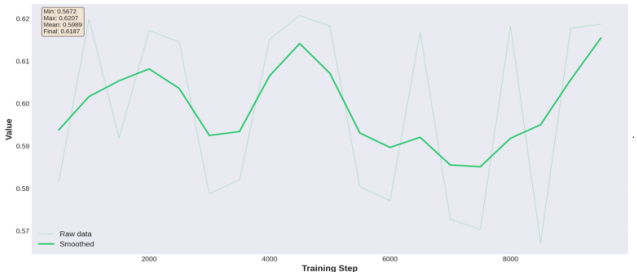

Figure 11. SSIM Evaluation Metrics Plot

suggesting saturation in structural improvements. The final checkpoint records the highest SSIM, indicating the closest structural similarity to the ground truth, though it remains below 1 due to inaccuracies in fine details and dynamic areas.

5.2.3. LPIPS (Learned Perceptual Image Patch Similarity)

The Learned Perceptual Image Patch Similarity (LPIPS) metric is used to evaluate perceptual similarity between a ground-truth image I and a rendered image $\hat{I}$ using deep feature representations from a pretrained convolutional neural network (CNN).[13]

Both images are passed through a fixed CNN, and feature maps $\phi_l(I)$ and $\phi_l(\hat{I})$ are extracted at multiple layers l. For each layer, a distance is computed as

$$d_l(I, \hat{I}) = \frac{1}{H_l W_l} \sum_{h,w} \left\| w_l \odot \left(\phi_l(I)_{h,w} - \phi_l(\hat{I})_{h,w} \right) \right\|_2^2 \tag{20}$$

where H_l and W_l are the spatial dimensions of the feature map, w_l represents learned channel-wise weights, and $\odot$ denotes element-wise multiplication.[13]

The overall LPIPS distance is obtained by summing the contributions across layers:

$$\text{LPIPS}(I, \hat{I}) = \sum_l d_l(I, \hat{I}) \tag{21}$$

As in the other metrics, LPIPS is computed for each test image at every checkpoint and then averaged across the test set. The resulting mean LPIPS values are plotted against training iteration, where lower LPIPS indicates higher perceptual similarity.

The LPIPS plot starts with high values, indicating that early renderings have blurry textures and ambiguous object forms, making them perceptually distant from the ground reality. LPIPS gradually drops during training, indicating that the model generates outputs that are more realistic. Early and mid-training are when improvement is fastest, and when the curve levels off, it slows down. Although the value is still over zero, which is consistent with the remaining artifacts in the produced images, the final checkpoint

Figure 12. LPIPS Evaluation Metrics Plot

obtains the lowest LPIPS, suggesting the best perceptual resemblance.

5.3. Comparison with Existing NeRF Methods and Discussion of Limitations

Method	PSNR ↑	SSIM ↑
Ours	13.52	0.59
S-NeRF	29.38	0.859
S-NeRF++	27.10	0.827
NeRF	26.50	0.811

Table 1. Comparison of reconstruction quality between our model and existing NeRF-based methods.

Our quantitative results are compared with other NeRF-based baselines in Table 1. Compared to S-NeRF, S-NeRF++, and the original NeRF, which all report PSNR values above 26 dB and SSIM values above 0.81, our model achieves a PSNR of 13.52 dB and an SSIM of 0.59, which are significantly lower. This clearly shows a performance discrepancy, but because the evaluation conditions are not perfectly aligned across approaches, the comparison should be read cautiously.[2, 9, 10]

Our model is trained on a single nuScenes scene, with limited viewpoints, sparse depth supervision, and a short training schedule, whereas the reference methods rely on curated datasets, static or semi-static environments, and significantly longer training times. These differences create a more challenging setup for our model.

Several factors contribute to the lower PSNR, SSIM, and LPIPS values observed in our results:

- **Dynamic scene modeled as static:** The nuScenes dataset contains moving vehicles, pedestrians, and changing traffic conditions, whereas our NeRF implementation assumes a fully static scene.[1] This mismatch introduces ghosting and blended geometry, reducing reconstruction fidelity.
- **Limited training time:** The model was trained for a shorter duration and did not fully converge, leading to softer textures and incomplete reconstruction quality.
- **Sparse multi-view coverage:** Using only a single scene with restricted camera viewpoints limits the model's ability to learn complete 3D structure, negatively affecting reconstruction accuracy.
- **Simplified architecture:** In contrast to advanced methods such as S-NeRF and S-NeRF++, our model does not incorporate dynamic-scene modeling, enhanced sampling strategies, or extensive regularization techniques.[8, 10]

Overall, although the absolute performance is lower than state-of-the-art NeRF variants, the results remain consistent with the challenges of this setting. They provide a functional proof-of-concept and highlight opportunities for improvement, particularly in modeling dynamic environments and extending training depth.

6. Future Work

We want to explore several areas in our future work. First, we plan to create better 3D visuals and videos of the scenes we have reconstructed. Allowing a virtual camera to move freely in the environment will help us understand how consistent the lighting is from different viewpoints.

Next, we want to compare our method to a Gaussian Splatting method.[15] Gaussian Splatting is known for training quickly and rendering in real time. We want to see how our NeRF approach matches up in quality and speed, especially for outdoor driving scenes. We also aim to test our model using more datasets from nuScenes. Adding more viewpoints and different urban settings should improve the model's understanding of the environment and help us evaluate its performance in various conditions.

Moreover, we intend to test our model on different datasets, like Waymo.[14] This will help us see how well our approach works with different camera setups or sensor configurations.

Finally, we faced a big challenge with moving objects. Since real-world scenes often contain moving cars, pedestrians, and other dynamic elements, we need to improve our model's ability to handle these changes. This remains an important area for future development.

References

[1] H. Caesar, V. Bankiti, A. Lang, S. Vora, V. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 3, 8

[2] 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. In *ECCV*, 2020. 3, 4, 8

[3] 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 *ICCV*, 2021. 3, 4

[4] M. Niemeyer and A. Geiger. RegNeRF: Regularizing Neural Radiance Fields for View Synthesis from Sparse Inputs. In *CVPR*, 2022. 3, 4, 5

[5] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Ramamoorthi, and P. Srinivasan. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In *CVPR*, 2022. 5

[6] Y. Rematas, A. Martin-Brualla, T. Funkhouser, and V. Ferrari. Urban Radiance Fields. In *CVPR*, 2022. 5

[7] R. Martin-Brualla, P. Hedman, B. Mildenhall, P. Srinivasan, and J. Barron. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*, 2021. 5

[8] Z. Xie, J. Zhang, W. Li, F. Zhang, and L. Zhang. S-NeRF: Neural Radiance Fields for Street Views. *arXiv preprint arXiv:2303.00749*, 2023.

[9] S. Sun et al. LidarRF: Neural Radiance Fields with LiDAR Supervision. In *CVPR*, 2024. 3, 4, 8

[10] Y. Chen et al. S-NeRF++: Neural Radiance Fields for Dynamic Scenes. In *TPAMI*, 2024/2025. 3, 8

[11] A. K. Jain. Fundamentals of Digital Image Processing. Prentice-Hall, 1989. 8

[12] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 2004. 6

[13] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*, 2018. 7

[14] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chou, V. Patnaik, P. Tsui, J. Gupta, A. Joshi, T. Li, S. Shah, B. Strohman, C. Meyer, J. Jackel, and D. Anguelov. The Waymo Open Dataset: Panoramic, High-Resolution, and Diverse Dataset for Autonomous Driving. In *CVPR*, 2020. 7
8

[15] B. Kerbl, G. Kopanas, T. Müller, and M. Wimmer. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. In *SIGGRAPH*, 2023. 8

[16] T. Müller, J. Evans, C. Schied, and A. Keller. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. In *SIGGRAPH*, 2022.