PointForward: Feedforward Driving Reconstruction through Point-Aligned Representations

Cheng Chi[1,4*] **Xianqi Wang**[2,1*] **Hongcheng Luo**[1,4*†] **Mingfei Tu**[1,4]
Gangwei Xu[5] **Zehan Zhang**[1,4] **Bing Wang**[1,4] **Guang Chen**[1,4]
Hangjun Ye[1,4] **Sida Peng**[3] **Xin Yang**[5,6✉] **Haiyang Sun**[1,4]

[1]Xiaomi EV [2]School of AIA, Huazhong University of Science and Technology [3]Zhejiang University
[4]Beijing Key Laboratory of Intelligent Driving Foundation Models and Simulation
[5]Huazhong University of Science and Technology [6]Optics Valley Laboratory
https://wm-research.github.io/PointForward



Figure 1: We present **PointForward**, a feedforward driving reconstruction framework through point-aligned representations. Compared to directly rendering results from the SOTA pixel-aligned method [4], it reconstructs fine structures without layering artifacts.

Abstract

High-fidelity reconstruction of driving scenes is crucial for autonomous driving. While recent feedforward 3D Gaussian Splatting (3DGS) methods enable fast reconstruction, their per-pixel Gaussian prediction paradigm often suffers from multi-view inconsistency and layering artifacts. Moreover, existing methods often model dynamic instances via dense flow prediction, which lacks explicit cross-view correspondence and instance-level consistency. In this paper, we propose **PointForward**, a feedforward driving reconstruction framework through point-aligned representations. Unlike pixel-aligned methods, we initialize sparse 3D queries in world space and aggregate multi-view image information via spatial-temporal fusion onto these queries, enforcing explicit cross-view consistency in a single feed-forward pass. To handle scene dynamics, we introduce scene graphs that explicitly organize moving instances during reconstruction. By leveraging 3D bounding boxes, our method enables instance-level motion propagation and temporally consistent dynamic representations. Extensive experiments show that PointForward achieves state-of-the-art performance on large-scale driving benchmarks.

[*] Equal contribution, [†] Project leader, [✉] Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Accurate reconstruction of driving scenes is fundamental to autonomous driving. Early methods based on Neural Radiance Fields (NeRFs) [23, 25] achieved impressive photorealism, but required hours of per-scene optimization and incurred prohibitive rendering costs. The advent of 3D Gaussian Splatting (3DGS) [13] dramatically improved efficiency by replacing implicit fields with explicit point-based primitives and a fast differentiable rasterizer. However, early 3DGS-based methods [7, 37, 41] still relied on per-scene optimization, which remained computationally expensive and ill-suited for large-scale real-time deployment.

Recently, feedforward 3DGS methods [3, 4, 5, 22, 27, 36, 38] have gained traction. By training on large-scale data, these models can infer Gaussians in a single forward pass, reducing reconstruction time from minutes to sub-seconds. Despite this speedup, two structural limitations hinder further improvements in accuracy. First, current methods primarily rely on per-pixel Gaussian prediction: Gaussians are independently predicted from image pixels in each view, yet the same physical point may be projected to slightly different 3D locations across views due to prediction inconsistencies. This leads to redundant Gaussians and multi-view inconsistencies, often manifesting as layering or ghosting artifacts in fine-grained regions. Second, previous feedforward methods model scene dynamics via dense flow prediction [4, 38]. They typically rely on 2D dynamic masks to label scene dynamics and then predict per-Gaussian 3D motion flows. 2D dynamic masks cannot fully capture the extent of 3D instances, and dense flow prediction models motion independently at the pixel level, lacking coherent instance-level representations. These limitations motivate us to develop a point-aligned feedforward reconstruction framework.

In this paper, we propose **PointForward**, a feedforward driving reconstruction framework that goes beyond per-pixel Gaussian prediction by operating on sparse spatial-temporal queries. To build a global point-aligned representation, PointForward first initializes a set of sparse 3D queries directly in world space. These queries are then projected onto all input views to aggregate corresponding image features and geometric cues. Through spatial-temporal fusion, each query is updated into a globally consistent and view-coherent 3D representation. As illustrated in Figure 1, PointForward faithfully reconstructs fine structures without duplicate Gaussians, whereas prior pixel-aligned methods often produce noticeable redundancy and artifacts.

For dynamic modeling, PointForward likewise adopts a point-aligned formulation. Inspired by scene graphs commonly used in per-scene reconstruction [7, 37], we explicitly organize dynamic instances and static backgrounds in 3D space using 3D bounding boxes. Queries associated with dynamic instances are transformed into canonical spaces, in which instance motion can be modeled consistently over time. During rendering, these dynamic queries are mapped back to the target world space according to the desired timestamp. This design enables instance-level motion propagation and temporally coherent dynamic representations, avoiding the fragmented motions often produced by dense flow prediction.

We evaluate PointForward on the Waymo Open Dataset [26] and nuScenes [2]. Experimental results show that PointForward consistently outperforms prior feedforward methods. On Waymo, PointForward achieves a global PSNR of 28.48 and a dynamic PSNR of 25.01. When directly generalized to nuScenes, it obtains a PSNR of 26.54, with further gains after fine-tuning. These results demonstrate the effectiveness of our point-aligned formulation and scene-graph-based dynamic modeling. Qualitative and Quantitative results further show that PointForward recovers fine details under extrapolated viewpoints while avoiding ghosting artifacts.

Our contributions can be summarized as follows:

- We propose **PointForward**, a feedforward driving reconstruction framework through point-aligned representations. It aggregates multi-view image information and enforces cross-view consistency in a single forward pass.
- We introduce a point-aligned representation that replaces per-pixel Gaussian prediction with sparse 3D queries. We further incorporate scene graphs for dynamic modeling, enabling instance-level motion propagation and temporally consistent dynamic reconstruction.
- We achieve state-of-the-art performance on the Waymo and nuScenes benchmarks, with consistent improvements in both image quality and metric performance.

2 Related Work

Per-Scene Reconstruction. 3D Gaussian Splatting (3DGS) [13] explicitly represents scenes using anisotropic Gaussian primitives and enables real-time high-quality novel view synthesis through differentiable rasterization, laying the foundation for subsequent driving-scene reconstruction methods. However, the original 3DGS is designed for static scenes and cannot directly handle dynamic instances or unbounded street environments. To model scene dynamics, early works introduced implicit temporal representations without explicit supervision. PVG [6] embeds periodic temporal signals into Gaussian representations for unified dynamic-static modeling. DeformableGS [41] introduces learnable deformation fields by mapping scenes into a canonical space and predicting time-dependent deformations. S^3Gaussian [11] further transfers self-supervised dynamic decomposition to the 3DGS framework through 4D consistency constraints. Recent methods show that explicit 3D supervision can further improve the quality of dynamic reconstruction by decomposing scenes into foreground instances and static backgrounds. DrivingGaussian [46] introduces incremental static Gaussians and composite dynamic Gaussian graphs for surround-view driving scenes. Street Gaussians [37] models time-varying appearances of dynamic vehicles with 4D spherical harmonics and jointly optimizes tracked poses. AutoSplat [14] improves foreground reconstruction under sparse views using geometric symmetry priors and LiDAR initialization. HUGS [45] incorporates semantic labels into Gaussian representations for joint reconstruction and scene understanding. OmniRe [7] further extends modeling targets to vehicles, pedestrians, and cyclists, combining rigid transforms, SMPL-based deformations, and self-supervised deformation fields within a unified scene graph framework. Despite their strong reconstruction quality, these methods still rely on per-scene optimization, which requires substantial training time for each scene and lacks cross-scene generalization.

Feedforward Reconstruction. To overcome the efficiency bottleneck of per-scene optimization, feedforward reconstruction methods are trained on large-scale datasets to enable cross-scene generalization and directly predict 3D Gaussian representations in a single forward pass, reducing reconstruction time from hours to seconds. In general scene reconstruction, pixelSplat [3] and MVSplat [5] establish the basic paradigm of feedforward 3DGS through two-view encoding and multi-view cost volumes, respectively. DepthSplat [36] further improves geometric accuracy by jointly modeling depth estimation and Gaussian prediction. VGGT [31] proposes alternating spatial-temporal attention within a unified Transformer backbone to predict depth, point clouds, and camera parameters jointly. For driving scenes, DrivingForward [30] predicts Gaussians independently for each image and aggregates them into 3D space, serving as an early feedforward method for surround-view driving scenarios. STORM [38] extends feedforward reconstruction to large-scale dynamic outdoor scenes by jointly predicting pixel-aligned Gaussians and motion fields in a self-supervised manner. DGGT [4] further removes the reliance on calibrated cameras by jointly predicting Gaussian representations and camera parameters, while introducing lifespan modeling for dynamic visibility over arbitrary-length sequences. Most existing feedforward methods still adopt pixel-aligned Gaussian prediction, in which Gaussians are decoded independently from image pixels, often leading to redundant primitives and limited cross-view consistency. To alleviate this issue, recent works have begun to explore pixel-unaligned paradigms. SparseSplat [44] reduces Gaussian counts through entropy-based adaptive sampling, while C3G [1] introduces learnable query tokens with multi-view cross-attention to predict compact Gaussian sets. However, these methods are mainly validated on small indoor scenes and have not yet demonstrated strong performance in large-scale driving environments.

Multi-View 3D Object Detection. Multi-view 3D object detection provides an important source of inspiration for sparse query-based scene representation. Early methods are mainly based on dense bird's-eye-view (BEV) representations. BEVDet [10] adopts lift-splat projection for view transformation, while BEVFormer [17] introduces deformable attention for BEV feature generation and spatial-temporal fusion. BEVDepth [16] further improves detection accuracy with explicit depth supervision, and BEVStereo [15] enhances geometric reasoning by incorporating temporal stereo cues. Another line of work performs detection with sparse object queries. PETR [21] utilizes 3D positional encoding and global cross-attention for query-based detection, while DETR3D [33] directly samples multi-view image features using sparse 3D reference points. Building upon this paradigm, Sparse4D [19] introduces sparse spatial-temporal fusion, in which a compact set of 3D queries interacts across views and time to achieve efficient and accurate dynamic object detection. Inspired by these advances, our method adopts sparse 3D queries from object detection to feedforward 3D scene reconstruction.

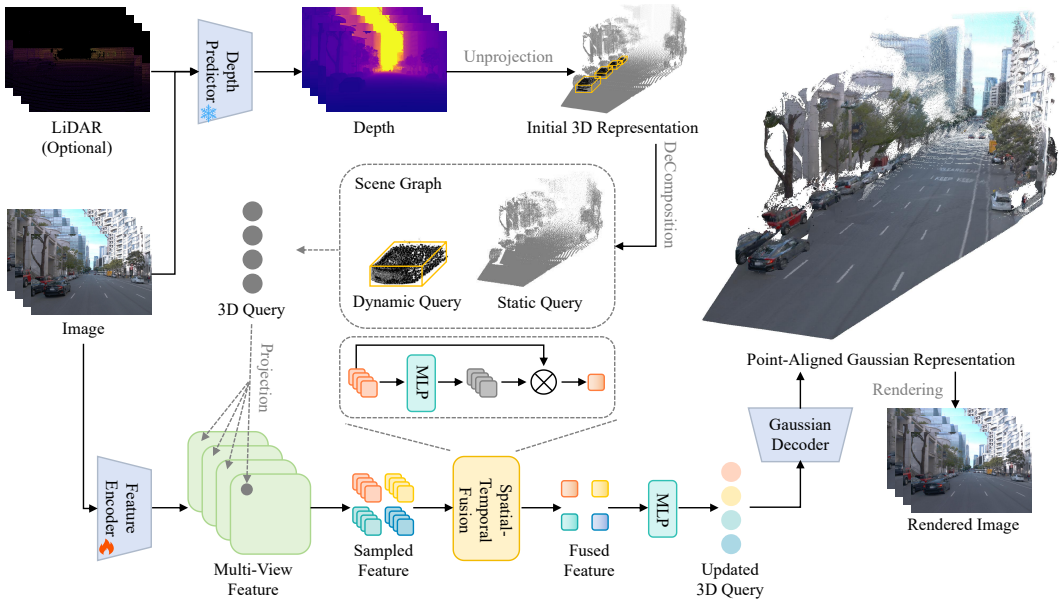

Figure 2: **Overview of PointForward.** We initialize sparse 3D queries in world space and project them onto multi-view image planes to aggregate features and geometric cues. A spatial-temporal fusion module produces coherent point-aligned representations, while scene-graph-based dynamic modeling ensures instance-level motion consistency for rendering dynamic driving scenes.

3 Method

We first introduce how PointForward generates a point-aligned representation (Section 3.1). To handle dynamic scenes, we introduce a scene graph that directly models dynamics in 3D space (Section 3.2). Dynamic queries are merged into canonical spaces, ensuring that each moving instance is globally unique across views. To achieve multi-view consistency, spatial-temporal fusion aggregates multi-view information onto each query, yielding a globally coherent 3D representation (Section 3.3). An overview of the model architecture is shown in Figure 2.

3.1 Point-Aligned Representation

Given a set of input images $\mathcal{I} = \{I_i \mid i = 1, \ldots, V\}$ with each $I_i \in \mathbb{R}^{H \times W \times 3}$. Each image has optional sparse LiDAR depth $D_i^L \in \mathbb{R}^{H \times W}$, camera extrinsics $[\mathbf{R}_i \mid \mathbf{t}_i]$, and camera intrinsics $\mathbf{K}_i$. We employ a pre-trained feature extractor (EfficientNetV2 [28]) and a pre-trained depth estimation model (InfiniDepth [42]) to obtain image features $\mathbf{F}_i \in \mathbb{R}^{H \times W \times C}$ and dense depth $\mathbf{D}_i \in \mathbb{R}^{H \times W}$, respectively. During training, the feature extractor is fine-tuned to adapt to the reconstruction task, while the depth estimation model remains frozen to preserve its monocular prior.

We then generate an initial dense query set $\mathcal{Q}_0 = \{\mathbf{q}_{i,j} \mid i = 1, \ldots, V,\ j = 1, \ldots, H \times W\}$. Each query $\mathbf{q}$ is a 12-dimensional vector comprising a 3D spatial coordinate, a 3-channel RGB color, and a 6-dimensional Plücker coordinate [24]. The 3D coordinate and Plücker coordinate of each query can be computed directly from the corresponding pixel depth and camera parameters. In essence, $\mathcal{Q}_0$ lifts image pixels into 3D space. At this stage, queries lack dynamic modeling and spatial-temporal fusion and therefore do not yet exhibit multi-view consistency.

Next, we perform dynamic modeling using scene graphs, with detailed procedures referred to Section 3.2. After this step, the initial query set $\mathcal{Q}_0$ is decomposed into a static query set $\mathcal{Q}_s = \{\mathbf{q}_{i,j} \mid i = 1, \ldots, V,\ j = 1, \ldots, N_s\}$ and a dynamic query set $\mathcal{Q}_d = \{\mathbf{q}_{i,j} \mid i = 1, \ldots, O,\ j = 1, \ldots, N_d\}$, where O denotes the number of dynamic instances in the scene. To reduce computation and avoid redundancy, we sample N_s static queries per view and N_d dynamic queries per instance. This yields our final point-aligned representation $\mathcal{Q} = \mathcal{Q}_s \cup \mathcal{Q}_d = \{\mathbf{q}_i\}_{i=1}^{N}$, where $N = V * N_s + O * N_d$ denotes the total number of sparse queries. It subsequently undergoes spatial-temporal fusion as detailed in Section 3.3.

3.2 Dynamic Modeling with Scene Graph

To model scene dynamics, we first obtain 3D bounding boxes for all moving instances. Let O be the number of dynamic instances. For each instance i, there is a set of bounding boxes $\mathcal{B}_i = \{\mathbf{b}_{i,j}\}_{j=1}^{B_i}$, where B_i is the number of available boxes. These boxes are temporally ordered, with $\mathbf{b}_{i,1}$ corresponding to the earliest timestamp. We designate $\mathbf{b}_{i,1}$ as the canonical reference box for instance i, whose local coordinate frame establishes the canonical space. Each box $\mathbf{b}_{i,j}$ is parameterized in world space by a center $\mathbf{c}_{i,j} \in \mathbb{R}^3$, dimensions $\mathbf{m}_{i,j} \in \mathbb{R}^3$, and a rotation matrix $\mathbf{R}_{i,j}^{\mathcal{B}}$. A 3D point $\mathbf{p} \in \mathbb{R}^3$ lies inside $\mathbf{b}_{i,j}$ if the following condition holds:

$$\left|(\mathbf{p} - \mathbf{c}_{i,j})(\mathbf{R}_{i,j}^{\mathcal{B}})^\top\right| \preceq \frac{\mathbf{m}_{i,j}}{2}, \tag{1}$$

where $\preceq$ denotes element-wise comparison.

Specifically, for a query $\mathbf{q} \in \mathcal{Q}_0$ with 3D spatial coordinate $\mathbf{p}$, if $\mathbf{p}$ lies inside any bounding box $\mathbf{b}_{i,j}$, $\mathbf{q}$ is assigned to instance i and marked as a dynamic query. Otherwise, it is treated as a static query belonging to the background. For each dynamic query associated with instance i, we further transform its world-space coordinate into the canonical space defined by the reference box $\mathbf{b}_{i,1}$:

$$\tilde{\mathbf{p}} = (\mathbf{p} - \mathbf{c}_{i,j})(\mathbf{R}_{i,j}^{\mathcal{B}})^\top \mathbf{R}_{i,1}^{\mathcal{B}} + \mathbf{c}_{i,1}, \tag{2}$$

where $\tilde{\mathbf{p}}$ denotes the canonicalized coordinate. After canonicalization, all dynamic queries belonging to the same instance are represented in a consistent, instance-centric coordinate frame, while static queries remain in the global world space. This scene graph formulation disentangles dynamic instance motion from static background geometry and provides structured motion priors for the subsequent spatial-temporal fusion.

Consequently, the initial dense query set $\mathcal{Q}_0$ is decomposed into a static query set $\mathcal{Q}_s$ and a dynamic query set $\mathcal{Q}_d$, as described in Section 3.1, where sparse sampling is applied to reduce redundancy and computational cost.

3.3 Spatial-Temporal Fusion

We first obtain a set of dynamic masks $\mathcal{M} = \{M_i \mid i = 1, \ldots, V\}$ by projecting all bounding boxes onto each image plane, where each $M_i \in \mathbb{R}^{H \times W}$ indicates dynamic regions in view i. For each $\mathbf{q} \in \mathcal{Q}$, we project its 3D spatial coordinate onto every input image I_i and sample the corresponding feature $\mathbf{f}_i \in \mathbb{R}^C$ from $\mathbf{F}_i$. Dynamic queries are first transformed from the canonical space back to world space before being projected. To avoid inconsistent motion features, if a static query falls inside a dynamic region in M_i, the sampled feature is set to zero. Besides $\mathbf{f}_i$, we also compute the absolute depth difference d_i between the query depth and the corresponding pixel depth from $\mathbf{D}_i$. This value reflects the geometric consistency of $\mathbf{f}_i$, where a smaller d_i indicates a more reliable observation. We then fuse the sampled feature $\mathbf{f}_i$ and depth difference d_i as:

$$\mathbf{f}_{s,i} = \phi\left([\phi(\mathbf{f}_i); \phi(d_i)]\right), \tag{3}$$

where $\phi(\cdot)$ denotes a multi-layer MLP operation applying to the feature dimension.

We further incorporate temporal cues for dynamic scenes. Let t_i denote the normalized timestamp of image I_i, and let t be the timestamp associated with query $\mathbf{q}$. The initial associated feature of $\mathbf{q}$ is denoted as $\mathbf{f}_q$, and compute the fusion weight for the i-th view as:

$$\begin{aligned} \mathbf{f}_{t,i} &= [\mathbf{f}_q \odot \mathbf{f}_{s,i}; t == t_i; t], \\ \omega_i &= \mathrm{Softmax}\big(\phi(\mathbf{f}_{t,i})\big), \end{aligned} \tag{4}$$

where $\odot$ denotes element-wise multiplication. We then obtain the fused feature $\mathbf{f}$ and lifespan parameter σ as:

$$\begin{aligned} \sigma &= \phi_v\left([\phi_c(\mathbf{f}_{t,1}), \ldots, \phi_c(\mathbf{f}_{t,V})]\right), \\ \mathbf{f} &= \sum_{i=1}^{V} \omega_i \mathbf{f}_{t,i}, \end{aligned} \tag{5}$$

where $\phi_c(\cdot)$ denotes a multi-layer MLP that reduces the feature dimension of $\mathbf{f}_{t,i}$ to a scalar, and $\phi_v(\cdot)$ denotes another multi-layer MLP that aggregates across views to produce the final lifespan parameter σ. Finally, we obtain the updated query representation by fusing all available information as:

$$\mathbf{q}' = \phi\left([\phi(\mathbf{q}); \phi(\mathbf{f}); \phi(t); \phi(\sigma)]\right). \tag{6}$$

3.4 Rendering

We decode each $\mathbf{q}'$ into a 3D Gaussian representation $\mathbf{g} \equiv (\mathbf{c}, \boldsymbol{\mu}, \mathbf{r}, \mathbf{s}, o)$, where the color feature $\mathbf{c} \in \mathbb{R}^{N_c}$, center $\boldsymbol{\mu} \in \mathbb{R}^3$, rotation quaternion $\mathbf{r} \in \mathbb{R}^4$, scale $\mathbf{s} \in \mathbb{R}^3$, and opacity $o \in \mathbb{R}$ are predicted from $\mathbf{q}'$. The opacity of each Gaussian at another timestamp t' is adjusted as:

$$o_{t'} = o \cdot e^{-\frac{1}{2}\left(\frac{t'-t}{\sigma+1}\right)^2}. \tag{7}$$

When rendering at timestamp t', Gaussians corresponding to dynamic queries are transformed from the canonical space back to the world space of the target timestamp. Following recent works such as UniSim [40] and SplatAD [9], we represent each Gaussian by a high-dimensional color feature $\mathbf{c}$ rather than directly predicting RGB values. As a result, Gaussian rendering produces a feature map rather than a color image. We then employ a lightweight multi-layer UNet to map the rendered feature map into three channels, yielding the final rendered RGB image.

3.5 Loss Function

We supervise the rendered images using a combination of pixel-wise ℓ_1 and perceptual losses. Specifically, the overall training loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{\ell_1} + \lambda \mathcal{L}_{\text{LPIPS}}, \tag{8}$$

where $\mathcal{L}_{\ell_1}$ denotes the ℓ_1 reconstruction loss and $\mathcal{L}_{\text{LPIPS}}$ denotes the perceptual LPIPS loss.

4 Experiment

4.1 Implementation Detail

We implement our PointForward using PyTorch and train it on 8 NVIDIA H20 GPUs. Training runs for 45K iterations with a global batch size of 8 and a learning rate of 5×10^{-4}. The training process uses the AdamW optimizer with a cosine learning rate scheduler, which includes a linear warm-up phase over the first 2K iterations. We use a image size of 576×768 and multi-resolution rendering supervision, randomly sampled from 350×518, 576×768, and 700×1036. We train PointForward on Waymo from scratch and fine-tune it on nuScenes from Waymo's checkpoint. Parameters are set as N_s = 100K, N_d = 10K, N_c = 128, λ = 0.2.

4.2 Dataset and Metric

We primarily conduct experiments on the Waymo Open Dataset [26], which contains 798 training and 202 validation sequences. Each sequence consists of a 20-second video captured at 10 FPS. During training, we randomly sample 20 frames (2 s) from the frontal three cameras for supervision, and the context frames are selected at 0 s, 0.5 s, 1.0 s, and 1.5 s within the sampled clip. During validation, we follow the same setup with DGGT [4]. Context frames are inputs, and the model predicts the remaining frames to validate rendering performance. We further evaluate zero-shot generalization and fine-tuning performance on the nuScenes dataset [2], which contains 750 training and 150 validation sequences. The clip sampling strategy follows the same protocol as Waymo. Longer video clip results are provided in the supplementary video.

We mainly report PSNR, SSIM, and Depth RMSE (D-RMSE) as the primary evaluation metrics. For extrapolated-view experiments, we also report FID to measure the distributional distance between rendered and ground-truth images.

4.3 Comparison with SOTA Method

Baseline. We compare our method with a range of existing approaches, including state-of-the-art feedforward pixel-aligned methods such as STORM [38] and DGGT [4]. For a fair comparison, we disable the diffusion-based rendering refinement module in DGGT for all extrapolated-view experiments and qualitative visualizations. This module is not specific to DGGT and can be applied to any method to refine rendered images. Removing it allows for a direct comparison of the raw Gaussian rendering quality across methods.

Table 1: **Quantitative results on Waymo.** We compare image quality and geometry metrics against both per-scene optimization and feedforward methods. PSNR, SSIM, and D-RMSE are reported.

Method	Dynamic Only			Full Image		
	PSNR↑	SSIM↑	D-RMSE↓	PSNR↑	SSIM↑	D-RMSE↓
EmerNeRF [39]	17.79	0.255	40.88	24.51	0.738	33.99
3DGS [13]	17.13	0.267	13.88	25.13	0.741	19.68
PVG [6]	15.51	0.128	15.91	22.38	0.661	13.01
DeformableGS [41]	17.10	0.266	12.14	25.29	0.761	14.79
LGM [29]	19.58	0.443	9.43	23.59	0.691	8.02
GS-LRM [43]	20.02	0.520	9.95	25.18	0.753	7.94
Depth Anything 3 [18]	15.96	0.429	12.20	23.27	0.686	11.66
AnySplat [12]	15.99	0.418	10.45	23.32	0.687	10.31
STORM [38]	22.10	0.624	7.50	26.38	0.794	5.48
DGGT [4]	22.80	0.651	6.55	27.41	0.846	3.47
Ours	**25.01**	**0.768**	**6.07**	**28.48**	**0.861**	**3.35**



Figure 3: **Qualitative results on Waymo.** PointForward produces sharper structures and cleaner object boundaries with significantly reduced ghosting artifacts.

Waymo. As shown in Table 1, PointForward achieves state-of-the-art performance on both the full image and dynamic regions. In particular, our method surpasses DGGT by 1.1 dB in PSNR on the full image and by 2.2 dB on dynamic regions, demonstrating the advantage of our point-aligned representation in enforcing multi-view consistency and improving dynamic reconstruction quality. As illustrated in Figure 3, our method produces sharper structures and cleaner object boundaries with significantly reduced ghosting artifacts. In contrast, AnySplat struggles to handle dynamic instances and large viewpoint changes, resulting in severe artifacts. STORM is limited by its training resolution and tends to generate over-smoothed results with missing fine details. DGGT often introduces ghosting in high-frequency regions and exhibits unstable rendering during rapid viewpoint changes due to its less coherent modeling.

We further evaluate reconstruction using only context frames. As shown in Table 2, PointForward outperforms DGGT by 3 dB in PSNR and nearly 0.2 in SSIM. Even when restricted to render context frames, PointForward still consistently outperforms prior methods, highlighting the effectiveness of our point-aligned Gaussian representation. This suggests that improved performance stems from a more coherent and expressive Gaussian representation.

Table 2: **Quantitative reconstruction results on Waymo.**

Method	PSNR↑	SSIM↑
AnySplat [12]	24.59	0.730
STORM [38]	26.55	0.851
DGGT [4]	30.54	0.884
Ours	**33.96**	**0.951**

Table 3: **Quantitative results on nuScenes.** We achieve SOTA performance on zero-shot and fine-tuning.

Method	Zero-Shot		Fine-Tuning	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑
STORM [38]	17.77	0.669	24.54	0.784
DGGT [4]	25.31	0.794	26.63	0.813
Ours	**26.54**	**0.821**	**27.50**	**0.826**



Figure 4: **Qualitative zero-shot results on nuScenes.**

nuScenes. We evaluate PointForward on nuScenes under both zero-shot and fine-tuning settings to assess its generalization and adaptation capabilities. As shown in Table 3, PointForward consistently outperforms prior methods in both settings, with more significant gains in the zero-shot scenario, achieving +1.2 dB in PSNR and +0.3 in SSIM. This demonstrates that our point-aligned representation is more stable and generalizable than pixel-aligned counterparts. As illustrated in Figure 4, prior methods often produce incorrect geometry and motion when applied to unseen scenarios. In contrast, PointForward accurately reconstructs scene structures and captures instance motions, even under cross-dataset conditions.

Extrapolated View. Since previous experiments were conducted on interpolated views, we further conduct extrapolated-view experiments to validate the effectiveness of point-aligned representation. For fair comparison, all methods take a single front-view camera as input. Note that DGGT predicts relative camera parameters and relative depth; therefore, we align its predicted depth with ground-truth depth to approximate absolute camera motion, which may introduce minor inaccuracies. As shown in Table 4, PointForward significantly outperforms STORM and DGGT in FID. Qualitative results in Figure 5 further demonstrate that our method preserves correct geometric structures in extrapolated viewpoints, such as lane layouts and vehicle shapes. These results highlight that point-aligned representation maintains stronger geometric consistency than pixel-aligned representation.

4.4 Ablation Study

Effectiveness of spatial-temporal fusion. As shown in Table 5, directly averaging the sampled multi-view information instead of applying spatial-temporal fusion leads to a noticeable performance drop. Removing depth difference information also degrades image quality. These results demonstrate the necessity of our spatial-temporal fusion module and the importance of incorporating geometric consistency cues during feature aggregation.

Table 4: **Quantitative extrapolated-view results on Waymo.** Lower FID is better.

Shift	STORM [38]	DGGT [4]	Ours
1 m	47.31	49.51	**16.95**
2 m	63.09	61.35	**25.80**
3 m	81.54	77.51	**33.71**
4 m	102.19	96.96	**42.63**
6 m	140.17	139.79	**65.99**

Table 5: **Ablation Study on Waymo.** w/o LiDAR depth still achieves SOTA performance.

Method	PSNR↑	SSIM↑
w/o weighted fusion	28.01	0.851
w/o depth-aware fusion	28.28	0.855
w/o lifespan	27.56	0.840
w/o LiDAR depth	28.00	0.847
Ours	**28.48**	**0.861**



Figure 5: **Qualitative extrapolated-view results on Waymo.** The camera is left shifted 3 meters.

Influence of lifespan. As shown in Table 5, removing the lifespan parameter results in only a 0.92 dB PSNR drop. In contrast, DGGT reports a much larger degradation of 3.11 dB. This suggests that our point-aligned representation is inherently more global and coherent, while the pixel-aligned representation relies more heavily on additional temporal parameters to model scene dynamics.

Independence of LiDAR depth. As shown in Table 5, PointForward still achieves 28.00 PSNR and 0.847 SSIM without using LiDAR depth, by directly adopting depths predicted from an external monocular depth estimation model [34, 35] such as MoGe-2 [32]. The performance still surpasses STORM and DGGT, achieving state-of-the-art results. This demonstrates that our method does not rely on LiDAR and can generalize effectively with only image-based geometric priors.

Query sampling and budget sensitivity. We uniformly sample static queries within each view and dynamic queries within each instance, without introducing texture-, or category-specific sampling heuristics. To study the sensitivity to the query budget, we independently train PointForward with different numbers of sampled static and dynamic queries while keeping all other settings unchanged. As shown in Table 6, halving both N_s and N_d reduces PSNR by only 0.03 dB and SSIM by 0.001. Even when trained with only one quarter of the default query budget, PointForward still achieves 28.25 PSNR and 0.856 SSIM. These results show that PointForward is robust to the choice of query budget and can maintain competitive reconstruction quality with substantially fewer queries.

Effect of image encoder and initialization. We further study the influence of the image encoder architecture and its initialization. As shown in Table 7, replacing the pretrained EfficientNetV2-S [28] with a randomly initialized counterpart results in only a small decrease from 28.48 to 28.40 PSNR. Using a pretrained ResNet-50 [8] also achieves comparable performance of 28.42 PSNR and 0.859 SSIM. These results indicate that PointForward is robust to the choice and initialization of the image encoder, while pretrained EfficientNetV2-S provides the best overall performance and is therefore adopted as our default setting.

Table 6: **Query budget sensitivity on Waymo.** PointForward is robust to different numbers of sampled queries.

N_s/N_d	PSNR↑	SSIM↑
25K / 2.5K	28.25	0.856
50K / 5K	28.45	0.860
100K / 10K	**28.48**	**0.861**

Table 7: **Effect of image encoder and initialization on Waymo.** The pretrained EfficientNetV2-S encoder gives the best performance.

Image encoder	Initialization	PSNR↑	SSIM↑
EfficientNetV2-S	Pretrained	**28.48**	**0.861**
EfficientNetV2-S	Random	28.40	0.855
ResNet-50	Pretrained	28.42	0.859

Table 8: **Robustness to predicted 3D box trajectories.** PointForward remains robust when replacing ground-truth box trajectories with detector-and-tracker predictions.

Dataset / Method	Box Trajectory	PSNR↑	SSIM↑	Dynamic PSNR↑
Waymo – STORM [38]	None	26.38	0.794	22.10
Waymo – DGGT [4]	None	27.41	0.846	22.80
Waymo – PointForward	GT	**28.48**	**0.861**	**25.01**
Waymo – PointForward	Predicted	28.18	0.856	24.50
nuScenes – PointForward	GT	**27.50**	**0.826**	**23.88**
nuScenes – PointForward	Predicted	27.25	0.823	23.43

Robustness to predicted 3D box trajectories. Our main experiments use ground-truth 3D box trajectories to isolate the reconstruction capability of different methods. We further evaluate PointForward with predicted trajectories to assess its robustness under practical settings. On Waymo, we retain GT trajectories during training and replace all test-time 3D boxes with detector-and-tracker predictions [37]. As shown in Table 8, this results in only a 0.30 dB PSNR, 0.005 SSIM, and 0.51 dB dynamic PSNR decrease. PointForward with predicted trajectories still substantially outperforms STORM and DGGT, particularly on dynamic regions.

On nuScenes, we use predicted box trajectories [20] during both training and inference, completely removing human-annotated dynamic trajectories from the pipeline. Despite detection, tracking, and temporal interpolation errors, performance decreases by only 0.25 dB in PSNR, 0.003 in SSIM, and 0.45 dB in dynamic PSNR. These results show that our scene-graph-based dynamic modeling is robust to trajectory prediction errors and does not require human-annotated trajectories at inference.

5 Conclusion

In this paper, we present PointForward, a feedforward driving reconstruction framework through point-aligned representations. PointForward introduces a point-aligned representation based on sparse 3D queries in world space, enabling explicit multi-view consistency and coherent global geometry. We further incorporate scene-graph-based dynamic modeling and spatial-temporal fusion to achieve consistent motion representation and robust feature aggregation across views and timestamps. Extensive experiments on Waymo and nuScenes demonstrate that PointForward achieves state-of-the-art performance and shows strong robustness under extrapolated viewpoints. We hope our work can inspire future research toward more scalable and consistent feedforward reconstruction frameworks.

Limitation. Current dynamic modeling mainly focuses on rigid instances represented by 3D bounding boxes. As a result, PointForward is less effective for highly non-rigid motions, such as pedestrians. Extending point-aligned representations toward more flexible non-rigid dynamic modeling remains an important direction for future work.

6 Acknowledgment

We use publicly available datasets, models, and code resources, including Waymo, nuScenes, and other public research resources. We confirm that all such resources are used solely for academic research purposes and not for any commercial activity, in accordance with their respective licenses and terms of use. This research is supported by National Natural Science Foundation of China (U25B2045, 62472184), the Fundamental Research Funds for the Central Universities, and the Innovation Project of Optics Valley Laboratory (Grant No. OVL2025YZ005)

References

[1] Honggyu An, Jaewoo Jung, Mungyeom Kim, Sunghwan Hong, Chaehyun Kim, Kazumi Fukuda, Minkyeong Jeon, Jisang Han, Takuya Narihira, Hyuna Ko, et al. C3g: Learning compact 3d representations with 2k gaussians. *arXiv preprint arXiv:2512.04021*, 2025.

[2] 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*, pages 11621–11631, 2020.

[3] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19457–19467, 2024.

[4] Xiaoxue Chen, Ziyi Xiong, Yuantao Chen, Gen Li, Nan Wang, Hongcheng Luo, Long Chen, Haiyang Sun, Bing Wang, Guang Chen, et al. Dggt: Feedforward 4d reconstruction of dynamic driving scenes using unposed images. *arXiv preprint arXiv:2512.03004*, 2025.

[5] 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*, pages 370–386. Springer, 2024.

[6] 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*, 134(3):83, 2026.

[7] Ziyu Chen, Jiawei Yang, Jiahui Huang, Riccardo De Lutio, Janick Martinez Esturo, Boris Ivanovic, Or Litany, Zan Gojcic, Sanja Fidler, Marco Pavone, et al. Omnire: Omni urban scene reconstruction. *arXiv preprint arXiv:2408.16760*, 2024.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] 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 Computer Vision and Pattern Recognition Conference*, pages 11982–11992, 2025.

[10] Junjie Huang, Guan Huang, Zheng Zhu, Yun Ye, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*, 2021.

[11] Nan Huang, Xiaobao Wei, Wenzhao Zheng, Pengju An, Ming Lu, Wei Zhan, Masayoshi Tomizuka, Kurt Keutzer, and Shanghang Zhang. S^3gaussian: Self-supervised street gaussians for autonomous driving. *arXiv preprint arXiv:2405.20323*, 2024.

[12] Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Transactions on Graphics (TOG)*, 44(6):1–16, 2025.

[13] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[14] Mustafa Khan, Hamidreza Fazlali, Dhruv Sharma, Tongtong Cao, Dongfeng Bai, Yuan Ren, and Bingbing Liu. Autosplat: Constrained gaussian splatting for autonomous driving scene reconstruction. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8315–8321. IEEE, 2025.

[15] Yinhao Li, Han Bao, Zheng Ge, Jinrong Yang, Jianjian Sun, and Zeming Li. Bevstereo: Enhancing depth estimation in multi-view 3d object detection with temporal stereo. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 1486–1494, 2023.

[16] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 1477–1485, 2023.

[17] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):2020–2036, 2024.

[18] 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*, 2025.

[19] Xuewu Lin, Tianwei Lin, Zixiang Pei, Lichao Huang, and Zhizhong Su. Sparse4d: Multi-view 3d object detection with sparse spatial-temporal fusion. *arXiv preprint arXiv:2211.10581*, 2022.

[20] Xuewu Lin, Zixiang Pei, Tianwei Lin, Lichao Huang, and Zhizhong Su. Sparse4d v3: Advancing end-to-end 3d detection and tracking. *arXiv preprint arXiv:2311.11722*, 2023.

[21] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *European conference on computer vision*, pages 531–548. Springer, 2022.

[22] 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 Computer Vision and Pattern Recognition Conference*, pages 11286–11296, 2025.

[23] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[24] Julius Plucker. Xvii. on a new geometry of space. *Philosophical Transactions of the Royal Society of London*, pages 725–791, 1865.

[25] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10318–10327, 2021.

[26] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

[27] Kaiyuan Tan, Yingying Shen, Mingfei Tu, Haohui Zhu, Bing Wang, Guang Chen, Hangjun Ye, and Haiyang Sun. Ufo: Unifying feed-forward and optimization-based methods for large driving scene modeling. *arXiv preprint arXiv:2602.20943*, 2026.

[28] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pages 10096–10106. PMLR, 2021.

[29] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2024.

[30] 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*, volume 39, pages 7374–7382, 2025.

[31] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025.

[32] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*, 2025.

[33] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on robot learning*, pages 180–191. PMLR, 2022.

[34] Gangwei Xu, Haotong Lin, Hongcheng Luo, Haiyang Sun, Bing Wang, Guang Chen, Sida Peng, Hangjun Ye, and Xin Yang. Pixel-perfect visual geometry estimation. *arXiv preprint arXiv:2601.05246*, 2026.

[35] Gangwei Xu, Haotong Lin, Hongcheng Luo, Xianqi Wang, Jingfeng Yao, Lianghui Zhu, Yuechuan Pu, Cheng Chi_, Haiyang Sun, Bing Wang, et al. Pixel-perfect depth with semantics-prompted diffusion transformers. *Advances in Neural Information Processing Systems*, 38:174731–174755, 2026.

[36] 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 Computer Vision and Pattern Recognition Conference*, pages 16453–16463, 2025.

[37] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians: Modeling dynamic urban scenes with gaussian splatting. In *European Conference on Computer Vision*, pages 156–173. Springer, 2024.

[38] Jiawei Yang, Jiahui Huang, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Apoorva Sharma, Maximilian Igl, Peter Karkus, Danfei Xu, et al. Storm: Spatio-temporal reconstruction model for large-scale outdoor scenes. *arXiv preprint arXiv:2501.00602*, 2024.

[39] Jiawei Yang, Boris Ivanovic, Or Litany, Xinshuo Weng, Seung Wook Kim, Boyi Li, Tong Che, Danfei Xu, Sanja Fidler, Marco Pavone, et al. Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. *arXiv preprint arXiv:2311.02077*, 2023.

[40] 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*, pages 1389–1399, 2023.

[41] 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*, pages 20331–20341, 2024.

[42] Hao Yu, Haotong Lin, Jiawei Wang, Jiaxin Li, Yida Wang, Xueyang Zhang, Yue Wang, Xiaowei Zhou, Ruizhen Hu, and Sida Peng. Infinidepth: Arbitrary-resolution and fine-grained depth estimation with neural implicit fields. *arXiv preprint arXiv:2601.03252*, 2026.

[43] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 1–19. Springer, 2024.

[44] Zicheng Zhang, Xiangting Meng, Ke Wu, and Wenchao Ding. Sparsesplat: Towards applicable feed-forward 3d gaussian splatting with pixel-unaligned prediction. *arXiv preprint arXiv:2604.03069*, 2026.

[45] Hongyu Zhou, Jiahao Shao, Lu Xu, Dongfeng Bai, Weichao Qiu, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. Hugs: Holistic urban 3d scene understanding via gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21336–21345, 2024.

[46] Xiaoyu Zhou, Zhiwei Lin, Xiaojun Shan, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Drivinggaussian: Composite gaussian splatting for surrounding dynamic autonomous driving scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21634–21643, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the core contributions of the paper, which are subsequently supported by experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in the Conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not contain theory assumptions and proofs. The focus is on the architecture, implementation, and evaluation of the proposed model.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides sufficient implementation details to reproduce the main experimental results. We also plan to release the code and model to help reproduce.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Datasets are publicly available. The code is not provided during submission, but we plan to release it in the future.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides comprehensive details on the experimental details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We follow the settings of previous work.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the number of GPUs used during training.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that no ethical issues arise.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: There are currently no direct or significant societal impacts associated with our work. Therefore, no discussion on positive or negative societal impacts is included.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not involve the release of models or datasets that pose a high risk of misuse. Therefore, no specific safeguards are necessary.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Datasets and pre-trained models we used are properly cited in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Although the assets are not released during submission, we plan to release them in the future.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not contain crowdsourcing experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve research with human subjects. Therefore, no IRB or equivalent ethical approval is required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Our work does not involve any important, original, or non-standard usage of large language models (LLMs). Any use of LLMs was limited to writing assistance and does not impact the scientific contributions of the paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.