# DriveR$^3$: Feed-Forward Gaussian Reconstruction for CameRa-LiDAR Simulation in Autonomous Driving

**Anonymous authors**
Paper under double-blind review

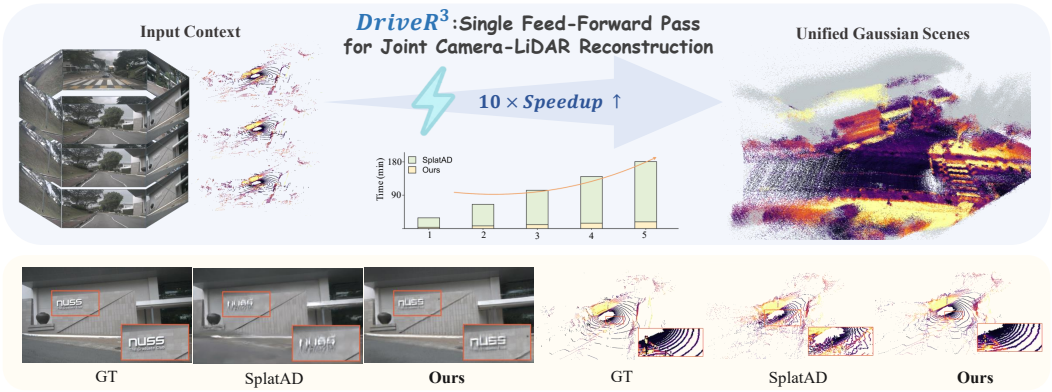


Figure 1: **DriveR$^3$ reconstructs unified dynamic Gaussian scenes for joint camera-LiDAR simulation in a single forward pass.** From sparse multi-view images and LiDAR observations, it predicts shared geometry and sensor-specific attributes for rendering at novel poses and times. The lower panels compare RGB and LiDAR intensity reconstruction with SplatAD.

## Abstract

Cameras and LiDAR provide complementary appearance and geometric information for autonomous driving, making their joint simulation essential for realistic evaluation of multi-sensor perception systems. However, existing joint reconstruction methods typically rely on time-consuming per-scene optimization, while efficient feed-forward reconstruction approaches primarily target visual geometry recovery and image synthesis, leaving joint camera-LiDAR simulation underexplored. To address this gap, we introduce DriveR$^3$, a feed-forward framework that directly reconstructs a unified dynamic Gaussian scene representation from sparse camera and LiDAR observations. Our key design establishes spatiotemporal LiDAR-Gaussian associations, enabling shared dynamic Gaussian geometry to be jointly learned from complementary image reconstruction and LiDAR range supervision. DriveR$^3$ further predicts Gaussian-level LiDAR intensity by adaptively fusing multi-source intensity cues. Experiments on nuScenes demonstrate that DriveR$^3$ enables feed-forward joint camera-LiDAR reconstruction while also improving visual reconstruction quality, highlighting the benefit of complementary sensor information. Compared with the per-scene optimization method SplatAD, DriveR$^3$ achieves better results on multiple LiDAR reconstruction metrics and an approximately $10\times$ speedup in scene reconstruction.

## 1 Introduction

Reliable evaluation of autonomous driving systems requires realistic closed-loop simulation, as the ego vehicle may deviate from recorded trajectories and encounter sensor observations that were never captured (Yang et al., 2023). This requirement becomes particularly important for end-to-end



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

driving models (Hu et al., 2023; Jiang et al., 2023; Guo et al., 2024; Yang et al., 2025b; Zhang et al., 2026b; Mao et al., 2026; Guo et al., 2026), which directly couple sensor observations with planning and control. Most prior reconstruction-based simulation methods optimize a separate representation for each scene, making it costly to scale simulation across diverse driving environments. Recent feed-forward reconstruction methods offer a promising alternative by directly inferring renderable scenes from recorded images. However, these methods are designed primarily for visual reconstruction and novel-view synthesis, while many advanced autonomous driving systems rely on camera-LiDAR multimodal fusion. Extending feed-forward reconstruction beyond vision is therefore essential, yet jointly reconstructing camera and LiDAR observations within a shared dynamic scene remains largely unexplored.

This gap reflects a broader divide between two lines of research. Multi-sensor simulators such as UniSim, NeuRAD, and SplatAD prioritize simulation fidelity by jointly reconstructing camera and LiDAR observations with dynamic neural or Gaussian scene representations (Yang et al., 2023; Tonderski et al., 2024; Hess et al., 2025). They can reproduce images, LiDAR returns, and sensor-specific effects with high quality, but require costly optimization for every new scene. Feed-forward reconstruction takes the opposite path by amortizing reconstruction across scenes. Geometry foundation models such as VGGT, $\pi^3$, InfiniteVGGT and CasaMaestro (Wang et al., 2025a; 2026b; Yuan et al., 2026b; Ji et al., 2026) infer scene geometry directly from images, while DrivingForward, STORM, DGGT, ReconDrive, and SurroundNEXO adapt this paradigm to dynamic driving scenes and Gaussian representations (Tian et al., 2025; Yang et al., 2025a; Chen et al., 2025; Yu et al., 2026a; Yuan et al., 2026a). These methods greatly improve scalability, yet remain centered on visual geometry and novel-view image synthesis. The two lines have therefore addressed different sides of the problem. One provides multi-sensor fidelity without efficient reconstruction, while the other provides efficient reconstruction without multi-sensor simulation.

Bridging these two lines of research requires more than incorporating LiDAR as an additional input or supervision signal. The central challenge lies in grounding sparse and asynchronous LiDAR measurements into visually inferred dynamic Gaussians in a feed-forward manner. Feed-forward Gaussian reconstruction is driven primarily by dense image features, whereas LiDAR returns sample scene surfaces sparsely and at different spatial locations and acquisition times. Without explicit spatio-temporal correspondence across the two modalities, LiDAR evidence cannot be reliably associated with the predicted Gaussians and thus provides only weak constraints on their geometry. As a result, the reconstructed scene may appear visually plausible while lacking the metric accuracy required for faithful range simulation. This challenge also extends beyond geometry because LiDAR intensity is a sensor-specific surface attribute that cannot be inferred from visual appearance alone and is observed only sparsely across space and time. The key obstacle is therefore not simply the lack of a LiDAR rendering branch. Rather, it is the lack of a unified cross-modal representation that establishes spatio-temporal associations between complementary measurements, shares geometry across sensors, and preserves modality-specific attributes.

To realize such a unified representation, we propose **DriveR**$^3$, a feed-forward framework that reconstructs a dynamic Gaussian scene from sparse multi-view camera and LiDAR observations in a single forward pass. At its core, DriveR$^3$ grounds sparse and asynchronous LiDAR returns onto visually predicted Gaussians through *spatio-temporal LiDAR–Gaussian association*. By aggregating multi-frame LiDAR evidence according to spatial proximity and sampling-time difference, this association establishes explicit correspondences between LiDAR measurements and the Gaussian representation. These correspondences allow the two modalities to jointly learn *shared dynamic geometry through complementary supervision*. Dense image reconstruction provides spatially comprehensive constraints on scene structure and appearance, while LiDAR range, surface, and free-space supervision anchors the representation to metric geometry. Camera and LiDAR rendering consequently share Gaussian position, scale, rotation, and motion, while retaining modality-specific opacity and appearance attributes. The same cross-modal grounding also supports *feed-forward LiDAR intensity reconstruction*. Because intensity is observed only at sparse LiDAR returns, we propagate the available measurements through complementary spatio-temporal and nearest-neighbor associations, and adaptively fuse the resulting estimates based on measurement support, uncertainty, and matching quality. In this way, DriveR$^3$ progresses from cross-modal grounding to shared geometric reconstruction and sensor-specific attribute recovery, enabling efficient camera-LiDAR simulation at novel poses and times without per-scene optimization.

Experiments on unseen nuScenes scenes demonstrate that DriveR$^3$ extends joint camera-LiDAR scene reconstruction to feed-forward inference without per-scene optimization. Compared with SplatAD, DriveR$^3$ achieves an approximately $10\times$ speedup in scene reconstruction while obtaining lower errors on multiple LiDAR reconstruction metrics. Incorporating LiDAR observations also improves visual reconstruction quality, indicating that complementary sensor information benefits reconstruction within the shared scene representation.

Our main contributions are as follows: **(i)** We propose **DriveR$^3$**, the first feed-forward framework that reconstructs unified dynamic Gaussian scenes from multi-view temporal images and LiDAR observations, supporting joint camera-LiDAR rendering without per-scene optimization. **(ii)** We establish spatio-temporal LiDAR–Gaussian associations and learn shared dynamic geometry through complementary camera-LiDAR supervision, integrating multi-frame sensor evidence into a common representation for joint rendering. **(iii)** We introduce a feed-forward LiDAR intensity reconstruction method that adaptively fuses multiple estimates according to their supporting evidence, producing Gaussian-level intensity attributes for LiDAR rendering at unobserved times.

## 2 Related Work

**Feed-Forward Visual Reconstruction.** Feed-forward reconstruction learns cross-scene priors to infer geometry or renderable representations directly from images. General visual geometry models predict depth, camera poses, and correspondences from image collections (Wang et al., 2024; 2025a). Subsequent methods extend this paradigm through equivariant prediction, unified geometric representations, and model scaling (Lin et al., 2025a; Wang et al., 2026b;a), while query-based frameworks jointly reason about geometry and dynamics (Zhang et al., 2026a). Another line of work directly predicts Gaussian primitives for novel-view synthesis (Chen et al., 2024b; Xu et al., 2025; Ye et al., 2025). For driving scenes, feed-forward reconstruction has been extended to surround-view and temporal observations (Lin et al., 2025b; Tian et al., 2025; Yang et al., 2025a), under different input and supervision assumptions (Chen et al., 2025; Wang et al., 2025b; Yu et al., 2026a;b), using pixel-aligned, volumetric, scaffold-based, or point-aligned scene representations (Miao et al., 2025; Chi et al., 2026; Shi et al., 2026). Overall, existing feed-forward methods primarily focus on visual geometry recovery and novel-view image synthesis. Although some incorporate LiDAR as an additional input or geometric supervision, they do not treat LiDAR returns and intensities as reconstruction and rendering targets.

**Neural Reconstruction for LiDAR and Multi-Sensor Simulation.** Neural scene representations enable LiDAR re-simulation by modeling scene geometry together with sensor-specific measurements. Neural-field methods reconstruct static and dynamic LiDAR observations (Huang et al., 2023; Tao et al., 2024; Wu et al., 2024; Zheng et al., 2024), while Gaussian-based approaches use splatting or ray tracing for efficient LiDAR rendering (Chen et al., 2024a; Jiang et al., 2025; Zhou et al., 2025). Multi-sensor methods further reconstruct shared scenes that support both camera and LiDAR rendering (Yang et al., 2023; Tonderski et al., 2024). SplatAD brings joint camera-LiDAR rendering to 3D Gaussian Splatting and models sensor-specific effects such as LiDAR intensity and ray dropouts (Hess et al., 2025), while other approaches incorporate LiDAR supervision or address cross-sensor inconsistencies (Kung et al., 2024; Turki et al., 2026). Despite their sensor fidelity and rendering efficiency, these methods optimize a separate representation for each scene. More recently, Instant NuRec explores feed-forward range and intensity reconstruction through a dedicated LiDAR branch (Huang et al., 2026). In contrast, DriveR$^3$ learns shared Gaussian geometry through complementary camera-LiDAR supervision and reconstructs Gaussian-level LiDAR intensity from multiple sources, enabling joint camera-LiDAR rendering without per-scene optimization.

## 3 Method

DriveR$^3$ reconstructs a unified dynamic Gaussian scene from camera and LiDAR context observations in a single forward pass, enabling joint rendering at novel poses and times. As shown in Figure 2, it grounds sparse, asynchronous LiDAR measurements onto image-aligned Gaussians through spatiotemporal association and predicts Gaussian-level intensity through adaptive multi-source fusion. Shared geometry is learned through complementary camera-LiDAR supervision.

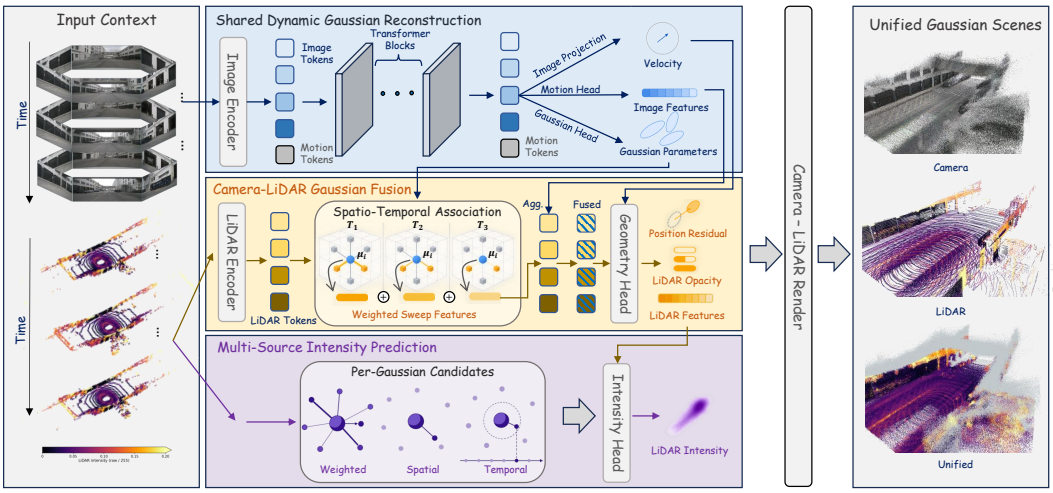


Figure 2: **Overview of DriveR³.** Shared Dynamic Gaussian Reconstruction builds a unified scene from camera and LiDAR context observations. Spatio-Temporal Camera-LiDAR Association grounds sparse, asynchronous LiDAR measurements onto image-aligned Gaussians, and Multi-Source LiDAR Intensity Prediction estimates Gaussian-level intensity from complementary evidence. The representation is trained in two stages using camera-LiDAR geometry supervision followed by intensity supervision.

### 3.1 Problem Formulation

Given camera observations $\mathcal{C}^{\text{cam}} = \{(I_n^v, \mathbf{K}_n^v, \mathbf{T}_n^v, t_n^v)\}_{n,v}$ and LiDAR sweeps $\mathcal{C}^{\text{lidar}} = \{(\mathcal{P}_m, \mathbf{T}_m^L, \tau_m)\}_m$, we learn a feed-forward mapping $\mathcal{G} = F_\theta(\mathcal{C}^{\text{cam}}, \mathcal{C}^{\text{lidar}})$ that predicts a dynamic Gaussian scene without per-scene optimization. Here, $I_n^v$, $\mathbf{K}_n^v$, $\mathbf{T}_n^v$, and $t_n^v$ denote the image, intrinsics, calibrated pose, and acquisition time of camera $v$ at context timestep $n$. Each sweep $\mathcal{P}_m = \{(\mathbf{p}_{mj}, \ell_{mj})\}_j$ contains return positions and intensities, with calibrated pose $\mathbf{T}_m^L$ and acquisition time $\tau_m$. All observations are aligned in a common metric coordinate system using sensor calibration and ego poses.

Images provide dense appearance cues, while LiDAR supplies metric surface constraints. We use the image-aligned Gaussians predicted by STORM (Yang et al., 2025a) as a shared scaffold for both sensors. Gaussian $i$ is initialized at $\boldsymbol{\mu}_i^0 = \mathbf{o}_i + d_i \mathbf{d}_i$, where $\mathbf{o}_i$ is the source camera origin, $d_i$ is the predicted camera-space depth, and $\mathbf{d}_i$ is the unnormalized back-projection direction expressed in the common coordinate frame. Each primitive $g_i$ contains a corrected center $\boldsymbol{\mu}_i$, scale $\mathbf{s}_i$, rotation quaternion $\mathbf{q}_i$, velocity $\mathbf{v}_i$, source time $t_i$, camera color $\mathbf{c}_i$, and sensor-specific opacities $o_i^C$ and $o_i^L$. Its covariance $\boldsymbol{\Sigma}_i$ is determined by scale and rotation following standard 3D Gaussian Splatting (Kerbl et al., 2023). Within each short window, we translate the primitives to query time $t$:

$$\boldsymbol{\mu}_i(t) = \boldsymbol{\mu}_i + \mathbf{v}_i(t - t_i), \qquad \mathcal{G}(t) = \{g_i(t)\}_i. \tag{1}$$

Here, $g_i(t)$ denotes the translated primitive. Camera and LiDAR rendering share the centers, covariances, and motions while retaining modality-specific opacity and appearance.

### 3.2 Spatio-Temporal LiDAR–Gaussian Association

LiDAR returns near a visual Gaussian may be temporally misaligned, especially on moving objects. We therefore aggregate context sweeps using spatial and temporal weights. A lightweight convolutional encoder extracts range-image features, which are averaged with positions and intensities within each occupied voxel of each sweep. Each entry $(\bar{\mathbf{p}}_u, \bar{\mathbf{h}}_u, \bar{\ell}_u, \tau_u)$ is indexed by a voxel–sweep pair $u$ and contains the centroid, mean feature, mean intensity, and acquisition time.

For a query position $\mathbf{x}$, let $\mathcal{N}(\mathbf{x})$ contain the indices of locally enumerated occupied voxel entries whose centroids lie within association radius $r$. We weight each entry by its spatial distance from $\mathbf{x}$ and its temporal difference from the source time $t_i$ of Gaussian $i$:

$$w_{iu}(\mathbf{x}) = \mathbf{1}[u \in \mathcal{N}(\mathbf{x})] \exp\left(-\frac{\|\mathbf{x} - \bar{\mathbf{p}}_u\|_2^2}{2\sigma_s^2} - \frac{(t_i - \tau_u)^2}{2\sigma_t^2}\right), \tag{2}$$

where $\mathbf{1}[\cdot]$ is the indicator function, and $\sigma_s$ and $\sigma_t$ are the spatial and temporal bandwidths. To refine geometry, we use these weights at the initial Gaussian center $\boldsymbol{\mu}_i^0$ to aggregate LiDAR features and compute observation support. With $W_i = \sum_u w_{iu}(\boldsymbol{\mu}_i^0)$, we obtain

$$\bar{\mathbf{h}}_i = \frac{\sum_u w_{iu}(\boldsymbol{\mu}_i^0)\bar{\mathbf{h}}_u}{\max(W_i, \epsilon)}, \qquad s_i = 1 - \exp(-W_i). \tag{3}$$

Here, $\bar{\mathbf{h}}_i$ is the aggregated LiDAR feature, and $s_i \in [0, 1)$ measures the support provided by the associated observations. The support vanishes without LiDAR evidence, and $\epsilon > 0$ ensures numerical stability.

We fuse the aggregated LiDAR feature with the projected, pixel-aligned image feature $\mathbf{f}_i^{\text{img}} \in \mathbb{R}^{24}$, giving $\mathbf{z}_i = \bar{\mathbf{h}}_i + \mathbf{f}_i^{\text{img}}$. The fused feature, initial Gaussian state, support $s_i$, and nearest-match distance form a descriptor $\mathbf{u}_i$, from which the center-correction head $H_\mu$ predicts a bounded correction:

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}_i^0 + \rho_\mu s_i \tanh H_\mu(\mathbf{u}_i). \tag{4}$$

Here, $\rho_\mu$ controls the correction magnitude, and the correction vanishes without LiDAR support. The same descriptor is also used to predict a LiDAR opacity residual and a feature $\mathbf{f}_i^L$ for intensity reconstruction. Scale, rotation, and motion retain the visual-head parameterization and are optimized through shared rendering supervision in Section 3.4. Details of LiDAR encoding, descriptor construction, and prediction heads are provided in Appendix A.1.

### 3.3 Reconstructing LiDAR Intensity from Multi-Source Evidence

Geometry determines which Gaussians contribute to a LiDAR ray, while sensor-dependent return intensity cannot be reliably inferred from RGB alone. We therefore estimate Gaussian-level intensity $\hat{\ell}_i \in [0, 1]$ from context LiDAR returns, keeping geometry and opacity fixed during intensity training. Measured intensities are normalized to the same $[0, 1]$ range.

**Complementary intensity candidates.** For the corrected Gaussian center $\boldsymbol{\mu}_i$, the spatiotemporally weighted candidate aggregates repeated local observations:

$$b_i^w = \frac{\sum_u w_{iu}(\boldsymbol{\mu}_i)\bar{\ell}_u}{\max(\sum_u w_{iu}(\boldsymbol{\mu}_i), \epsilon)}. \tag{5}$$

To preserve local information from individual measurements, the spatial candidate $b_i^s$ takes the intensity of the closest raw return within radius $r$ across all context sweeps. The temporal candidate $b_i^t$ first selects the sweep closest to $t_i$ among those containing a return within the same radius, then takes the intensity of its closest return.

The candidate values, variances, supports, match counts, nearest distances, and representative times are fed into an MLP together with the frozen feature $\mathbf{f}_i^L$ to predict fusion logits $z_i^a$ for $a \in \{w, s, t\}$ and an intensity residual $r_i^{\text{int}}$. We retain the weighted candidate and mask unsupported nearest-neighbor candidates, defining the active set as $\mathcal{V}_i = \{w\} \cup \{a \in \{s, t\} : s_i^a > 0\}$. We then fuse the candidates and apply a bounded residual correction:

$$\omega_i^a = \begin{cases} \dfrac{\exp(z_i^a)}{\sum_{a' \in \mathcal{V}_i} \exp(z_i^{a'})}, & a \in \mathcal{V}_i, \\ 0, & a \notin \mathcal{V}_i, \end{cases} \tag{6}$$

$$b_i^{\text{mix}} = \sum_a \omega_i^a b_i^a, \qquad b_i^+ = C\left(b_i^{\text{mix}} + \rho_{\text{int}} \tanh r_i^{\text{int}}\right).$$

Here, $\rho_{\text{int}}$ bounds the residual magnitude, and $C$ clips to $[0, 1]$. A zero-initialized learnable gate introduces the fused correction around $b_i^w$, producing the final Gaussian intensity $\hat{\ell}_i$ while starting prediction from the weighted candidate. LiDAR rendering combines these intensities using compositing weights and observation support. The candidate statistics, gating formulation, and intensity rendering readout are detailed in Appendix A.2.

### 3.4 Learning Shared Dynamic Geometry from Complementary Supervision

DriveR$^3$ first learns shared dynamic geometry through complementary camera and LiDAR supervision, then trains the intensity branch with geometry and opacity fixed. This two-stage strategy lets range supervision shape scene surfaces while preventing intensity optimization from perturbing the geometry. In Stage I, we freeze the visual encoder and jointly train the Gaussian, motion, sky, and affine heads, together with the LiDAR encoder, association module, and residual heads. The two modalities jointly supervise the shared geometry and motion. To accommodate their different visibility responses, we retain camera opacity $o_i^C$ and predict a LiDAR logit residual $\delta_i = H_o(\mathbf{u}_i)$ to obtain $o_i^L$. We center the residual within each context group and regularize the two opacity fields through a distance-stratified anchor loss $\mathcal{L}_{\text{anchor}}$.

Camera rendering uses Gaussian splatting with STORM's sky and camera-affine heads, while LiDAR rendering uses SplatAD's differentiable spherical rasterizer (Hess et al., 2025). A differentiable opacity-crossing readout $\hat{r}_k^{\text{soft}}$ provides the LiDAR range estimates used for geometry supervision. Range supervision fits measured surfaces and penalizes missing predictions on a fixed set $\Omega_0$ of initially supported returns, while line-of-sight supervision suppresses opacity in observed free space. Signed range differences between neighboring rays are constrained by the smooth-surface and depth-edge losses $\mathcal{L}_{\text{surface}}$ and $\mathcal{L}_{\text{edge}}$. The readout loss $\mathcal{L}_{\text{readout}}$ aligns the soft range with a detached hard opacity-median range readout, while camera loss $\mathcal{L}_{\text{cam}}$ combines MSE, LPIPS (Zhang et al., 2018), and SSIM (Wang et al., 2004). Together, the Stage I objective is

$$\begin{aligned}\mathcal{L}_{\text{geo}} = \lambda_r \mathcal{L}_{\text{range}} + \lambda_{\text{surf}} \mathcal{L}_{\text{surface}} + \lambda_e \mathcal{L}_{\text{edge}} + \lambda_{\text{rd}} \mathcal{L}_{\text{readout}} \\ + \lambda_c \mathcal{L}_{\text{cam}} + \lambda_a \mathcal{L}_{\text{anchor}} + \lambda_{\text{los}} \mathcal{L}_{\text{los}},\end{aligned} \tag{7}$$

where the $\lambda$ coefficients weight the respective losses.

**Multi-source LiDAR intensity learning.** Stage II optimizes only the multi-source intensity branch, with geometry, motion, and both opacity networks frozen. For each Gaussian, the weighted candidate provides an intensity estimate from nearby context returns. To form a reference for training, we render this candidate using its own observation support, rather than the fused intensity and support. The target-return error evaluates the final prediction, while regularization toward the weighted reference keeps the learned correction tied to measured evidence. We normalize the target error by the fixed training-set error of a constant-mean predictor, so its scale does not depend on the current batch. Details of the opacity parameterization, rendering readout, loss definitions, and training-set normalization are provided in Sections A.3 and A.4 of the appendix.

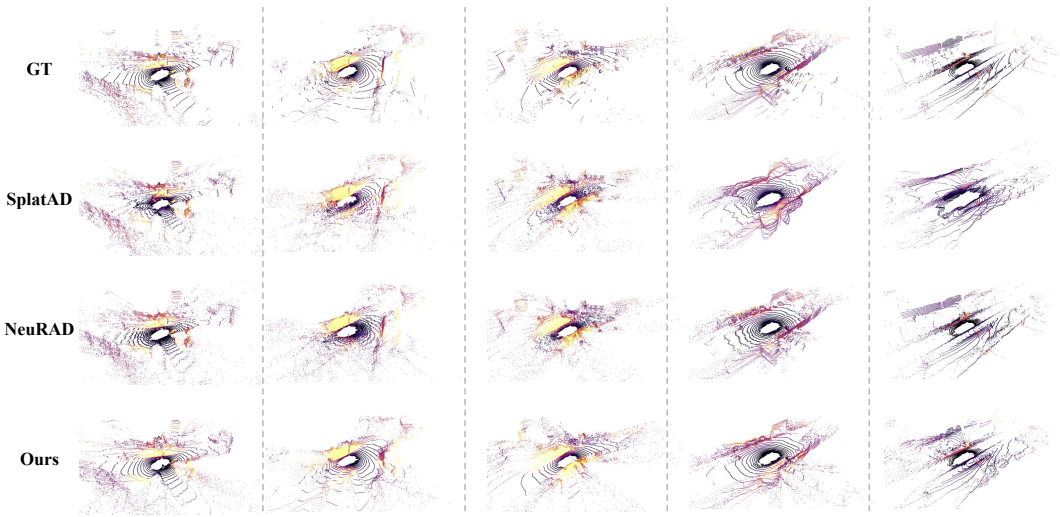


Figure 3: Qualitative comparison of LiDAR reconstruction on nuScenes.

# 4 RESULTS

## 4.1 EXPERIMENTAL SETUP

**Implementation details.** We use STORM-L/16 with $288 \times 512$ input images and 24-channel image and LiDAR features for spatiotemporal association. Spatiotemporal association uses a voxel size of $0.5\,\mathrm{m}$, a neighborhood radius of $1\,\mathrm{m}$, and spatial and temporal kernel bandwidths of $0.5\,\mathrm{m}$ and $0.6\,\mathrm{s}$, respectively. Geometry is trained for five epochs with the visual encoder frozen. Intensity fusion is then trained for one epoch with geometry, motion, and both opacity networks fixed.

**Dataset.** We conduct experiments on nuScenes using the official split of 700 training and 150 validation scenes. Each 20-frame window contains ten context timesteps and ten held-out target timesteps. Each context timestep provides six surround-view images and one top-mounted LiDAR scan. We evaluate reconstruction at the held-out target timesteps in unseen scenes, with the main quality comparison conducted on 50 scenes randomly sampled from the 150-scene validation split.

**Metrics.** We evaluate camera reconstruction using PSNR, SSIM, and LPIPS, and LiDAR reconstruction using Depth L2, range MAE/RMSE, intensity RMSE, and squared Chamfer distance (CD). Intensity is normalized by dividing raw values by 255. Main-table quality scores are first averaged within each scene and then equally across scenes. We evaluate reconstruction efficiency using scene construction time and speedup. Details of LiDAR evaluation and reconstruction-time measurement are provided in Sections B.2 and B.3 of the appendix.

## 4.2 COMPARISON WITH EXISTING METHODS

**Reconstruction quality.** Table 1 compares the camera and LiDAR reconstruction quality of DriveR$^3$ with existing methods. We additionally include PVG (Chen et al., 2026) and DeformableGS (Yang et al., 2024) as scene-specific visual reconstruction baselines. While extending feed-forward reconstruction to LiDAR range and intensity, DriveR$^3$ improves all three camera metrics over the visual feed-forward baseline STORM (Yang et al., 2025a), including a PSNR gain of $0.82\,\mathrm{dB}$. These results show that the joint framework can expand sensor outputs while improving image quality, consistent with learning shared geometry from complementary observations.

Compared with the per-scene optimization method SplatAD (Hess et al., 2025), DriveR$^3$ achieves lower errors on all five LiDAR metrics without test-time optimization. In particular, CD decreases by approximately $43.1\%$, indicating better agreement between the reconstructed and measured point-cloud geometry. Although camera PSNR and SSIM remain below SplatAD, DriveR$^3$ achieves competitive LiDAR reconstruction quality through feed-forward prediction while supporting both sensor modalities within the same dynamic scene representation.

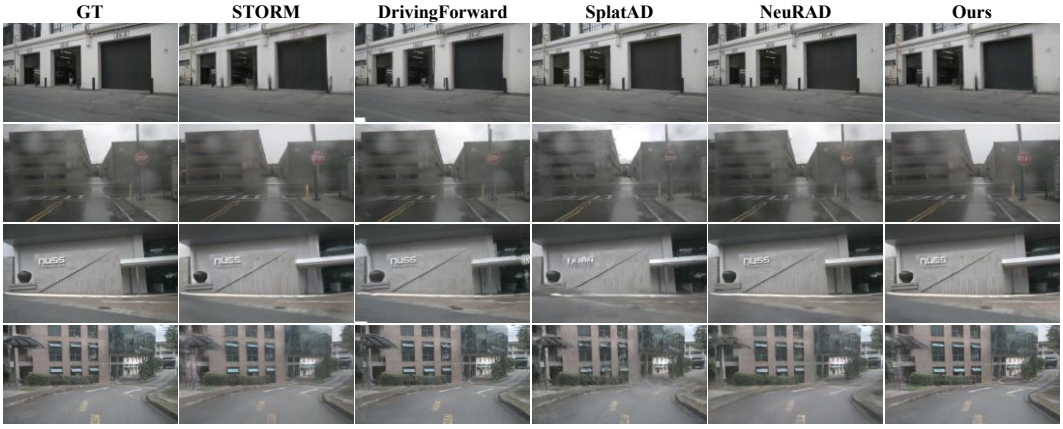


Figure 4: Qualitative comparison of RGB on nuScenes.

Table 1: Camera-LiDAR reconstruction on nuScenes Caesar et al. (2020).

| Method | FFD | Joint | Camera | | | LiDAR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | PSNR ↑ | SSIM ↑ | LPIPS ↓ | $L2_{Depth}$ ↓ | $MAE_{Range}$ ↓ | $RMSE_{Range}$ ↓ | $RMSE_{Intensity}$ ↓ | CD ↓ |
| PVG | ✗ | ✗ | 20.36 | 0.669 | 0.469 | — | — | — | — | — |
| DeformableGS | ✗ | ✗ | 16.47 | 0.548 | 0.542 | — | — | — | — | — |
| NeuRAD | ✗ | ✓ | **25.32** | 0.742 | 0.180 | 0.09 | **1.02** | 4.35 | 0.0493 | 12.83 |
| SplatAD | ✗ | ✓ | 25.24 | **0.783** | 0.293 | 1.10 | 1.51 | 4.58 | 0.0488 | 5.11 |
| STORM | ✓ | ✗ | 21.86 | 0.665 | 0.320 | — | — | — | — | — |
| ReconDrive | ✓ | ✗ | 23.49 | 0.755 | 0.147 | — | — | — | — | — |
| DrivingForward | ✓ | ✗ | 24.53 | 0.754 | **0.137** | — | — | — | — | — |
| **DriveR$^3$(Ours)** | ✓ | ✓ | 22.68 | 0.682 | 0.247 | **0.09** | 1.47 | **4.05** | **0.0467** | **2.91** |

Table 2: Reconstruction efficiency Results. † indicates optimization time excluding initialization, normalized by window count.

| Method | Test-time Optimization | Steps ↓ | Reconstruction | |
|---|---|---|---|---|
| | | | Build Time ↓ (s/window) | Speedup ↑ |
| NeuRAD† | Yes | 20k | 430.07 | 0.55× |
| SplatAD† | Yes | 30k | 235.54 | 1.0× |
| **DriveR$^3$(Ours)** | **No** | **0** | **23.52** | **10.02×** |

**Reconstruction efficiency.** Table 2 compares scene construction efficiency on five validation scenes using RTX 4090 GPUs. DriveR$^3$ replaces per-scene iterative optimization with feed-forward prediction, requiring an average of 23.52 seconds to construct each window. After normalizing scene-optimization time by window count, DriveR$^3$ achieves speedups of $10.02\times$ over SplatAD at 30k optimization steps and $18.29\times$ over NeuRAD (Tonderski et al., 2024) at 20k steps. These results show that feed-forward joint reconstruction reduces the cost of constructing new scenes, allowing driving observations to be converted more efficiently into unified representations for camera and LiDAR rendering.

### 4.3 Ablation Studies

We evaluate the contributions of our components on 20 randomly sampled validation scenes. Association and opacity ablations are performed at inference time using the same trained model. Intensity ablations keep geometry, opacity, and rendering support fixed.

**Spatiotemporal association.** Table 3 evaluates the effect of associating contextual LiDAR observations with visual Gaussians. This association supplies metric geometric evidence, improving Depth L2, range MAE/RMSE, and CD. In particular, range RMSE decreases from 5.72 to 4.91 m. These results support the use of sparse LiDAR observations to improve the shared geometry through spatiotemporal association.

**Sensor-specific opacity.** Sharing camera opacity degrades LiDAR reconstruction, whereas sharing LiDAR opacity reduces camera quality (Table 3). Using sensor-specific opacity reduces Depth L2 from 0.64 to 0.23 $m^2$ relative to shared camera opacity and improves PSNR by 2.07 dB relative to shared LiDAR opacity. These complementary effects support sharing geometry and motion while retaining distinct opacity fields to accommodate the response of each sensor.

**Multi-source intensity estimation.** Table 4 compares the training-set mean predictor, weighted estimation, and our full intensity estimation pipeline. Weighted estimation substantially outperforms the mean predictor, showing that contextual LiDAR observations provide useful evidence for intensity reconstruction. Adaptive multi-source fusion and residual refinement further reduce intensity RMSE from 0.053 to 0.050. Overall, the full pipeline reduces RMSE by 32.79% relative to the mean predictor, supporting its effectiveness in recovering return intensity from contextual observations.

Table 3: Ablation study on association and opacity.

| Variant | Camera | | | LiDAR | | | |
|---|---|---|---|---|---|---|---|
| | PSNR ↑ | SSIM ↑ | LPIPS ↓ | $L2_{Depth}$ ↓ | $MAE_{Range}$ ↓ | $RMSE_{Range}$ ↓ | CD ↓ |
| Without association evidence | 22.99 | 0.683 | 0.252 | 1.15 | 2.79 | 5.72 | 5.01 |
| Shared camera opacity | 22.92 | 0.678 | 0.254 | 0.64 | 2.29 | 5.27 | 4.96 |
| Shared LiDAR opacity | 20.85 | 0.595 | 0.384 | 0.23 | 1.96 | 4.91 | 4.11 |
| **DriveR$^3$(Ours)** | 22.92 | 0.678 | 0.254 | 0.23 | 1.96 | 4.91 | 4.11 |

Table 4: Ablation study on intensity accuracy.

| Variant | LiDAR intensity | | |
|---|---|---|---|
| | MAE ↓ | RMSE ↓ | RMSE reduction (%) ↑ |
| Training-set mean | 0.055 | 0.075 | — |
| Weighted estimate | 0.032 | 0.053 | 28.71 |
| **DriveR$^3$(Ours)** | 0.029 | 0.050 | 32.79 |

### 4.4 Further Discussion

Our experiments show that DriveR$^3$ achieves efficient joint camera-LiDAR reconstruction without per-scene optimization, while maintaining competitive LiDAR reconstruction quality. There remains scope to improve image PSNR and SSIM relative to NeuRAD and SplatAD. One possible explanation is that per-scene optimization can fit scene-specific appearance more closely through dense image supervision.

During testing, we observed occasional collapsed reconstructions from per-scene optimization, including in nighttime scenes. Weak image cues and sparse LiDAR returns can leave the geometry of an individual scene insufficiently constrained. DriveR$^3$ instead learns geometric priors across scenes and predicts a representation without fitting each test scene anew. In these difficult cases, it produced more stable reconstructions than the per-scene methods in our qualitative observations.

The visual quality of the current method is still constrained by the STORM backbone. Although ReconDrive achieves better visual reconstruction, its LiDAR reconstruction quality remains limited. Given the limited availability of stronger, suitable visual backbones, we will incorporate such models as they become openly available to further improve joint camera-LiDAR reconstruction quality.

## 5 Conclusion

We presented DriveR$^3$, a feed-forward framework that learns across scenes to reconstruct a unified dynamic scene directly from sparse camera and LiDAR observations. Its central design grounds sparse, asynchronous LiDAR returns onto image-aligned dynamic Gaussians through spatio-temporal association. This grounding enables image and metric LiDAR supervision to jointly shape the same geometry, while sensor-specific opacity preserves each sensor's response characteristics. Our multi-source intensity estimation also propagates sparse returns to Gaussian-level attributes, allowing the representation to synthesize both LiDAR range and intensity. Once trained, DriveR$^3$ converts observations from an unseen scene into a representation shared by both sensor renderers in a single forward pass, eliminating repeated per-scene optimization.

Experiments on nuScenes demonstrate the resulting balance between reconstruction quality and efficiency. Compared with the visual feed-forward baseline STORM, DriveR$^3$ improves all evaluated image metrics, including a $0.82\,\mathrm{dB}$ PSNR gain, while extending feed-forward reconstruction to LiDAR. Compared with the per-scene joint reconstruction method SplatAD, it achieves lower errors on multiple LiDAR metrics, including an $43.1\%$ reduction in squared Chamfer distance, and constructs scenes substantially faster without test-time optimization. Nevertheless, camera quality and range MAE still trail the strongest optimization-based baseline, leaving room to improve visual priors and fine geometric detail. Future work will therefore explore stronger suitable visual backbones as they become openly available, as well as richer cross-modal associations, to further improve the fidelity and generalization of joint camera–LiDAR reconstruction.

### Reproducibility Statement

The main paper defines DriveR$^3$, including its unified dynamic Gaussian scene representation, spatiotemporal LiDAR–Gaussian association, multi-source intensity estimation, sensor-specific opacity, and two-stage training objectives. The supplementary material further describes the observation encoding and coordinate and time conventions, association and intensity-rendering details, training and input configurations, data splits and target-frame construction, metric definitions and aggregation procedures, and the protocol for measuring scene-reconstruction time. The experimental section reports the evaluation settings used for the main comparisons and ablation studies, thereby supporting reproduction of the reported results.

### Ethics Statement

This work uses only the publicly available nuScenes dataset and involves no additional recruitment of human participants or collection of personal data by the authors. The use of the data, the release of derived assets, and any redistribution of the original data must comply with the nuScenes license and terms of use. Because nuScenes primarily covers particular road, environmental, and sensor conditions in Boston and Singapore, our results should not be interpreted as establishing equal applicability across all geographic regions, weather conditions, traffic participants, or sensor configurations. DriveR$^3$ is intended for research on multi-sensor scene reconstruction and simulation; its simulated outputs should not replace real-world testing, system-level safety validation, or safety certification for autonomous-driving systems.

### AI Use Statement

In this work, we used generative AI tools to assist with discussions of research methodology and experimental design; the implementation and debugging of method, data-preprocessing, and evaluation code; the analysis and interpretation of experimental results; literature search and organization; and the translation of Chinese drafts into English, manuscript organization, language editing, and LaTeX formatting. The authors reviewed all AI-assisted content. The relevant code and experimental conclusions were verified through actual execution, testing, and manual inspection of configurations, logs, and evaluation metrics, and all citations were checked by the authors. Generative AI was not used to fabricate or alter experimental data or results. All final scientific decisions were made by the authors, who take responsibility for the entire content of the paper, including its scientific claims, equations, citations, code, experimental results, and conclusions.

## References

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. URL https://arxiv.org/abs/1308.3432.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11621–11631, 2020.

Qifeng Chen, Sheng Yang, Sicong Du, Tao Tang, Rengan Xie, Peng Chen, and Yuchi Huo. LiDAR-GS: Real-time LiDAR re-simulation using gaussian splatting. *arXiv preprint arXiv:2410.05111*, 2024a. URL https://arxiv.org/abs/2410.05111.

Xiaoxue Chen, Ziyi Xiong, Yuantao Chen, Gen Li, Nan Wang, Hongcheng Luo, Long Chen, Haiyang Sun, Bing Wang, Guang Chen, Hangjun Ye, Hongyang Li, Ya-Qin Zhang, and Hao Zhao. DGGT: Feedforward 4D reconstruction of dynamic driving scenes using unposed images. *arXiv preprint arXiv:2512.03004*, 2025. URL https://arxiv.org/abs/2512.03004.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D Gaussian Splatting from sparse multi-view images. In *European Conference on Computer Vision*, 2024b. URL https://arxiv.org/abs/2403.14627.

Yurui Chen, Chun Gu, Junzhe Jiang, Xiatian Zhu, and Li Zhang. Periodic Vibration Gaussian: Dynamic urban scene reconstruction and real-time rendering. *International Journal of Computer Vision*, 2026.

Cheng Chi, Xianqi Wang, Hongcheng Luo, Mingfei Tu, Gangwei Xu, Zehan Zhang, Bing Wang, Guang Chen, Hangjun Ye, Sida Peng, Xin Yang, and Haiyang Sun. PointForward: Feedforward driving reconstruction through point-aligned representations. *arXiv preprint arXiv:2605.11594*, 2026. URL https://arxiv.org/abs/2605.11594.

Mingzhe Guo, Zhipeng Zhang, Yuan He, Ke Wang, and Liping Jing. End-to-end autonomous driving without costly modularization and 3d manual annotation, 2024. URL https://arxiv.org/abs/2406.17680.

Mingzhe Guo, Yixiang Yang, Chuanrong Han, Rufeng Zhang, Shirui Li, Ji Wan, and Zhipeng Zhang. Flowad: Ego-scene interactive modeling for autonomous driving, 2026. URL https://arxiv.org/abs/2603.13399.

Georg Hess, Carl Lindström, Maryam Fatemi, Christoffer Petersson, and Lennart Svensson. SplatAD: Real-time LiDAR and camera rendering with 3D Gaussian Splatting for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2411.16816.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Jiahui Huang, Jiawei Ren, Michal Tyszkiewicz, Bjoern Haefner, Michael Shelley, Xin Kang, Seung Wook Kim, Ning Xu, Qi Wu, Janick Martinez Esturo, Shengyu Huang, Nick Schneider, Laura Leal-Taixé, Zan Gojcic, and Sanja Fidler. Instant NuRec: Feed-forward 3D Gaussian reconstruction for driving scene simulation. *arXiv preprint arXiv:2607.14203*, 2026. URL https://arxiv.org/abs/2607.14203.

Shengyu Huang, Zan Gojcic, Zian Wang, Francis Williams, Yoni Kasten, Sanja Fidler, Konrad Schindler, and Or Litany. Neural LiDAR fields for novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18236–18246, 2023. URL https://arxiv.org/abs/2305.01643.

Yuzhou Ji, Xiaotian Yang, and Zhipeng Zhang. Casamaestro: Multi-view panoramas for house-scale 3d reconstruction. *arXiv preprint arXiv:2606.31086*, 2026.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. *ICCV*, 2023.

Junzhe Jiang, Chun Gu, Yurui Chen, and Li Zhang. GS-LiDAR: Generating realistic LiDAR point clouds with panoramic gaussian splatting. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2501.13971.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. doi: 10.1145/3592433. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

Pou-Chun Kung, Xianling Zhang, Katherine A. Skinner, and Nikita Jaipuria. LiHi-GS: LiDAR-supervised gaussian splatting for highway driving scene reconstruction. *arXiv preprint arXiv:2412.15447*, 2024. URL https://arxiv.org/abs/2412.15447.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025a. URL https://arxiv.org/abs/2511.10647.

Junhong Lin, Kangli Wang, Shunzhou Wang, Songlin Fan, Ge Li, and Wei Gao. VGD: Visual geometry gaussian splatting for feed-forward surround-view driving reconstruction. *arXiv preprint arXiv:2510.19578*, 2025b. URL https://arxiv.org/abs/2510.19578.

Runhao Mao, Hanshi Wang, Yixiang Yang, Qianli Ma, Jingmeng Zhou, and Zhipeng Zhang. The blind spot of adaptation: Quantifying and mitigating forgetting in fine-tuned driving models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Sheng Miao, Jiaxin Huang, Dongfeng Bai, Xu Yan, Hongyu Zhou, Yue Wang, Bingbing Liu, Andreas Geiger, and Yiyi Liao. EVolSplat: Efficient volume-based gaussian splatting for urban view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2503.20168.

Chen Shi, Shaoshuai Shi, Xiaoyang Lyu, Chunyang Liu, Kehua Sheng, Bo Zhang, and Li Jiang. UniSplat: Unified spatio-temporal fusion via 3D latent scaffolds for dynamic driving scene reconstruction. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.04595.

Tang Tao, Longfei Gao, Guangrun Wang, Yixing Lao, Peng Chen, Hengshuang Zhao, Dayang Hao, Xiaodan Liang, Mathieu Salzmann, and Kaicheng Yu. LiDAR-NeRF: Novel LiDAR view synthesis via neural radiance fields. In *Proceedings of the ACM International Conference on Multimedia*, 2024. doi: 10.1145/3664647.3681482. URL https://arxiv.org/abs/2304.10406.

Qijian Tian, Xin Tan, Yuan Xie, and Lizhuang Ma. DrivingForward: Feed-forward 3D Gaussian Splatting for driving scene reconstruction from flexible surround-view input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. URL https://arxiv.org/abs/2409.12753.

Adam Tonderski, Carl Lindström, Georg Hess, William Ljungbergh, Lennart Svensson, and Christoffer Petersson. NeuRAD: Neural rendering for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2311.15260.

Haithem Turki, Qi Wu, Xin Kang, Janick Martinez Esturo, Shengyu Huang, Ruilong Li, Zan Gojcic, and Riccardo de Lutio. SimULi: Real-time LiDAR and camera simulation with unscented transforms. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.12901.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a. URL https://arxiv.org/abs/2503.11651.

Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-$\Omega$. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a. URL https://arxiv.org/abs/2605.15195.

Jingkang Wang, Henry Che, Yun Chen, Ze Yang, Lily Goli, Sivabalan Manivasagam, and Raquel Urtasun. Flux4D: Flow-based unsupervised 4D reconstruction. In *Advances in Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=FeUGQ6AiKR.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2312.14132.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. $\pi^3$: Permutation-equivariant visual geometry learning. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://arxiv.org/abs/2507.13347.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861. URL https://doi.org/10.1109/TIP.2003.819861.

Hanfeng Wu, Xingxing Zuo, Stefan Leutenegger, Or Litany, Konrad Schindler, and Shengyu Huang. Dynamic LiDAR re-simulation using compositional neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2312.05247.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting gaussian splatting and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2410.13862.

Jiawei Yang, Jiahui Huang, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Maximilian Igl, Apoorva Sharma, Peter Karkus, Danfei Xu, Boris Ivanovic, Yue Wang, and Marco Pavone. STORM: Spatio-temporal reconstruction model for large-scale outdoor scenes. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://arxiv.org/abs/2501.00602.

Yixiang Yang, Chuanrong Han, Runhao Mao, et al. Survey of general end-to-end autonomous driving: A unified perspective. *TechRxiv*, December 2025b. doi: 10.36227/techrxiv.176523315.56439138/v1. URL https://doi.org/10.36227/techrxiv.176523315.56439138/v1.

Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. UniSim: A neural closed-loop sensor simulator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. URL https://arxiv.org/abs/2308.01898.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3D Gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20331–20341, 2024.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D Gaussian Splats from sparse unposed images. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.24207.

Haibao Yu, Kuntao Xiao, Jiahang Wang, Ruiyang Hao, Yuxin Huang, Guoran Hu, Haifang Qin, Bowen Jing, Yuntian Bo, and Ping Luo. ReconDrive: Fast feed-forward 4D Gaussian Splatting for autonomous driving scene reconstruction. *arXiv preprint arXiv:2603.07552*, 2026a. URL https://arxiv.org/abs/2603.07552.

Zhongrui Yu, Zhao Wang, Yijia Xie, Yida Wang, Xueyang Zhang, Yifei Zhan, and Kun Zhan. StreetForward: Perceiving dynamic street with feedforward causal attention. *arXiv preprint arXiv:2603.19552*, 2026b. URL https://arxiv.org/abs/2603.19552.

Shuai Yuan, Runxi Tang, Yuzhou Ji, Fudong Ge, Hanshi Wang, Yifei Wang, Xianming Zeng, Jianyun Xu, Xingliang Liu, Yanfeng Wang, and Zhipeng Zhang. SurroundNEXO: Egocentric metric bridging for spatially consistent geometry in autonomous driving. *arXiv preprint arXiv:2606.16960*, 2026a. URL https://arxiv.org/abs/2606.16960.

Shuai Yuan, Yantai Yang, Xiaotian Yang, Xupeng Zhang, Zhonghao Zhao, Lingming Zhang, and Zhipeng Zhang. Infinitevggt: Visual geometry grounded transformer for endless streams, 2026b.

Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K. Barral, Raia Hadsell, Zoubin Ghahramani, Andrew Zisserman, Junlin Zhang, and Mehdi S. M. Sajjadi. Efficiently reconstructing dynamic scenes one D4RT at a time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a. URL https://arxiv.org/abs/2512.08924.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

Yiwei Zhang, Xuesong Chen, Jin Gao, Hanshi Wang, Fudong Ge, Weiming Hu, Shaoshuai Shi, and Zhipeng Zhang. Onedrive: Unified multi-paradigm driving with vision-language-action models, 2026b. URL https://arxiv.org/abs/2604.17915.

Zehan Zheng, Fan Lu, Weiyi Xue, Guang Chen, and Changjun 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*, pp. 5145–5154, 2024. URL https://arxiv.org/abs/2404.02742.

Chenxu Zhou, Lvchang Fu, Sida Peng, Yunzhi Yan, Zhanhua Zhang, Yong Chen, Jiazhi Xia, and Xiaowei Zhou. LiDAR-RT: Gaussian-based ray tracing for dynamic LiDAR re-simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2412.15199.

# Supplementary Material

## DriveR$^3$: Feed-Forward Gaussian Reconstruction for Camera-LiDAR Simulation in Autonomous Driving

In this supplementary material, we provide the implementation, training, and evaluation details referenced in the main paper. The contents are organized as follows:

- **Section A (Method Implementation and Training Details).** We describe observation encoding and association, intensity fusion and rendering, sensor-specific opacity, and the training objectives.
- **Section B (Evaluation Details).** We specify context construction, metric computation, and reconstruction-time measurement.

## A Method Implementation and Training Details

### A.1 Observation Encoding and Gaussian Association

**Coordinates and time.** For a Gaussian generated from camera $v$ at context timestep $n$, let $\mathbf{R}_n^v$ denote the camera-to-common-frame rotation and $\tilde{\mathbf{p}}_i$ its source pixel in homogeneous coordinates. The back-projection direction is

$$\mathbf{d}_i = \mathbf{R}_n^v (\mathbf{K}_n^v)^{-1} \tilde{\mathbf{p}}_i. \tag{A.1}$$

This direction is not normalized. Accordingly, $d_i$ in $\boldsymbol{\mu}_i^0 = \mathbf{o}_i + d_i \mathbf{d}_i$ is camera-space axial depth. We reserve $r_{ki}$ and $r_k^*$ for distances along LiDAR rays.

All camera and LiDAR acquisition times are expressed in seconds relative to the same window origin. The source time $t_i$ belongs to the camera observation that produced Gaussian $i$, while $\tau_u$ is the acquisition time of the LiDAR sweep containing voxel entry $u$. Normalized times used internally by the visual backbone are converted back to seconds for association and motion. The velocity $\mathbf{v}_i$ therefore has units of meters per second.

**Feature encoders.** The visual pathway uses STORM-L/16 with 16 motion tokens and an input image resolution of $288 \times 512$. A linear layer maps each visual token to $p^2 \times 24$ channels, where $p = 16$. These outputs are rearranged into pixel-aligned features $\mathbf{f}_i^{\text{img}} \in \mathbb{R}^{24}$. Thus, $\mathbf{f}_i^{\text{img}}$ denotes the projected feature, which is added directly to the aggregated LiDAR feature.

The LiDAR encoder contains two $3 \times 3$ convolutional layers with 32 channels, GroupNorm, and GELU, followed by a $1 \times 1$ projection to 24 channels. Its ten input channels comprise validity, bounded log encodings of range, intensity, and elongation, sine and cosine of azimuth and elevation, an intra-sweep firing-time offset, and relative sweep time. Unavailable elongation and firing-time offsets are set to zero for nuScenes. The feature encoding is separate from the explicit intensity values used for reconstruction, which use raw intensity divided by 255.

**Association neighborhood.** Each sweep is voxelized independently at $0.5\,\text{m}$ resolution. For each query, we enumerate the surrounding $3 \times 3 \times 3$ voxel neighborhood and retain occupied entries whose centroids lie within $r = 1\,\text{m}$. The spatial and temporal bandwidths are $\sigma_s = 0.5\,\text{m}$ and $\sigma_t = 0.6\,\text{s}$. The set $\mathcal{N}(\mathbf{x})$ in Eq. 2 therefore refers to this locally enumerated candidate set. Positions, features, and intensities are averaged within each voxel before association.

**Residual descriptor.** The fused feature $\mathbf{z}_i$ is combined with the initial Gaussian state and matching evidence:

$$\begin{aligned} \mathbf{u}_i = \big[ &\mathbf{z}_i,\ \boldsymbol{\mu}_i^0 / L_\mu,\ \log \mathbf{s}_i,\ \text{logit}(\text{sg}(o_i^C)), \\ &\mathbf{v}_i / V_0,\ s_i,\ C(d_i^{\text{near}} / L_\mu) \big], \end{aligned} \tag{A.2}$$

where $L_\mu = 75\,\text{m}$, $V_0 = 20\,\text{m/s}$, and $d_i^{\text{near}}$ is the nearest associated centroid distance. Scale values are lower-bounded before taking logarithms, and opacity values are clipped before taking logits. The descriptor contains 24 feature dimensions and 12 state dimensions, giving 36 inputs in total.

The residual head has two 64-dimensional hidden layers, with LayerNorm and GELU after the first layer and GELU after the second. It predicts the center correction, LiDAR opacity residual, and a 16-dimensional feature $\mathbf{f}_i^L$. The hyperbolic tangent in Eq. 4 acts independently on each coordinate, so

$$\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_i^0\|_\infty \le \rho_\mu s_i.$$

Scale, rotation, and motion retain the visual-head parameterization.

### A.2 Intensity Candidates, Fusion, and Rendering

**Candidate selection.** The spatial candidate selects the closest raw return within radius $r$ across context sweeps. The temporal candidate first selects the sweep closest to $t_i$ among those with a return within the same radius, then uses its closest return. Spatial distance breaks ties between sweeps with equal time differences.

Geometry and intensity use association at different locations. The geometric support $s_i$ is evaluated at the initial center $\boldsymbol{\mu}_i^0$. The weighted intensity candidate is evaluated at the corrected center $\boldsymbol{\mu}_i$, with

$$W_i^{\text{int}} = \sum_u w_{iu}(\boldsymbol{\mu}_i), \qquad s_i^w = 1 - \exp(-W_i^{\text{int}}). \tag{A.3}$$

Consequently, $s_i^w$ is not generally equal to the geometric support $s_i$. For a nearest-neighbor candidate, support is computed as $1-\exp(-w)$ using the spatial-temporal kernel evaluated at the selected raw return. An unmatched candidate has zero support.

**Candidate statistics.** The weighted path pools intensity and squared intensity separately within each voxel. Let $\overline{\ell^2}_u$ denote the mean squared raw/255 intensity in voxel entry $u$. The weighted variance is

$$v_i^w = \max\left(\frac{\sum_u w_{iu}(\boldsymbol{\mu}_i)\overline{\ell^2}_u}{\max(W_i^{\text{int}}, \epsilon)} - (b_i^w)^2,\ 0\right). \tag{A.4}$$

Using $\overline{\ell^2}_u$ preserves within-voxel variation; squaring the voxel mean would discard this information. The two nearest-neighbor candidates use individual measurements, so their variances are zero and their match counts are zero or one.

For the weighted candidate, $n_i^w$ counts matched occupied voxel–sweep entries, $d_i^w$ is the nearest matched centroid distance, and $\bar{\tau}_i^w$ is the weighted mean acquisition time. For a nearest-neighbor candidate, the distance and time refer to its selected raw return. Each candidate contributes six features:

$$\begin{aligned}\mathbf{e}_i^a = \big[b_i^a,\ C(v_i^a),\ s_i^a,\ C\left(\frac{\log(1+n_i^a)}{\log 33}\right),\\ C(d_i^a/r),\ C(|t_i - \bar{\tau}_i^a|/\sigma_t)\big].\end{aligned} \tag{A.5}$$

Here, $C$ clips each input to $[0,1]$. The three feature groups and $\mathbf{f}_i^L \in \mathbb{R}^{16}$ produce 34 inputs to an MLP with two 64-dimensional GELU hidden layers. The network predicts three fusion logits and one intensity residual.

**Gated fusion.** Fusion uses the active candidate set $\mathcal{V}_i$ defined in Section 3.3. The weighted candidate remains active even when its support is zero, while unsupported nearest-neighbor candidates are masked. A zero-initialized learnable scalar $\gamma$ defines $g = \tanh\gamma$. The final intensity and rendering support are

$$\begin{aligned}\hat{\ell}_i &= C\big(b_i^w + g(b_i^+ - b_i^w)\big),\\ s_i^{\text{int}} &= (1-|g|)s_i^w + |g|\sum_a \omega_i^a s_i^a.\end{aligned} \tag{A.6}$$

Prediction therefore starts from the weighted candidate. Because $g$ can be negative, the final intensity is a correction around $b_i^w$ and need not be a convex combination of the candidates. The residual bound $\rho_{\text{int}}$ controls the additional residual in Eq. 6, rather than the entire difference between $\hat{\ell}_i$ and $b_i^w$.

**Intensity rendering.** For intersecting Gaussians ordered by depth along LiDAR ray $k$, $r_{ki}$ and $a_{ki}$ denote their ranges and alpha contributions. We use

$$T_{ki} = \prod_{j<i}(1 - a_{kj}), \qquad \pi_{ki} = T_{ki} a_{ki}, \qquad A_k = \sum_i \pi_{ki}.$$

The supported compositing mass is $Z_k = \sum_i \pi_{ki} s_i^{\mathrm{int}}$, giving

$$\hat{y}_k = \begin{cases} \dfrac{\sum_i \pi_{ki} s_i^{\mathrm{int}} \hat{\ell}_i}{Z_k}, & Z_k > \epsilon_{\mathrm{int}}, \\ \bar{\ell}_{\mathrm{train}}, & \text{otherwise.} \end{cases} \tag{A.7}$$

The fallback $\bar{\ell}_{\mathrm{train}}$ is the mean intensity of training returns and is kept fixed. The rendering threshold $\epsilon_{\mathrm{int}}$ is distinct from the numerical denominator bound $\epsilon$ used in association.

### A.3 Sensor-Specific Opacity

Gaussians are grouped by their source image, corresponding to the same context timestep and camera view. For the group $\mathcal{B}_{b(i)}$ containing Gaussian $i$, we compute the camera-opacity-weighted residual mean. Let $\tilde{o}_j = \max(\mathrm{sg}(o_j^C), \epsilon_o)$ and $\breve{o}_i^C = \mathrm{clip}(\mathrm{sg}(o_i^C), \epsilon_o, 1 - \epsilon_o)$, with $\epsilon_o = 10^{-5}$. The LiDAR opacity is

$$\begin{aligned} \bar{\delta}_{b(i)} &= \frac{\sum_{j \in \mathcal{B}_{b(i)}} \tilde{o}_j \delta_j}{\sum_{j \in \mathcal{B}_{b(i)}} \tilde{o}_j}, \\ o_i^L &= \sigma\big(\mathrm{logit}(\breve{o}_i^C) + \delta_i - \bar{\delta}_{b(i)}\big). \end{aligned} \tag{A.8}$$

Here, sg denotes stop-gradient and $\sigma$ is the sigmoid function. Centering removes the group's common logit shift, while stop-gradient prevents this branch from directly updating camera opacity.

A distance-stratified anchor loss further constrains the difference between the two opacity fields. Let $d_i^L$ be the distance from the time-transformed Gaussian center to the target LiDAR origin. We use

$$\mathcal{L}_{\mathrm{anchor}} = \frac{\sum_i \beta(d_i^L) \left(o_i^L - \mathrm{sg}(o_i^C)\right)^2}{\max\left(\sum_i \beta(d_i^L), 1\right)}, \tag{A.9}$$

where

$$\beta(d) = \begin{cases} 1, & d < d_{\mathrm{near}}, \\ \beta_{\mathrm{mid}}, & d_{\mathrm{near}} \le d < d_{\mathrm{mid}}, \\ 0, & d \ge d_{\mathrm{mid}}. \end{cases}$$

The distance used to assign these weights is detached. The anchor therefore acts through LiDAR opacity rather than through changes to the Gaussian's distance group.

### A.4 Range Readout and Training Objectives

**Differentiable range readout.** We obtain a differentiable range estimate by concentrating compositing weights around a target accumulated-opacity level. Using the quantities defined in Appendix A.2, let

$$u_{ki} = 1 - T_{ki} + \frac{\pi_{ki}}{2}, \qquad c_k = \begin{cases} 0.5, & A_k > 0.5, \\ A_k/2, & A_k \le 0.5. \end{cases}$$

The smoothed weights and range estimate are

$$\zeta_{ki} = \pi_{ki} \exp\left(-\frac{|u_{ki} - c_k|}{\tau_r}\right), \qquad \hat{r}_k^{\mathrm{soft}} = \frac{\sum_i \zeta_{ki} r_{ki}}{\max\left(\sum_i \zeta_{ki}, 10^{-12}\right)}. \tag{A.10}$$

The temperature $\tau_r$ controls concentration around the target opacity level. Conditional on the rasterizer's contributor selection and ordering, the same expression is used for the forward readout and its gradients.

**Range and free-space supervision.** The fixed supervision set $\Omega_0$ is formed from valid measured returns with finite initial render outputs and sufficient initial accumulation. It remains unchanged during geometry optimization. For target range $r_k^*$, define $e_k = \min(|\hat{r}_k^{\text{soft}} - r_k^*|, M)$. The range and line-of-sight losses are

$$\begin{aligned} \mathcal{L}_{\text{range}} &= \frac{1}{R_{\text{norm}}|\Omega_0|} \sum_{k\in\Omega_0} \left[h_k e_k + (1-h_k)M\right], \\ \mathcal{L}_{\text{los}} &= \frac{1}{|\Omega_0|} \sum_{k\in\Omega_0} \sum_{i: r_{ki} < r_k^* - \varepsilon_{\text{los}}} a_{ki}. \end{aligned} \tag{A.11}$$

Here, $h_k = \mathbf{1}[A_k > \eta]$ indicates sufficient accumulation, $M$ is the missing-prediction penalty and error cap, $R_{\text{norm}}$ normalizes range, and $\varepsilon_{\text{los}}$ is the free-space margin.

During optimization, $h_k$ is replaced by the straight-through expression

$$p_k = \sigma\left(\frac{A_k - a_0}{\tau_a}\right), \qquad \tilde{h}_k = h_k + \kappa\left(p_k - \text{sg}(p_k)\right). \tag{A.12}$$

This retains the hard missing-prediction penalty in the forward pass while providing a smooth opacity gradient (Bengio et al., 2013). The hard threshold $\eta$ and backward surrogate center $a_0$ are separate parameters.

**Local surface structure.** We form pairs of horizontally or vertically adjacent rays whose two members belong to $\Omega_0$. Horizontal adjacency includes the azimuth seam; vertical adjacency does not wrap. For a pair $(k, l)$, define the signed range-difference error

$$e_{kl} = (\hat{r}_k^{\text{soft}} - \hat{r}_l^{\text{soft}}) - (r_k^* - r_l^*).$$

The smooth-surface set $\mathcal{E}_s$ contains pairs with $|r_k^* - r_l^*| < \theta_s$, while the edge set $\mathcal{E}_e$ contains pairs with $|r_k^* - r_l^*| > \theta_e$. The two losses are

$$\mathcal{L}_{\text{surface}} = \text{mean}_{(k,l)\in\mathcal{E}_s} H_{\delta_s}(e_{kl}), \qquad \mathcal{L}_{\text{edge}} = \text{mean}_{(k,l)\in\mathcal{E}_e} H_{\delta_e}(e_{kl}), \tag{A.13}$$

where $H_\delta$ is the Smooth-L1 function

$$H_\delta(e) = \begin{cases} e^2/(2\delta), & |e| < \delta, \\ |e| - \delta/2, & \text{otherwise.} \end{cases}$$

**Readout and camera supervision.** On supervised rays with $A_k > 0.5$, we align the differentiable range with a detached hard opacity-median range:

$$\mathcal{L}_{\text{readout}} = \text{mean}_{k\in\Omega_0: A_k > 0.5} H_{\delta_{\text{rd}}}\left(\hat{r}_k^{\text{soft}} - \text{sg}(r_k^{\text{med}})\right). \tag{A.14}$$

When no eligible pairs or rays are present, the corresponding structure or readout term is zero. Camera supervision combines pixel and perceptual terms:

$$\mathcal{L}_{\text{cam}} = \mathcal{L}_{\text{MSE}} + \lambda_p \mathcal{L}_{\text{LPIPS}} + \lambda_s(1 - \text{SSIM}). \tag{A.15}$$

These terms and the opacity anchor are combined using Eq. 7.

**Intensity supervision.** Stage II keeps geometry, motion, and both opacity networks fixed. The weighted reference $y_k^w$ is obtained by replacing $(\hat{\ell}_i, s_i^{\text{int}})$ with $(b_i^w, s_i^w)$ in Eq. A.7. Its normalization therefore uses $Z_k^w = \sum_i \pi_{ki} s_i^w$, and it uses the same training-mean fallback when support is insufficient.

For the valid measured target returns $\Omega$, including returns that use the fallback, the intensity objective is

$$\mathcal{L}_{\text{int}} = \frac{1}{|\Omega| V_{\text{train}}} \sum_{k\in\Omega} (\hat{y}_k - \ell_k^*)^2 + \frac{\lambda_b}{|\Omega|} \sum_{k\in\Omega} H_1(\hat{y}_k - y_k^w). \tag{A.16}$$

Here, $\ell_k^*$ is the target intensity and $\lambda_b$ weights regularization toward the weighted reference. The fixed normalizer $V_{\text{train}}$ is the training-set MSE of the constant predictor. Both $V_{\text{train}}$ and $\bar{\ell}_{\text{train}}$ are retained from training rather than estimated from target batches. The intensity population $\Omega$ is distinct from the initially supported geometry population $\Omega_0$.

# B Evaluation Details

## B.1 Input Configuration

Context frames use offsets $\{0, 2, 4, 6, 8, 10, 12, 14, 16, 19\}$ within each input window. Target observations are excluded from scene construction. DriveR$^3$ uses the current window's context, whereas scene-specific baselines aggregate context across the scene. All methods are evaluated on common targets.

## B.2 LiDAR Metrics and Aggregation

Evaluation retains ego-filtered observed returns with $0 < r_k^* \leq 80\,\mathrm{m}$. For DriveR$^3$, evaluated distance is the accumulated depth $D_k^{\mathrm{acc}} = \sum_i \pi_{ki} r_{ki}$ without opacity normalization; the soft readout is used during geometry training. For evaluated distance $D_k$, range MAE and RMSE are computed per scan, while Depth L2 is

$$\mathrm{L2}_{\mathrm{Depth}} = \mathrm{median}_k[(D_k - r_k^*)^2]. \tag{B.1}$$

Chamfer distance uses symmetric squared nearest-neighbor distances between point clouds back-projected along common target-return directions. Ablation scores are averaged over all evaluated images or scans.

## B.3 Reconstruction-Time Measurement

Timing starts after warm-up on exclusively allocated GPUs and uses CUDA synchronization. Model loading, disk I/O, metric computation, and result saving are excluded. DriveR$^3$ constructs one representation per window and reuses it for all target queries. Scene-specific times report total scene optimization divided by the window count. Speedup uses matched measurement sets and excludes the feed-forward model's one-time offline training cost.