**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

# LiHi-GS: **Li**DAR-Supervised Gaussian Splatting for **Hi**ghway Driving Scene Reconstruction

Pou-Chun Kung[1,2,†], Xianling Zhang[1], Katherine A. Skinner[2], Nikita Jaipuria[1]

***Abstract*—Photorealistic 3D scene reconstruction plays an important role in autonomous driving, enabling the generation of novel data from existing datasets to simulate safety-critical scenarios and expand training data without additional acquisition costs. Gaussian Splatting (GS) facilitates real-time, photorealistic rendering with an explicit 3D Gaussian representation of the scene, providing faster processing and more intuitive scene editing than the implicit Neural Radiance Fields (NeRFs). While extensive GS research has yielded promising advancements in autonomous driving applications, they overlook two critical aspects. First, existing methods mainly focus on low-speed and feature-rich urban scenes and ignore the fact that highway scenarios play a significant role in autonomous driving. Second, while LiDARs are commonplace in autonomous driving platforms, existing methods learn primarily from images and use LiDAR only for initial estimates or without precise sensor modeling, thus missing out on leveraging the rich depth information LiDAR offers and limiting the ability to synthesize LiDAR data. In this paper, we propose a novel GS method for dynamic scene synthesis and editing with improved scene reconstruction through LiDAR supervision and support for LiDAR rendering. Unlike prior works that are tested mostly on urban datasets, to the best of our knowledge, we are the first to focus on the more challenging and highly relevant highway scenes for autonomous driving, which feature sparse sensor views and monotone backgrounds. A project page is available at https://umautobots.github.io/lihi_gs.**

***Index Terms*—Deep Learning for Visual Perception, Mapping, Sensor Fusion**

## I. Introduction

While there has been a lot of recent progress in semi-supervised and weakly supervised deep learning, in practice, most vision tasks for automated driving still rely on supervised learning and often fail to generalize to unseen scenarios. No matter how big the size of the dataset, capturing long tails is impractical, and neglecting them can have devastating consequences [1]. Dataset diversity is thus key to successful real-world deployment. However, data collection and human labeling remain time-intensive and costly.

Synthetic data offers a cost-effective approach to enhance diversity and capture long tails [2]. One such source is gaming engine-based simulation (e.g., CARLA [3]), which provides perfect annotation for free but lacks realism. More recently, Neural Radiance Fields (NeRFs) [4] have emerged as a popular choice for photorealistic novel view synthesis and scene editing. Prior works have shown promising results in autonomous driving scenarios [5, 6, 7, 8, 9, 10]. However, due to the sampling step in NeRF, these approaches face limitations such as computationally intensive rendering and degraded rendering quality at longer distances [11], which are important for highway driving.

In contrast, 3D Gaussian Splatting's (GS) [12] explicit scene representation enables faster and improved rendering for larger scenes and longer distances, coupled with intuitive scene editing. Recent works have shown promising results for autonomous driving scenarios [13, 14, 15, 16, 17]. However, they are limited to urban environments even though autonomous driving applications extend far beyond city streets. In particular, highway scenarios form a major portion of the operating domain for commercial Advanced Driver Assistance Systems (ADAS) and pose distinct challenges for scene reconstruction, such as sparse sensor viewpoints, uniform backgrounds, and repetitive patterns. All of these combined make geometry learning for highway scene reconstruction difficult compared to the feature-rich and lower-speed urban settings.

LiDAR offers dense and precise depth measurements that enhance 3D scene understanding in challenging highway scenarios where camera images suffer from sparse viewpoints and a lack of features. However, existing works only superficially use LiDAR either for initialization [15, 16] or basic positional alignment [13, 18] and, therefore, fail to generalize to LiDAR-sparse regions. More recent approaches attempt to leverage LiDAR depth measurements by projecting point clouds onto camera image planes for depth supervision [14, 19, 20, 21]. However, they only make use of the LiDAR measurements that overlap with camera views, make a strong simplifying assumption that the LiDAR sensor is physically close to the cameras, and fail to support LiDAR novel view synthesis.

To address these limitations, we propose LiHi-GS, a GS method with explicit LiDAR sensor modeling. LiHi-GS not only enables LiDAR supervision during training, resulting in significantly improved scene geometry learning and novel view image rendering, but it also provides the ability for realistic novel view LiDAR rendering. While prior works have primarily focused on urban close-range scene reconstruction and editing (0-50 meters) [5, 14], we bridge this research gap by conducting comprehensive evaluations on highway scenes with objects at 200 meters and beyond, where the benefits of LiDAR supervision become particularly evident (see Figure 1). Our key contributions are as follows:

- Proposed a differentiable LiDAR rendering model for GS,

Manuscript received: April 6, 2025; Revised June 23, 2025; Accepted October 2, 2025.

This paper was recommended for publication by Editor Pascal Vasseur upon evaluation of the Associate Editor and Reviewers' comments.

[1]X. Zhang and N. Jaipuria are with Latitude AI. {xzhang, njaipuria}@lat.ai.

[2]P. Kung and K. A. Skinner are with the Department of Robotics, University of Michigan, Ann Arbor, MI 48109. {pckung, kskin}@umich.edu.

† The project was carried out during P. Kung's internship at Latitude AI.

Digital Object Identifier (DOI): see top of this page.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

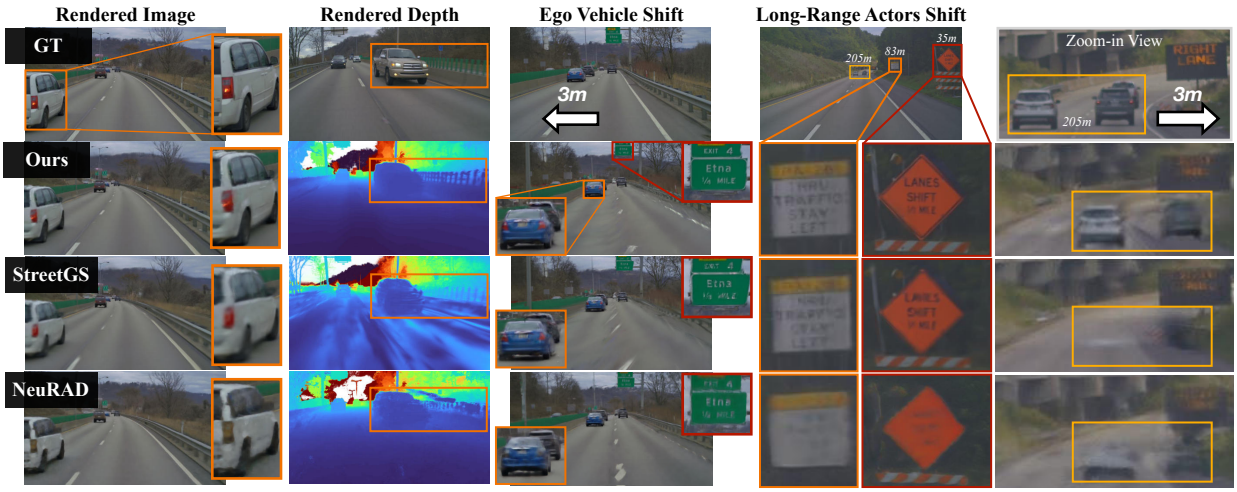



Fig. 1. LiHi-GS (second row) provides higher quality color and depth renderings for interpolated novel views and for ego/actor shifts compared to state-of-the-art NeRF and GS-based methods [5, 14] LiHi-GS does particularly well on actor shifts at longer-ranges (205 meters).

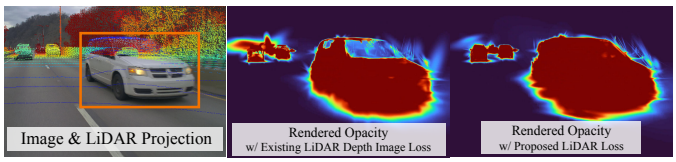



Fig. 2. (Left) Issues with LiDAR projected pseudo-depth supervision, highlighted in the orange box. Points from both near and far distances can map to the same image pixel, resulting in depth ambiguity. In the rendered opacity view (middle), vehicles appear distorted in the case of pseudo-depth supervision, whereas LiHi-GS (right) preserves object geometry and integrity.

enabling both LiDAR rendering and supervision.

- Demonstrated the importance of LiDAR supervision in GS for image and LiDAR novel view synthesis.
- LiHi-GS outperforms state-of-the-art (SOTA) methods in both image and LiDAR synthesis, particularly for view interpolation, ego-view changes, and scene editing tasks.
- Presented the first comprehensive study on long-range highway scene reconstruction and editing, addressing a crucial yet underdeveloped use case in existing research.

## II. Related Work

### A. NeRF-Based Driving Scene Synthesis

NeRF [4] was originally designed for static room-scaled scenes. Follow-up works like [22] extended it to city-scale reconstruction but are also limited to modeling static backgrounds. To model dynamic actors, Neural Scene Graph (NSG) [6] and MARS [23] create a scene graph in which each actor is modeled as an independent NeRF model, and annotated 3D poses are used for scene composition. SUDS [24] and EmerNeRF [25] are unsupervised dynamic scene modeling methods that do not require annotated poses. Some studies also estimate ego-vehicle poses [26, 27].

### B. LiDAR-Integrated NeRFs

Many recent works extend the NeRF formulation for LiDAR rendering [28, 29, 30]. Others focus on LiDAR-only novel view synthesis, like LiDAR-NeRF [8] and NFL [9]. DyNFL [31] and LiDAR4D [32] further extend it to dynamic scenes. UniSim [7] follows NSG-style dynamic scene modeling and supports both image and LiDAR supervision. NeuRAD [5] further addresses multiple approximations in UniSim and models camera rolling shutter, LiDAR ray drop, and LiDAR intensity for more realistic novel view synthesis.

### C. GS-Based Driving Scene Synthesis

3DGS [12] explicitly represents scenes with Gaussians. The Gaussian projection and rasterization expedited training and also enabled real-time rendering. DeformGS [33] extended 3DGS to reconstruct deformable objects. PVG [34] is one of the first GS methods to reconstruct autonomous driving scenes. Recent works [14, 13, 15, 16] extend the NSG idea to GS and show promising results for driving scenarios.

### D. LiDAR-Integrated GS

In contrast to NeRFs, LiDAR measurements can be integrated into GS in multiple ways, of which the most common is initializing Gaussians with the LiDAR point cloud as a geometric prior [15, 16]. However, this approach fails to fully integrate LiDAR data into the training pipeline, leading to potential inaccuracies in scene geometry as the model may overfit to camera images used for training. To tightly integrate LiDAR measurements into GS training, some studies indirectly supervise the GS model with a LiDAR-supervised NeRF model [17] or a LiDAR point cloud map represented by a Gaussian Mixture Model [35]. However, this indirect training fails to provide LiDAR rendering capability.

Alternatively, some works introduce an additional loss to encourage Gaussian centers to align with the LiDAR point cloud [13, 18]. This approach, however, assumes that the LiDAR point cloud fully covers the camera-visible region, resulting in degraded image quality in areas without LiDAR measurements. More recent methods, including StreetGS [14], Omnire [19], and [20, 21], attempt to integrate LiDAR data directly into the training pipeline by projecting the LiDAR depth point cloud onto the camera image plane to create a pseudo-depth image for supervision. However, this approach has limitations. First, the pseudo-depth image assumes depth

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

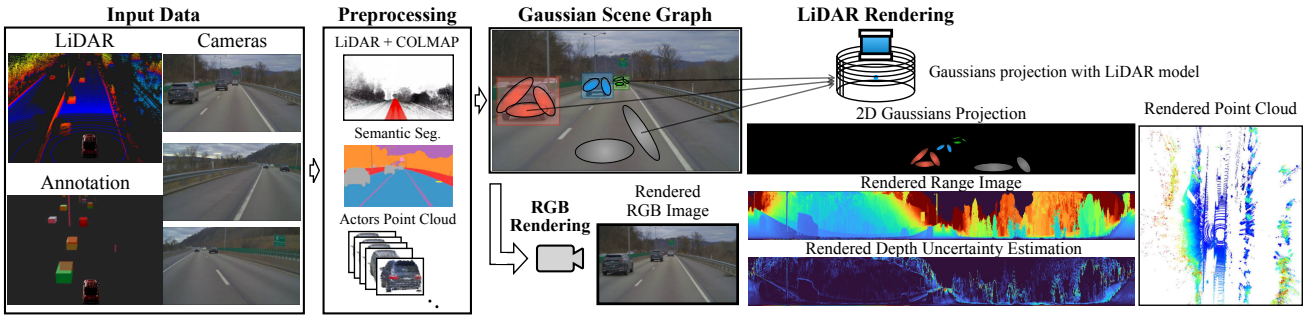



Fig. 3. System overview. LiHi-GS takes multiple cameras, LiDAR, and annotated 3D poses as input. During preprocessing, a LiDAR map combined with a COLMAP sparse point cloud is used to initialize the static scene, while LiDAR points aggregated with human-annotated bounding boxes are used to initialize dynamic objects. During training, images and LiDAR scans are rendered separately and compared with the corresponding ground truth images and LiDAR scans for scene reconstruction.

measurements originate from the camera view, while in autonomous driving setups, LiDAR sensors are typically offset from the cameras. The side effect is shown in Figure 2. Second, this method fails to leverage LiDAR points outside the camera's field of view. To fully leverage LiDAR supervision in GS training, we propose a differentiable LiDAR model for GS that projects 3D Gaussians onto LiDAR range image frames.

Concurrent works such as LiDAR-GS [36] and GS-LiDAR [37] propose LiDAR modeling focused exclusively on LiDAR data synthesis without image data synthesis capability, while LiHi-GS supports both image and LiDAR synthesis. SplatAD [38] is the work most similar to ours, modeling LiDAR and supporting both image and LiDAR synthesis. However, it focuses solely on urban scenes with close-range actors, whereas our method is validated on challenging highway scenes and supports long-range actor reconstruction.

## III. Method

Figure 3 illustrates an overview of the method. Section III-A presents the 3D Gaussian scene representation for dynamic scenes, including the LiDAR visibility for LiDAR rendering. Section III-B discusses the camera modeling convention. Finally, Section III-C describes the proposed LiDAR modeling framework with four key components.

### A. 3D Gaussian Scene Representation

The scene is modeled as a set of 3D Gaussians, $\mathbf{G}$, also referred to as the splat model. $\mathbf{G}$ comprises of $N$ Gaussians, each with mean $\mu$, rotation quaternion $q$, scaling vector $S$, opacity $\alpha$, spherical harmonic (SH) coefficients $sh$, and LiDAR visibility rate $\gamma$:

$$\mathbf{G} = \{G_i : (\mu_i, q_i, S_i, \alpha_i, sh_i, \gamma_i) \mid i = 1, ..., N\}. \quad (1)$$

Each Gaussian's covariance is parameterized by $\Sigma = RSS^TR^T$, where $S \in \mathbb{R}^3$ is a 3D scale vector with square roots of $\Sigma$'s eigenvalues and $R \in \mathrm{SO}(3)$ is the rotation matrix computed from quaternion $q$.

*1) LiDAR Visibility:* In addition to opacity $\alpha$ denoting the opaque state in an image, a LiDAR visibility rate $\gamma$ is introduced to handle the fundamental differences in how LiDAR and camera sensors perceive the environment. For example, LiDAR reflects off of surfaces based on their material and geometry. Low-reflective or semi-transparent surfaces can be missing in LiDAR measurements. Moreover, LiDAR has a limited sensing range compared to cameras. For example, the VLS-128 LiDAR can perceive only 5% of targets $>$ 180m, whereas cameras can capture many more distant objects. Therefore, camera and LiDAR visibility are decoupled for each Gaussian via $\alpha_i^L = \alpha_i\gamma_i$ to better handle the challenging open highway scenes with many targets at far distances.

*2) Gaussian Scene Graph:* Dynamic driving scenes are reconstructed by separating the background and actors' splat models following the NSG [6] used in [13, 14, 15]. The time sequence of actor transformations, obtained from human-labeled bounding boxes, is then used to combine the background and actor splat models with means $\mu_{ij} = R_j\mu_{ij}^o + T_j$ and covariances $\Sigma_{ij} = R_j\Sigma_{ij}^oR_j^T$, where $\mu_{ij}^o$, $\Sigma_{ij}^o$ are the $i$th Gaussian mean and covariance in the $j$th object model, and $R_j$, $T_j$ are object rotation and translation.

### B. Camera Modeling for Gaussian Splatting

*1) Color Rendering:* To render an image from a camera view from the Gaussian primitive, we need to project 3D Gaussians into a 2D image plane as follows:

$$\mu_i^I = KW\mu_i \quad (2)$$

$$\Sigma_i^I = JW\Sigma_iW^TJ^T \quad (3)$$

where $W$ is the camera pose with respect to the world frame, $K$ is the camera intrinsic matrix, and $J$ is the Jacobian of the projective transformation. The pixel color is computed as:

$$\hat{\mathbf{C}}(p) = \sum_{i\in N} c_i\alpha_i' \prod_{j=1}^{i-1}(1-\alpha_j') \quad (4)$$

where $p$ is the pixel in an image, and $c_i$ is the color of a Gaussian obtained using $sh_i$ and view direction $v_{view}$. The opacity of a Gaussian at pixel $p$ is $\alpha'$, defined as:

$$\alpha_i' = \alpha_i \exp\left(-\frac{1}{2}(p-\mu_i^I)^T{\Sigma^I}_i^{-1}(p-\mu_i^I)\right) \quad (5)$$

where $\mu_i^I$ and $\Sigma_i^I$ are the Gaussian center and covariance in projected 2D image space. The rasterization process is denoted as $\boldsymbol{\pi}$, and the rendered color image is $\hat{\mathbf{C}} = \boldsymbol{\pi}(c, \alpha, \mu^I, \Sigma^I)$.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

*2) Depth Rendering:* Recent studies project LiDAR point cloud into the image plane to generate pseudo-depth images $D$ for GS training [14, 20, 21]. The rendered depth image is $\hat{D} = \boldsymbol{\pi}(d, \alpha, \mu^I, \Sigma^I)$ , where $d$ is the depth of the Gaussian center. The depth loss is computed as $\mathcal{L}_{depth} = \|\mathbf{D} - \hat{\mathbf{D}}\|_1$ .

### C. LiDAR Modeling for Gaussian Splatting

*1) Range Image Rendering:* To render LiDAR range images from 3D Gaussians, we first convert 3D Gaussians from the original Cartesian coordinates to spherical coordinates. The mean $\mu = (x, y, z)$ is converted as:

$$\mu^{sph} = \begin{bmatrix} r \\ \theta \\ \phi \end{bmatrix} = \begin{bmatrix} \sqrt{x^2+y^2+z^2} \\ \arctan 2(y, x) \\ \arcsin\left(\frac{z}{r}\right) \end{bmatrix} \tag{6}$$

and the 3D covariance matrix $\Sigma$ is converted as:

$$\Sigma^{sph} = J\Sigma J^T \tag{7}$$

where J is the Jacobian of Eq. 6:

$$J = \begin{bmatrix} \frac{x}{r} & \frac{y}{r} & \frac{z}{r} \\ \frac{-y}{x^2+y^2} & \frac{x}{x^2+y^2} & 0 \\ \frac{-xz}{r^2\sqrt{r^2-z^2}} & \frac{-yz}{r^2\sqrt{r^2-z^2}} & \frac{\sqrt{r^2-z^2}}{r^2} \end{bmatrix} \tag{8}$$

Next, we project 3D Gaussians into the LiDAR range image frame for rasterization. The LiDAR range image $\mathbf{R} \in \mathbb{R}^{\mathcal{M}\times\mathcal{W}}$ is collected from $\mathcal{M}$-beam LiDAR with inclination angles $\Phi = \{\phi_1, \phi_2, \ldots, \phi_{\mathcal{M}}\}$ and azimuth resolution $\mathcal{W}$. The Gaussian mean in the range image is:

$$\mu^L = \begin{bmatrix} v \\ u \end{bmatrix} = \begin{bmatrix} \arg\min_i |\phi - \phi_i| \\ \frac{\theta \mathcal{W}}{2\pi} \end{bmatrix} \tag{9}$$

The Gaussian covariance in the LiDAR range image is:

$$\Sigma^L = A\Sigma^{sph}A^T \tag{10}$$

$$A = \begin{bmatrix} \frac{1}{\Delta\theta} & 0 \\ 0 & \frac{\mathcal{W}}{2\pi} \end{bmatrix} \tag{11}$$

Given the non-uniform inclination beam angle distribution, the Jacobian $A$ is dependent on the elevation resolution $\Delta\theta_v$.

Finally, the LiDAR range image is rendered as:

$$\hat{\mathbf{R}} = \boldsymbol{\pi}(d, \alpha^L, \mu^L, \Sigma^L), \tag{12}$$

where $\alpha^L$ is LiDAR visibility introduced in Sec. III-A1.

*2) 2D Gaussian Scale Compensation:* In the original GS image rasterization model, a small Gaussian far from the camera image plane becomes invisible in the projected pixel due to numerical instability. However, since LiDAR emits laser beams to measure the distance of objects, a Gaussian should remain fully visible if its center is sufficiently close to the ray, regardless of its scale. To address Gaussians that are lost during projection, we rescale the 2D Gaussian when it is near the ray and has a scale below the visible threshold. Eigen decomposition is applied to Gaussians to get their 2D scales after projection $\Sigma_i^L = R_i^L S_i^L {R_i^L}^{-1}$, where $S_i^L = \text{diag}(s_{i1}^2, s_{i2}^2)$.

Visible scale is dependent on the distance of the Gaussian:

$$s_{vis}^i = d_i \tan\left(\frac{2\pi}{\mathcal{W}}\right) \tag{13}$$

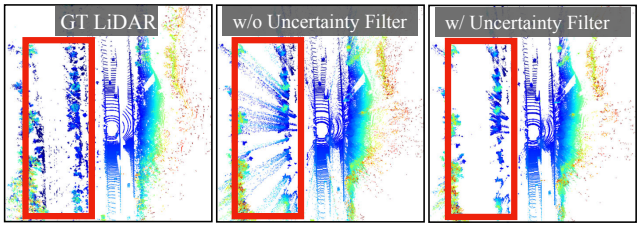



Fig. 4. Depth uncertainty filter removes floating artifacts on object edges from the rendered point cloud.

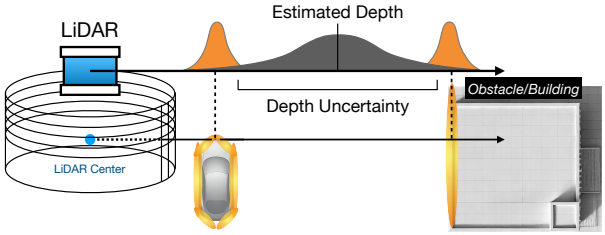



Fig. 5. LiDAR depth rendering can lead to incorrect depth estimates at object edges, causing floating artifacts between adjacent objects. To address this, depth uncertainty is used to filter out the artifact on object edges, as shown in Figure 4.

We adjust the Gaussian scale to be invisible if three standard deviations fall below the visible scale $s_{vis}$.

$$\tilde{s}_{ij} = \begin{cases} \frac{s_{vis}^i}{3}, & \text{if } s_{ij} < \frac{s_{vis}^i}{3} \\ s_{ij}, & \text{otherwise} \end{cases} \tag{14}$$

Finally, the 2D covariance matrix is constructed using the updated scale $\tilde{\Sigma}_i^L = R_i^L \,\text{diag}(\tilde{s}_{i1}, \tilde{s}_{i2}) {R^L}^{-1}$.

*3) Depth Uncertainty Rendering:* We note that the point cloud rendered using GS can include floating artifacts on object edges (Figure 4). This is mainly because depth discontinuities are hard to represent using the inherently continuous Gaussian distribution. To render a crisp and clean point cloud with clear object edges, we introduce depth uncertainty rendering to identify pixels that cause floating noise in the rendered image, as shown in Figure 5. An uncertainty threshold is then applied to filter out floating points. Depth uncertainty is calculated as the incremental depth variance along the CUDA rasterization. The total accumulated alpha $\Lambda_i$ along a ray at the $i$-th Gaussian at a pixel is:

$$\Lambda_i = \Lambda_{i-1} + {\alpha^L}'_i, \tag{15}$$

where $i$ is depth-sorted index, $\Lambda$ is initialized with $\Lambda_0 = 0$, and $\alpha_n^{L'}$ is the opacity contributed by a Gaussian at a pixel.

Depth uncertainty $\Gamma_i$ can then be computed incrementally with Welford's online algorithm:

$$m_i = m_{i-1} + \frac{{\alpha^L}'_i}{\Lambda_i}(d_i - m_{i-1}) \tag{16}$$

$$\Gamma_i = \Gamma_{i-1} + {\alpha^L}'_i \cdot \frac{\Lambda_{i-1}}{\Lambda_i}(d_i - m_{i-1})(d_i - m_{i-1})^T \tag{17}$$

where $m_i$ is the running mean. An uncertainty threshold $\Gamma_{th}$ is then applied to filter out pixels with large depth uncertainty. Figure 4 demonstrates the rendered point cloud with and without the uncertainty filter.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

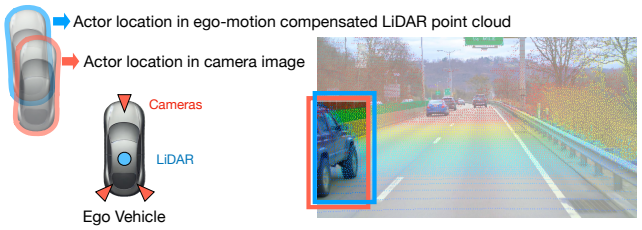



Fig. 6. Camera-LiDAR actor pose misalignment for high-speed actors.

*4) Camera-LiDAR Decoupled Pose Optimization:* Our method heavily relies on accurately labeled poses for actor reconstruction. However, human annotations can be imperfect, and the camera and LiDAR measurements can also be misaligned for fast-moving actors due to the temporal observation difference, as shown in Figure 6. Unlike [5, 14], which optimize a unified pose, we propose a decoupled pose optimization to account for pose misalignment between sensors. Thus, the GS model for LiDAR and camera rendering is constructed separately using different poses $T^j_{lidar}$ and $T^j_{camera}$.

### D. Training Losses

The total optimization objective is as follows:

$$\mathcal{L} = \mathcal{L}_c + \lambda_1 \mathcal{L}_{lidar} + \lambda_2 \mathcal{L}^L_{opa} + \lambda_3 \mathcal{L}^s_{reg} + \lambda_4 \mathcal{L}^c_{opa} \quad (18)$$

$$\mathcal{L}^c_{opa} = \mathcal{L}_{sky} + \mathcal{L}_{fg} + \mathcal{L}^{obj}_{reg} \quad (19)$$

| Our Dataset (Highway Scenes) | | | | | | |
|---|---|---|---|---|---|---|
| Method | Image | | | | LiDAR | |
| | PSNR†↑ | PSNR↑ | LPIPS↓ | SSIM↑ | Mean↓ | Med.↓ |
| Instant-NGP [39] | 18.73 | 25.94 | 0.37 | 0.856 | - | - |
| 3DGS [12] | 17.60 | 27.36 | 0.32 | 0.918 | - | - |
| NeuRAD [5] | 19.87 | 30.38 | 0.28 | 0.911 | 2.30 | 0.38 |
| StreetGS [14] | 24.56 | 29.44 | 0.32 | 0.921 | 8.70* | 4.73* |
| SplatAD [38] | 24.71 | 29.89 | 0.34 | 0.925 | 2.23 | 0.44 |
| LiHi-GS (Ours) | 26.43 | 30.69 | 0.27 | 0.927 | 1.66 | 0.43 |

*Rendering LiDAR with our proposed LiDAR rendering.

TABLE I
Image and LiDAR novel view rendering on self-collected dataset. PSNR† denotes the PSNR of moving objects. Best results in red, second best in orange, and third in yellow.

| Waymo Open Dataset (Urban Scenes) | | | | | | |
|---|---|---|---|---|---|---|
| Method | Image | | | | LiDAR | |
| | PSNR†↑ | PSNR↑ | LPIPS↓ | SSIM↑ | Mean↓ | Med.↓ |
| Instant-NGP [39] | 25.09 | 28.70 | 0.19 | 0.887 | - | - |
| 3DGS [12] | 25.05 | 28.86 | 0.13 | 0.937 | - | - |
| DeformGS [33] | 21.42 | 28.86 | 0.17 | 0.876 | - | - |
| PVG [34] | 25.24 | 29.23 | 0.19 | 0.863 | - | - |
| Omnire [19] | 26.30 | 29.22 | 0.15 | 0.866 | - | - |
| NeuRAD [5] | 28.83 | 29.26 | 0.20 | 0.844 | 0.69 | 0.07 |
| StreetGS [14] | 29.02 | 33.24 | 0.15 | 0.943 | 5.00* | 2.08* |
| LiHi-GS (Ours) | 29.53 | 33.81 | 0.13 | 0.945 | 0.37 | 0.06 |

*Rendering LiDAR with our proposed LiDAR rendering.

TABLE II
Image and LiDAR novel view rendering on the Waymo Open Dataset. PSNR† denotes the PSNR of moving objects. Best results in red and second best in orange.

Color loss $\mathcal{L}_c$ follows the color image loss from [12]. LiDAR loss is calculated as the L1 difference between the target and rendered range images. $\mathcal{L}^L_{opa}$ encourages LiDAR visibility $\alpha^L = 1$ for pixels with depth returns in the range image. This loss term is crucial considering LiDAR's 360° field of view, which includes regions outside camera coverage. $\mathcal{L}^s_{reg}$ is a scale regularization term designed to make Gaussians more evenly shaped, preventing spike artifacts and providing more accurate depth rendering, inspired by [40]. $\mathcal{L}^c_{opa}$ encompasses all opacity losses from the camera image. $\mathcal{L}_{sky}$ ensures sky pixels have low opacity, while $\mathcal{L}_{fg}$ guides foreground pixels to have high opacity. Additionally, $\mathcal{L}^{obj}_{reg}$ is a regularization term used to improve decomposition effects, following [14].

## IV. Experiments

### A. Datasets

Existing research often relies on open-source datasets for evaluation that are dominated by texture-rich urban scenarios with dense sensor coverage. This limits their applicability to feature-sparse and view-sparse highway scenarios, a key use case for autonomous driving. To address this gap, we conduct experiments on a self-collected dataset with challenging highway environments. Our data collection vehicle is equipped with a VLS-128 LiDAR and three cameras facing front, back-left, and back-right, synchronized at 10 Hz. We gather highway data across three U.S. states (Michigan, Pennsylvania, and North Carolina). Four challenging segments, ranging from 16 to 25 seconds in length and featuring multiple moving objects, high ego speeds, small-scale far-field objects, and monotonous backgrounds, are used in our experiments. We sampled every 10th frame for evaluation, and used the remaining frames exclusively for training.

Additionally, to evaluate LiHi-GS on urban scenes and public datasets, we conduct experiments on the Waymo open dataset [41] on sequences used in [42] and pandaSet [43] and nuScenes [44] datasets on sequences used in [38].

### B. Baselines

LiHi-GS is compared with multiple baselines on both our dataset and the Waymo dataset. On our dataset, we compare against: Instant-NGP [39], an image-only NeRF-based method; NeuRAD [5], a NeRF-based method for driving scenes that supports LiDAR supervision and rendering; and StreetGS [14], a GS-based method designed for autonomous driving. For 3DGS [12], we use a re-implementation from splatfacto [45]. We also deploy the official implementation of concurrent work, SplatAD [38], on our highway dataset.

On the Waymo dataset, we perform benchmarking experiments with Omnire [19], PVG [34], and DeformGS [33] using [19] implementation, which supports the Waymo data format. We also compare with GS-LiDAR [37], on LiDAR-only novel view rendering on Waymo sequences used in [37].

We additionally provide comparisons on the PandaSet and nuScenes datasets with baselines reported in the SplatAD [38], which are included on the project page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

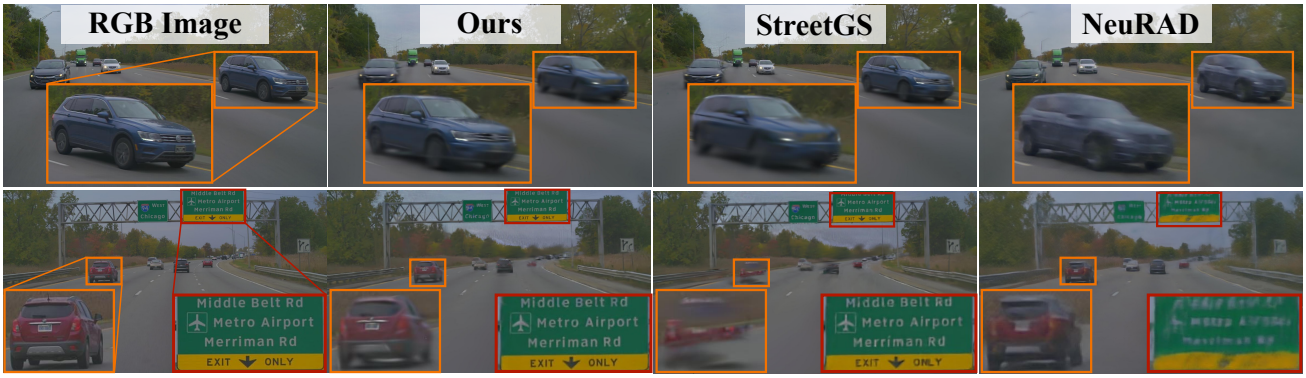



Fig. 7. Qualitative comparison of image rendering. LiHi-GS delivers the best overall image quality, successfully capturing details on moving actors and a distant traffic sign, whereas the results from StreetGS and NeuRAD appear blurred.

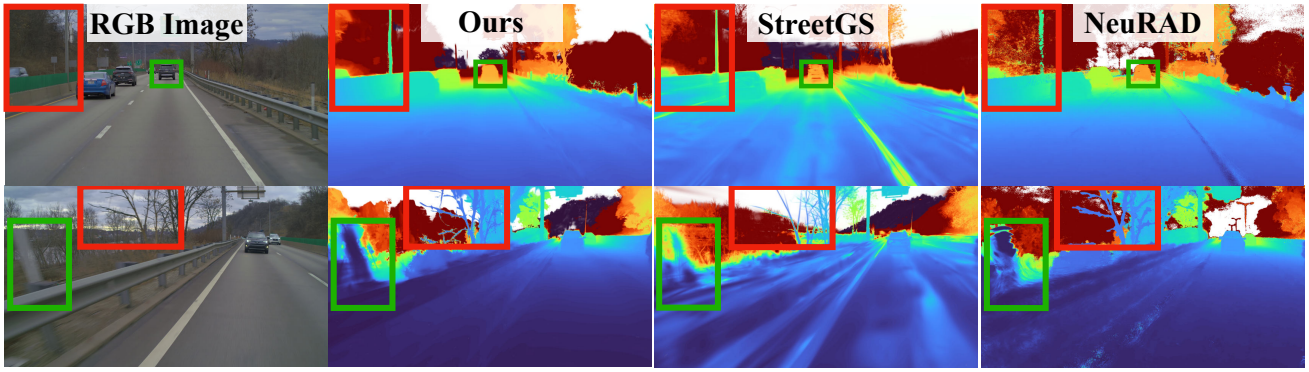
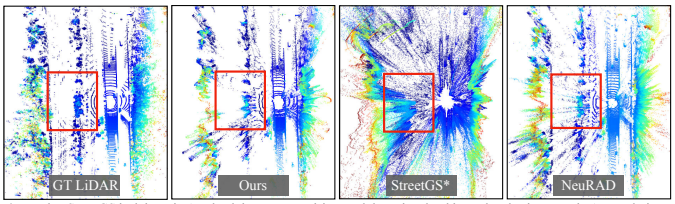



Fig. 8. Qualitative comparison of depth rendering. Results show cleaner geometry over StreetGS due to the added LiDAR supervision. NeuRAD depth also has overall correct geometry, but is noisier. Using a 500m depth threshold to classify the sky (white) showed that NeuRAD introduces noisy depth measurements on the sky and fails to model the distant mountain.


Fig. 9. Qualitative comparison of image rendering on our highway dataset. LiHi-GS produces sharper and cleaner reconstructions of moving vehicles compared to the blurry results of SplatAD, demonstrating the effectiveness of decoupled camera–LiDAR pose optimization for moving actor reconstruction.




*Note that StreetGS is doing LiDAR depth image supervision, and the point cloud is rendered using our LiDAR rendering method. This indicates the scene geometry can be wrong with depth image supervision.

Fig. 10. Point cloud synthesized using different methods. Our method provides a clean point cloud with minimal noise on object edges.


Fig. 11. Qualitative comparison of novel view rendering on the Waymo Dataset. LiHi-GS captures clear details on the vehicle.

### C. Novel View Image and LiDAR Synthesis

Table I shows the quantitative comparison of our LiHi-GS with SOTA baselines on our highway data for novel view rendering quality. LiHi-GS outperforms all baselines on all image metrics including moving object reconstruction, highlighting the effectiveness of the proposed LiDAR supervision. LiHi-GS also surpasses SplatAD, demonstrating the importance of the proposed decoupled pose optimization and LiDAR visibility components, which are specifically designed for highway scenes. Qualitative comparison of rendered image and depth is shown in Figures 7 and 8. Comparison with SplatAD is shown in Figure 9. LiHi-GS captures finer details in the rendered image and accurately renders the scene geometry in rendered depth. In terms of LiDAR rendering performance, LiHi-GS consistently has the lowest L1 mean error due to the decoupled pose optimization and uncertainty filter that is lacking in NeuRAD, as shown in Figure 10. The actor/ego shift experiments (Table IV) demonstrate that LiHi-GS yields better geometry reconstruction.

Table II shows the quantitative results on the public Waymo Open Dataset [41]. We observed a similar trend of improvement as experiments conducted on our self-collected dataset. LiHi-GS outperforms existing works on both image and LiDAR novel view synthesis. The overall improvement of LiHi-GS on Waymo urban data is smaller than in highway scenes, suggesting LiDAR supervision is more critical in challenging highway scenarios. However, the qualitative results in Figure 11 show that LiHi-GS still captures more details with

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

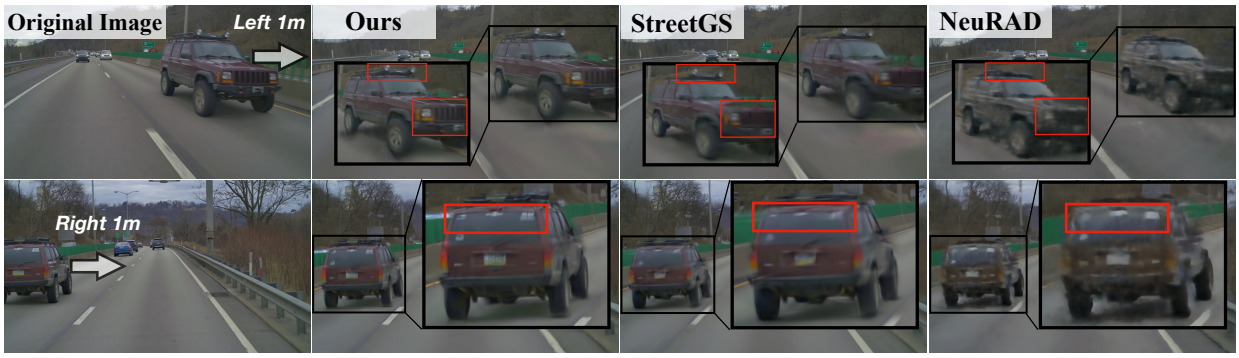



Fig. 12. Scene editing with lateral actor shift. LiHi-GS has the best overall image quality, given the accurate geometry learned from LiDAR supervision.

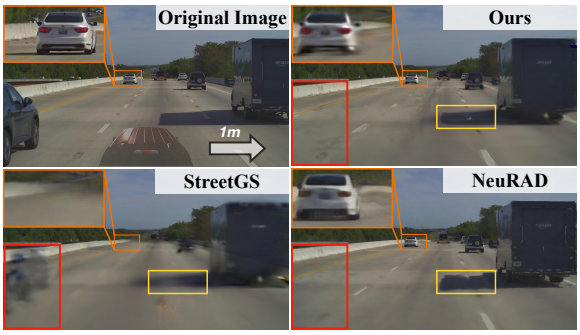



Fig. 13. Scene editing with lateral ego-vehicle shift. LiHi-GS reconstructs vehicles with correct color, better shadow shape, and fewer ghost artifacts.

LiDAR supervision on urban scenes. The image and LiDAR rendering achieve 72 and 5 FPS on a NVIDIA A100 GPU.

We report additional results on the nuScenes [44] and PandaSet [43] datasets on our project page, demonstrating LiHi-GS's strong generalization across public datasets and outperforming SplatAD, thanks to our camera–LiDAR decoupled pose optimization and LiDAR visibility rate.

Table III compares LiDAR-only LiHi-GS with concurrent GS-LiDAR [37] on the Waymo Dataset. LiHi-GS provides better LiDAR range image synthesis and offers a competitive median error while providing extra camera rendering.

| LiDAR-Only Evaluation | | | | |
|---|---|---|---|---|
| Method | LiDAR Range Image | | | |
| | PSNR↑ | LPIPS↓ | SSIM↑ | Med.↓ |
| GS-LiDAR [37] | 22.25 | 0.07 | 0.839 | 0.020 |
| LiHi-GS (Ours) | 23.00 | 0.05 | 0.894 | 0.023 |

TABLE III
Comparing LiDAR novel view rendering with GS-LiDAR on Waymo sequences used in [37] with LiDAR-only training.

| | Ego Lateral Shift (FID) | | | Actor Lateral Shift (FID) | | |
|---|---|---|---|---|---|---|
| | 1m | 2m | 3m | 1m | 2m | 3m |
| NeuRAD | 63.2 | 82.42 | 101.45 | 51.25 | 55.61 | 79.70 |
| StreetGS | 56.09 | 69.59 | 82.24 | 43.65 | 54.48 | 65.86 |
| LiHi-GS | 40.70 | 57.47 | 80.69 | 30.91 | 43.72 | 54.50 |

TABLE IV
FID (↓) scores when shifting ego vehicle or actors pose.

| Module Ablations | Image | | | LiDAR |
|---|---|---|---|---|
| | PSNR↑ | IPIPS↓ | SSIM↑ | Mean↓ |
| w/o $\mathcal{L}_{opa}^{L}$ | 30.54 | 0.229 | 0.927 | 2.94 |
| w/o $\mathcal{L}_{reg}^{s}$ | 29.65 | 0.234 | 0.926 | 2.76 |
| w/o Decoupled pose optimization | 30.49 | 0.228 | 0.927 | 2.67 |
| w/o 2D scale compensation | 30.22* | 0.245* | 0.923* | 2.71* |
| w/o LiDAR visibility | 30.59 | 0.228 | 0.928 | 2.67 |
| **Proposed Full** | 31.63 | 0.208 | 0.935 | 2.61 |
| LiDAR Loss Ablations | | | | |
| Only $\mathcal{L}_c$ [13] | 29.59 | 0.293 | 0.923 | 8.66 |
| $+\mathcal{L}_{depth}$ [14, 20, 21] | 29.70 | 0.285 | 0.925 | 6.23 |
| + Proposed $\mathcal{L}_{\text{lidar}}$ | 31.63 | 0.208 | 0.935 | 2.61 |

*Disabling 2D scale compensation caused an exponential increase in memory usage, leading the job to crash midway.

TABLE V
Ablation studies.

## D. Scene Editing With Ego and Actor Shifting

We further evaluate the rendering quality with ego-vehicle and surrounding actors shifting. We apply lateral shifting for both ego and actors from 1 meter to 3 meters. Since the ground truth after scene editing does not exist, we follow [5] to report the FID score to analyze the data synthesis quality. Table IV shows that LiHi-GS generates the most realistic image from new viewpoints that are off from the original trajectory and learns a better representation of actors. The qualitative results of actor and ego shifting are shown in Figures 12 and 13.

## E. Ablations

Table V presents the quantitative results when removing the different proposed designs from our pipeline. Removing LiDAR opacity loss $\mathcal{L}_{opa}^{L}$ leads to the highest LiDAR L1 mean error. Scale regularization $\mathcal{L}_{reg}^{s}$ also has a great impact on both image and LiDAR quality. Rendered depth without scale regularization is inaccurate and can degrade the image quality during LiDAR supervision. Disabling decoupled pose optimization leads to camera-LiDAR pose misalignment and worse PSNR rendering quality. The model without 2D scale compensation crashed midway through training due to memory overhead. Without scale compensation, LiDAR supervision inflates the size of distant Gaussians to match the projected LiDAR pixels, leading to unrealistic geometry and excessive memory usage. In the end, LiDAR visibility significantly improves both image and LiDAR rendering quality. This reflects the fundamental differences between the two sensors.

We also compare the impact of the proposed loss with depth image loss used in prior works. The results show that depth image loss does not enhance image rendering quality and can sometimes degrade it. In contrast, our LiDAR supervision improves both image rendering quality and LiDAR accuracy.

**IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.**

## V. Conclusions

In summary, we introduce a differentiable GS LiDAR rendering model for challenging highway scenes. Our results demonstrate that LiHi-GS achieves SOTA performance in novel view synthesis. We highlight the importance of LiDAR supervision in learning accurate scene geometry, which significantly enhances image rendering quality, especially under actor or ego shifts. Unlike prior works that primarily focus on urban datasets, our approach is the first to evaluate on monotonous highway data, bridging the gap between existing research and real-world autonomous driving applications.

## References


[1] "Tesla autopilot feature was involved in 13 fatal crashes, us regulator says." [Online]. Available: https://www.theguardian.com/technology/2024/apr/26/tesla-autopilot-fatal-crash

[2] X. Zhang, N. Tseng, A. Syed, R. Bhasin, and N. Jaipuria, "Simbar: Single image-based scene relighting for effective data augmentation for automated driving vision tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 3718–3728.

[3] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[4] 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," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[5] A. Tonderski, C. Lindström, G. Hess, W. Ljungbergh, L. Svensson, and C. Petersson, "Neurad: Neural rendering for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 895–14 904.

[6] J. Ost, F. Mannan, N. Thuerey, J. Knodt, and F. Heide, "Neural scene graphs for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2856–2865.

[7] Z. Yang, *et al.*, "Unisim: A neural closed-loop sensor simulator," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1389–1399.

[8] T. Tao, *et al.*, "Lidar-nerf: Novel lidar view synthesis via neural radiance fields," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 390–398.

[9] S. Huang, *et al.*, "Neural lidar fields for novel view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 18 236–18 246.

[10] A. Krishnan, *et al.*, "Lane: Lighting-aware neural fields for compositional scene synthesis," *arXiv preprint arXiv:2304.03280*, 2023.

[11] H. Turki, D. Ramanan, and M. Satyanarayanan, "Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 922–12 931.

[12] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[13] X. Zhou, Z. Lin, X. Shan, Y. Wang, D. Sun, and M.-H. Yang, "Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 634–21 643.

[14] Y. Yan, *et al.*, "Street gaussians for modeling dynamic urban scenes," *arXiv preprint arXiv:2401.01339*, 2024.

[15] M. Khan, *et al.*, "Autosplat: Constrained gaussian splatting for autonomous driving scene reconstruction," *arXiv preprint arXiv:2407.02598*, 2024.

[16] H. Zhou, *et al.*, "Hugs: Holistic urban 3d scene understanding via gaussian splatting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 336–21 345.

[17] C. Zhao, *et al.*, "Tclc-gs: Tightly coupled lidar-camera gaussian splatting for surrounding autonomous driving scenes," *arXiv preprint arXiv:2404.02410*, 2024.

[18] S. Hong, J. He, X. Zheng, C. Zheng, and S. Shen, "Liv-gaussmap: Lidar-inertial-visual fusion for real-time 3d radiance field map rendering," *IEEE Robotics and Automation Letters*, 2024.

[19] Z. Chen, *et al.*, "Omnire: Omni urban scene reconstruction," *arXiv preprint arXiv:2408.16760*, 2024.

[20] N. Huang, *et al.*, "S3 gaussian: Self-supervised street gaussians for autonomous driving," *arXiv preprint arXiv:2405.20323*, 2024.

[21] J. Cui, *et al.*, "Letsgo: Large-scale garage modeling and rendering via lidar-assisted gaussian primitives," *arXiv preprint arXiv:2404.09748*, 2024.

[22] M. Tancik, *et al.*, "Block-nerf: Scalable large scene neural view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8248–8258.

[23] Z. Wu, *et al.*, "Mars: An instance-aware, modular and realistic simulator for autonomous driving," in *CAAI International Conference on Artificial Intelligence*. Springer, 2023, pp. 3–15.

[24] H. Turki, J. Y. Zhang, F. Ferroni, and D. Ramanan, "Suds: Scalable urban dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 375–12 385.

[25] J. Yang, *et al.*, "Emernerf: Emergent spatial-temporal scene decomposition via self-supervision," *arXiv preprint arXiv:2311.02077*, 2023.

[26] F. Tosi, *et al.*, "How nerfs and 3d gaussian splatting are reshaping slam: a survey," *arXiv preprint arXiv:2402.13255*, vol. 4, p. 1, 2024.

[27] X. Li, *et al.*, "Stereo 3d gaussian splatting slam for outdoor urban scenes," *arXiv preprint arXiv:2507.23677*, 2025.

[28] K. Rematas, *et al.*, "Urban radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 932–12 942.

[29] A. Carlson, M. S. Ramanagopal, N. Tseng, M. Johnson-Roberson, R. Vasudevan, and K. A. Skinner, "Cloner: Camera-lidar fusion for occupancy grid-aided neural representations," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2812–2819, 2023.

[30] S. Isaacson, P.-C. Kung, M. Ramanagopal, R. Vasudevan, and K. A. Skinner, "Loner: Lidar only neural representations for real-time slam," *IEEE Robotics and Automation Letters*, 2023.

[31] H. Wu, X. Zuo, S. Leutenegger, O. Litany, K. Schindler, and S. Huang, "Dynamic lidar re-simulation using compositional neural fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 988–19 998.

[32] Z. Zheng, F. Lu, W. Xue, G. Chen, and C. Jiang, "Lidar4d: Dynamic neural fields for novel space-time view lidar synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5145–5154.

[33] Z. Yang, X. Gao, W. Zhou, S. Jiao, Y. Zhang, and X. Jin, "Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 20 331–20 341.

[34] Y. Chen, C. Gu, J. Jiang, X. Zhu, and L. Zhang, "Periodic vibration gaussian: Dynamic urban scene reconstruction and real-time rendering," *arXiv preprint arXiv:2311.18561*, 2023.

[35] C. Jiang, R. Gao, K. Shao, Y. Wang, R. Xiong, and Y. Zhang, "Li-gs: Gaussian splatting with lidar incorporated for accurate large-scale reconstruction," *arXiv preprint arXiv:2409.12899*, 2024.

[36] Q. Chen, S. Yang, S. Du, T. Tang, P. Chen, and Y. Huo, "Lidar-gs: Real-time lidar re-simulation using gaussian splatting," *arXiv preprint arXiv:2410.05111*, 2024.

[37] J. Jiang, C. Gu, Y. Chen, and L. Zhang, "Gs-lidar: Generating realistic lidar point clouds with panoramic gaussian splatting," *arXiv preprint arXiv:2501.13971*, 2025.

[38] G. Hess, C. Lindström, M. Fatemi, C. Petersson, and L. Svensson, "Splatad: Real-time lidar and camera rendering with 3d gaussian splatting for autonomous driving," *arXiv preprint arXiv:2411.16816*, 2024.

[39] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM transactions on graphics (TOG)*, vol. 41, no. 4, pp. 1–15, 2022.

[40] T. Xie, *et al.*, "Physgaussian: Physics-integrated 3d gaussians for generative dynamics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4389–4398.

[41] J. Mei, *et al.*, "Waymo open dataset: Panoramic video panoptic segmentation," in *European Conference on Computer Vision*. Springer, 2022, pp. 53–72.

[42] L. AI, "Street-gaussians-ns," 2024. [Online]. Available: https://github.com/LightwheelAI/street-gaussians-ns/tree/main

[43] P. Xiao, *et al.*, "Pandaset: Advanced sensor suite dataset for autonomous driving," in *2021 IEEE international intelligent transportation systems conference (ITSC)*. IEEE, 2021, pp. 3095–3101.

[44] H. Caesar, *et al.*, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[45] V. Ye, *et al.*, "gsplat: An open-source library for gaussian splatting," *arXiv preprint arXiv:2409.06765*, 2024.