

Targetless LiDAR-Camera Calibration with Neural Gaussian Splatting

Haebeom Jung¹, Namtae Kim¹, Jungwoo Kim², and Jaesik Park^{1*}

Abstract—Accurate LiDAR-camera calibration is crucial for multi-sensor systems. However, traditional methods often rely on physical targets, which are impractical for real-world deployment. Moreover, even carefully calibrated extrinsics can degrade over time due to sensor drift or external disturbances, necessitating periodic recalibration. To address these challenges, we present a Targetless LiDAR-Camera Calibration (TLC-Calib) that jointly optimizes sensor poses with a neural Gaussian-based scene representation. Reliable LiDAR points are frozen as anchor Gaussians to preserve global structure, while auxiliary Gaussians prevent local overfitting under noisy initialization. Our fully differentiable pipeline with photometric and geometric regularization achieves robust and generalizable calibration, consistently outperforming existing targetless methods on the KITTI-360, WAYMO, and FAST-LIVO2 datasets. In addition, it yields more consistent Novel View Synthesis results, reflecting improved extrinsic alignment. The project page is available at: <https://www.haebeom.com/tlc-calib-site/>.

Index Terms—Sensor fusion, calibration and identification, computer vision for transportation.

I. INTRODUCTION

RECENT advances in Novel View Synthesis (NVS) have enabled increasingly sophisticated reconstruction of 3D scenes from 2D images [1], [2], [3], [4]. Despite these innovations, achieving higher rendering quality and precise 3D geometry often requires accurate geometry using multi-sensor fusion, such as the integration of LiDAR and multiple cameras. This complementary fusion provides richer and more accurate spatial information, and recent studies [5], [6] have reported substantial performance gains, especially in NVS tasks.

However, neural rendering techniques in multi-sensor setups rely heavily on accurate knowledge of each sensor’s mounting position and orientation, known as sensor extrinsics. These parameters are not necessarily static. Over time, even slight mechanical vibrations, thermal expansion, or physical impacts can induce subtle shifts in sensor positioning, resulting in misalignment and the need for periodic recalibration.

Target-based calibration methods [7], [8], [9] are widely adopted as a standard solution. For instance, placing checkerboard patterns or spherical reflectors within the shared field

of view enables accurate pose estimation. While effective, this approach can require costly infrastructure or large-scale target installations, especially in systems with multiple sensors or wide baselines. Moreover, even carefully calibrated target-based methods often struggle to align LiDAR and camera data at far distances, limiting their utility in real-world scenarios.

By contrast, targetless methods [10], [11], [12] calibrate sensors using only raw sensor data, leveraging environmental features such as planes or edges [12]. These methods eliminate the need for physical targets. Nevertheless, they face significant challenges due to the intrinsic differences between LiDAR and camera modalities, particularly the sparsity of LiDAR point clouds. Deep learning-based approaches [13], [14] attempt to bridge this gap. However, such approaches typically require large labeled datasets and often struggle to generalize to new sensor configurations or scenes. Although NeRF-based methods [15], [16], [17] can jointly optimize scene representations and sensor poses, their implicit volumetric nature results in high computational overhead, often scaling with the number of images.

In contrast, we employ a neural Gaussian representation to enable efficient and scalable optimization. We propose *TLC-Calib*, a targetless LiDAR-camera calibration framework built upon this representation. By leveraging differentiable rendering, our method jointly optimizes sensor extrinsics and the scene representation without relying on explicit calibration targets or external supervision. Joint optimization is essential, as pose estimation is tightly coupled with the underlying scene representation. Recent studies [18] show that even small pose inaccuracies can severely degrade NVS quality, highlighting the importance of precise camera calibration. To support scalable optimization across diverse scenes, we introduce adaptive voxel control, which automatically adjusts anchor density based on scene scale and motion, eliminating the need for manual voxel resolution tuning. Reliable LiDAR points are designated as anchor Gaussians to preserve global structure, while auxiliary Gaussians provide local flexibility and mitigate overfitting under inaccurate initial poses. This design enables robust optimization even with noisy initialization and improves alignment quality across diverse environments.

In summary, the primary contributions of this paper are as follows: (i) We ensure metric scene scale by designating reliable LiDAR points as anchor Gaussians to preserve overall scene structure, while auxiliary Gaussians regularize local geometry under challenging initialization. (ii) We integrate adaptive voxel control and Gaussian scale regularization to reduce redundant anchor Gaussians and suppress over-dominant anisotropic Gaussians that hinder optimization

¹H. Jung, N. Kim, and J. Park are with the Department of Interdisciplinary Program in Artificial Intelligence, Seoul National University, Seoul 08826, South Korea (e-mail: {haebeom.jung, knt0613, jaesik.park}@snu.ac.kr)

²J. Kim is with the Department of Artificial Intelligence, Yonsei University, Seoul 03722, South Korea (e-mail: jungwkim@yonsei.ac.kr)

This work was supported by IITP Grant (RS-2021-II211343: AI Graduate School Program at Seoul National University (5%) and RS-2023-00227993: Detailed 3D reconstruction for urban areas from unstructured images (60%)) and NRF Grant (No.2023R1A1C200781211 (35%)) funded by the Korea Government (MSIT).

stability. (iii) We validate our approach on three real-world setups, including two autonomous driving datasets and a handheld solid-state LiDAR setup, demonstrating strong generalization, high calibration accuracy, and rendering quality.

II. RELATED WORK

A. Targetless Sensor Calibration

Targetless calibration methods align sensors using environmental cues instead of physical markers. Edge-based approaches [10], [19] extract geometric edges from point clouds and images to estimate sensor extrinsics. In parallel, learning-based approaches have also been actively studied. RegNet [13] employs convolutional neural networks to predict extrinsic parameters between LiDAR scans and images, while LCCNet [14] improves upon this by introducing a cost volume for more robust estimation. Additionally, segmentation-based methods maximize overlap regions [20], align object edges [21], or leverage SAM-based masks for calibration [22]. However, the accuracy of such methods is often limited by the quality of segmentation.

B. Neural Rendering for Sensor Calibration

Neural rendering methods such as NeRF [3] and 3DGS [4] have been extended to jointly refine camera poses and scene geometry via photometric loss [15], [16], [17], [23]. However, these approaches are primarily designed for camera-only systems and are difficult to extend to multi-sensor settings due to scale ambiguity and modality gaps between LiDAR and camera data. Recently, neural rendering has also been explored for LiDAR-camera extrinsic calibration. Early NeRF-based methods [15], [16], [17] formulate calibration as a radiance field optimization problem, but their high computational cost limits practical applicability. To alleviate this limitation, recent works [23], [24] adopt 3DGS to accelerate optimization. 3DGS-Calib [23] fixes Gaussians on LiDAR points to guide calibration, but its reliance on hash-grid encodings makes the optimization sensitive to scene complexity and hyperparameter choices. RobustCalib [24] introduces a two-stage strategy that learns geometric constraints from LiDAR point clouds using 2DGS, followed by extrinsic calibration with reprojection and triangulation losses. However, its performance depends on the quality of the estimated surface normals, which may degrade under sparse LiDAR observations. In contrast, TLC-Calib introduces anchor and auxiliary Gaussians to construct a fully differentiable scene representation that extends beyond LiDAR-overlapped regions, enabling robust and generalizable calibration across diverse environments.

III. METHOD

A. Preliminary: 3DGS with Differentiable Pose Rasterization

We employ 3D Gaussian Splatting (3DGS) as a differentiable scene representation for jointly optimizing the scene and camera poses. The scene is represented as a set of anisotropic 3D Gaussians, each parameterized by a center $\mu_i \in \mathbb{R}^3$, covariance Σ_i , opacity α_i , and view-dependent

color coefficients c_i . We consider a multi-camera setup with C cameras, indexed by $c \in \{1, \dots, C\}$. Given the camera intrinsics \mathbf{K} and camera pose $\mathbf{T}^c \in \text{SE}(3)$, each Gaussian is projected onto the image plane as a 2D Gaussian \mathbf{G}_i^{2D} [25].

After sorting Gaussians in a front-to-back order along the viewing direction, the rendered color at pixel \mathbf{u} is obtained via alpha compositing:

$$\mathbf{C}(\mathbf{u}) = \sum_{i=1}^N c_i \alpha_i \mathbf{G}_i^{2D}(\mathbf{u}) \prod_{j=1}^{i-1} (1 - \alpha_j \mathbf{G}_j^{2D}(\mathbf{u})). \quad (1)$$

The rendered color in Eq. (1) is an explicit function of the camera pose through the projection of Gaussian means and view-dependent colors. This allows gradients of a photometric loss \mathcal{L} to be analytically propagated to the camera pose, following the pose-differentiable rasterization framework of Gaussian Splatting SLAM [26]. Using the chain rule, the gradient of the loss with respect to the camera pose is expressed as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{T}^c} = \sum_i \left(\frac{\partial \mathcal{L}}{\partial \mu_i^{2D}} \frac{\partial \mu_i^{2D}}{\partial \mu_i^c} \frac{\partial \mu_i^c}{\partial \mathbf{T}^c} + \frac{\partial \mathcal{L}}{\partial c_i} \frac{\partial c_i}{\partial \mathbf{T}^c} \right), \quad (2)$$

where μ_i^{2D} denotes the projected mean. All required Jacobians follow standard rigid-body transformation rules on $\text{SE}(3)$.

Following [26], the camera pose is updated directly on the Lie group as

$$\mathbf{T}^c \leftarrow \exp\left(-\lambda \frac{\partial \mathcal{L}}{\partial \mathbf{T}^c}\right) \mathbf{T}^c, \quad (3)$$

ensuring geometrically consistent and fully differentiable pose optimization.

B. Overview

We propose a neural Gaussian-based approach for targetless calibration in a LiDAR and multi-camera setup. We adopt the LiDAR as the reference sensor, treating its coordinate frame as the global reference and calibrating all cameras relative to it. This design choice is driven by the wide horizontal field of view and precise 3D geometry of LiDAR, which enables reliable odometry estimation through SLAM [27], ICP-based registration [28], or fusion with GPS and IMU for drift-free trajectories. Such multi-sensor fusion pipelines are widely adopted in autonomous driving datasets, including KITTI-360 [29] and WAYMO [30]. Building on this, we aggregate LiDAR point clouds over time using odometry, ensuring geometric consistency before calibration. In this paper, we assume that LiDAR poses are given and that sensor timestamps are well synchronized. An overview of the proposed pipeline is shown in Fig. 1.

C. Neural Scene Representation

1) *Anchor Gaussians*: Since LiDAR serves as the reference sensor, we aggregate point clouds across timestamps $t \in \{1, \dots, T\}$ to form $\mathcal{P} = \bigcup_{t=1}^T \mathcal{P}_t$, where \mathcal{P}_t denotes each LiDAR scan. To control point density, we voxelize \mathcal{P} with an adaptively determined voxel size ε^* (Sec. III-E) and select a

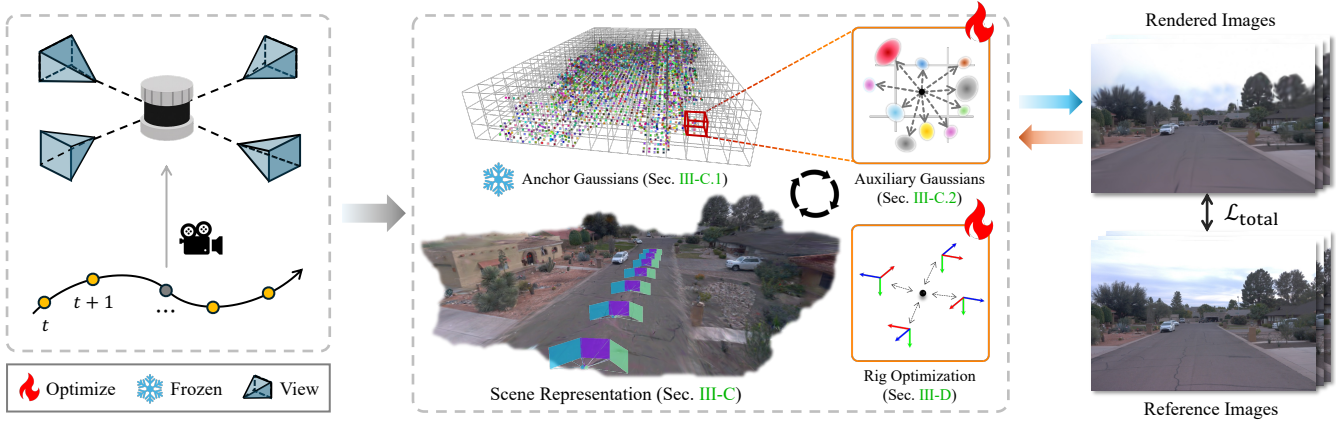


Fig. 1: Overview of the TLC-Calib pipeline. After aggregating LiDAR scans into a globally aligned point cloud, anchor Gaussians serve as fixed geometric references (their positions are not optimized), while auxiliary Gaussians adapt to local geometry and guide extrinsic optimization through photometric loss. Unlike anchor Gaussians, auxiliary Gaussians serve as learnable buffers around anchors, helping the optimization avoid local minima. Additionally, the camera rig optimization strategy jointly refines all cameras with respect to the scene, ensuring consistent and stable calibration across views.

subset of representative points. Voxelization is used solely for downsampling and spatial indexing. Anchor Gaussians are instantiated at the original coordinates of the selected LiDAR points, without shifting them to voxel centers or grid-aligned locations. Each selected LiDAR point $\mathbf{p}_j \in \mathcal{P}$ directly defines an anchor Gaussian with center $\mathbf{v}_i = \mathbf{p}_j$, providing a stable reference in real-world coordinates. Anchor positions remain fixed throughout training to preserve global scale and mitigate drift in LiDAR-camera calibration. In addition, anchors with persistently low opacity are treated as floaters and removed during training.

2) *Auxiliary Gaussians*: While anchor Gaussians remain static, we introduce auxiliary Gaussians to refine local geometry and improve pose convergence. Following [31], each anchor Gaussian \mathbf{v}_i is associated with a learned feature vector \mathbf{f}_i that encodes local geometric context. For each camera c , we use a view-dependent input $\mathbf{d}_{i,c}$, defined as the normalized viewing direction from the anchor to the camera center. The scalar l_i denotes a learned scale parameter of the anchor Gaussian. For each anchor \mathbf{v}_i , a lightweight MLP $F_{\text{auxiliary}}$ predicts a fixed set of positional offsets $\delta_i = \{\delta_{i,1}, \dots, \delta_{i,K}\}$, where K is the number of auxiliary Gaussians per anchor. The center of each auxiliary Gaussian, $\mathbf{m}_{i,k}$, is obtained by adding its offset to the corresponding anchor position:

$$\delta_i = F_{\text{auxiliary}}(\mathbf{f}_i, \mathbf{d}_{i,c}, l_i), \quad \mathbf{m}_{i,k} = \mathbf{v}_i + \delta_{i,k}. \quad (4)$$

Other Gaussian attributes such as covariance $\Sigma_{i,k}$, color $\mathbf{c}_{i,k}$, and opacity $\alpha_{i,k}$ are decoded via separate MLPs conditioned on $\{\mathbf{f}_i, \mathbf{d}_{i,c}, l_i\}$.

3) *Role of Auxiliary Gaussians*: Auxiliary Gaussians provide local, learnable support around anchors, allowing geometry and appearance to adjust during pose optimization. They further enable gradient propagation in sparse or LiDAR-unobserved regions, such as the sky or upper building areas, by introducing trainable structures beyond LiDAR coverage. This mechanism distinguishes our approach from prior methods [23], [24], which regresses Gaussian attributes directly from LiDAR points, thereby constraining the optimization to LiDAR-observed regions (see Fig. 2 for

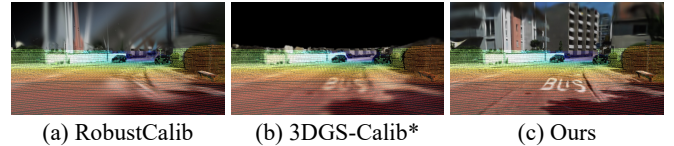


Fig. 2: Rendering results of (a)-(c), with the initial LiDAR points overlaid. Methods in (a) and (b) focus on LiDAR-observed regions with fixed geometry and consequently omit photometrically informative areas. In contrast, our method (c) explicitly represents camera-observed regions beyond LiDAR coverage.

comparison). Because the rendering loss cannot propagate to areas beyond LiDAR coverage, these methods discard photometrically informative regions. In contrast, our auxiliary Gaussians expand spatial coverage while preserving global scale consistency, as they are derived from anchor features and absorb supervision from nearby pixels even in LiDAR-unobserved areas. The effectiveness of this scheme is empirically validated in the ablation study in Sec. IV-F.3.

D. Joint Optimization of Scene and Exinsics

We denote the LiDAR-to-camera extrinsic for camera c as \mathbf{T}_c^e . Given the scene representation described previously, we jointly optimize the 3D Gaussians \mathbf{G} and the extrinsic parameters $\{\mathbf{T}_c^e\}_{c=1}^C$ corresponding to each of the C cameras.

Formally, the optimization objective is:

$$\min_{\mathbf{G}, \mathbf{T}_c^e} \sum_{c=1}^C \sum_{t=1}^T \mathcal{L}_{\text{total}}(I'_{c,t}, I_{c,t}; \mathbf{G}, \mathbf{T}_c^e), \quad (5)$$

where $I'_{c,t}$ is the rendered image (as in Eq. 1), and each camera c provides T observed images.

In practice, we adopt a *camera rig optimization* strategy based on per-image sequential updates. At each iteration, a single training image (c, t) is randomly sampled and rendered to compute its photometric loss. The gradient computed from this image is then immediately applied to the shared extrinsic \mathbf{T}_c^e of camera c :

$$\mathbf{T}_c^e \leftarrow \mathbf{T}_c^e - \alpha \nabla_{\mathbf{T}_c^e} \mathcal{L}_{\text{photo}}(I'_{c,t}, I_{c,t}), \quad (6)$$

where α is the step size and $\mathcal{L}_{\text{photo}}$ denotes the pose-differentiable photometric loss (Eq. 2). Because all frames

TABLE I: Baseline comparison on KITTI-360 [29]. Calibration performance is measured by success rate (SR, %), rotation error, and translation error across motion scenarios. SR denotes cameras within 1° and 20cm. Errors are averaged over 10 runs with color-coded rankings.

Scenes	CalibAnything [22]			INF [15]			RobustCalib [24]			3DGS-Calib* [23]			Ours		
	SR↑	R(°)↓	t(cm)↓	SR↑	R(°)↓	t(cm)↓	SR↑	R(°)↓	t(cm)↓	SR↑	R(°)↓	t(cm)↓	SR↑	R(°)↓	t(cm)↓
Straight	50.0	2.01±1.54	39.9±34.8	100.	0.22±0.12	10.9±1.44	100.	0.21±0.13	12.3±3.16	75.0	0.84±0.32	15.7±5.65	100.	0.11±0.04	12.7±1.43
Small zigzag	50.0	1.74±1.40	16.6±12.0	50.0	0.82±0.65	56.9±54.7	100.	0.44±0.12	9.53±3.35	50.0	0.84±0.80	18.1±8.86	100.	0.15±0.02	10.1±0.73
Large zigzag	0.00	4.61±1.59	78.5±21.4	100.	0.20±0.10	10.4±1.83	92.5	0.70±2.39	11.6±13.1	0.00	2.36±0.50	47.1±19.0	100.	0.09±0.03	6.17±2.27
Small rotation	0.00	9.29±2.62	92.3±15.4	75.0	0.23±0.11	17.4±3.54	7.50	3.45±6.15	106.±105.	75.0	1.84±2.27	18.7±10.9	100.	0.21±0.03	6.21±1.67
Large rotation	37.5	1.88±1.35	55.6±49.3	47.5	0.57±0.47	67.4±51.0	35.0	11.5±20.8	58.0±44.2	50.0	0.94±0.32	18.4±1.99	100.	0.09±0.03	9.18±1.04
Mean	27.5	3.91±3.39	56.6±40.3	74.5	0.41±0.45	32.6±41.5	67.0	3.26±10.6	39.4±63.7	50.0	1.36±1.28	23.6±16.0	100.	0.13±0.05	8.86±2.90

TABLE II: Training time comparison with baseline methods.

	CalibAnything	INF	RobustCalib	3DGS-Calib*	Ours
Time↓	> 5h	> 4h	> 1h	~ 0.15h	~ 0.18h

captured by camera c share a common extrinsic, the updated \mathbf{T}_c^e is immediately applied to all images from that camera. This per-view update allows each observation to directly correct pose misalignments, without requiring gradient accumulation across multiple views.

To analyze the behavior of the joint optimization, we examine how different scene representations shape the underlying energy landscape. We measure the photometric loss surface by sampling pose perturbations around the dataset calibration and evaluating the rendered-to-image discrepancy. The results show that the choice of representation strongly influences the optimization landscape. 3DGS [4] produces a rugged surface due to view-dependent overfitting, leading to unstable pose updates. In contrast, our representation preserves global structure via anchor Gaussians while reducing local ambiguity with auxiliary Gaussians, resulting in a smoother landscape and more reliable convergence (see Fig. 3).

E. Adaptive Voxel Control

To balance spatial resolution and computational efficiency, we select the voxel size ϵ^* such that the number of voxels after downsampling matches a target value V_{target} , using a binary search:

$$\epsilon^* = \arg \min_{\epsilon} |V(\epsilon) - V_{\text{target}}|. \quad (7)$$

The target voxel count V_{target} is defined proportional to the LiDAR trajectory length, $D_{\text{traj}} = \sum_{t=1}^{T-1} \|\mathbf{T}_{t+1}^L - \mathbf{T}_t^L\|_2$, as $V_{\text{target}} = \beta D_{\text{traj}}$, where β is a proportionality constant. This formulation enables *adaptive voxel control* (AVC), which dynamically regulates the number of anchor Gaussians according to the overall scene scale. Importantly, AVC also influences calibration accuracy. An excessively small ϵ^* leads to over-densified anchors, which can hinder optimization stability. Conversely, an excessively large ϵ^* reduces geometric coverage and degrades calibration precision. These effects are summarized in Tab. V.

F. Loss Function

We define the total loss as a combination of photometric supervision and a regularization term:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{photo}} \mathcal{L}_{\text{photo}} + \lambda_{\text{scale}} \mathcal{L}_{\text{scale}}. \quad (8)$$

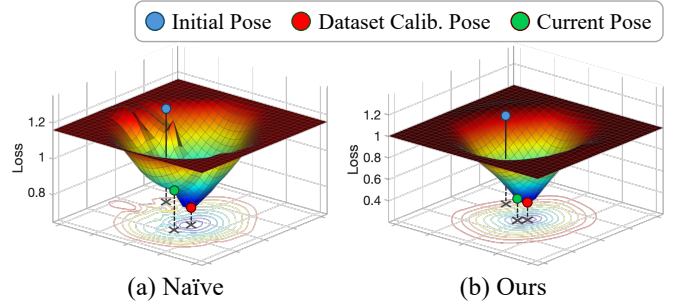


Fig. 3: An empirical example of optimization landscapes. We construct the loss surface by sampling pose perturbations around the dataset calibration and evaluating the photometric loss of rendered views. (a) The naïve baseline (3DGS [4] + rig optimization) overfits individual views, yielding an irregular landscape where the pose becomes trapped in local minima and fails to reach the dataset calibration. (b) Our method mitigates view-dependent overfitting using neural Gaussians, producing a smoother loss surface and more stable convergence toward the dataset calibration.

The photometric loss $\mathcal{L}_{\text{photo}}$ follows [4], combining LI and D -SSIM terms for image reconstruction. The regularization term $\mathcal{L}_{\text{scale}}$ penalizes degenerate, highly anisotropic Gaussians by constraining their aspect ratios. It is applied only to Gaussians that pass the view frustum filter. Let \mathcal{V} be the set of valid Gaussians, with each $\mathbf{s}_i \in \mathbb{R}^3$ representing scale along each axis. The loss is defined as:

$$\mathcal{L}_{\text{scale}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \max \left(\frac{\max(\mathbf{s}_i)}{\min(\mathbf{s}_i)} - \sigma, 0 \right), \quad (9)$$

where σ is a predefined threshold (see Sec. IV-C). If $|\mathcal{V}| = 0$, the loss is set to zero. This term stabilizes training by softly constraining Gaussian shapes while allowing adaptation to local geometry.

IV. EXPERIMENTAL EVALUATION

A. Dataset

We evaluate our method on two public autonomous driving datasets, KITTI-360 [29] and WAYMO [30], as well as one handheld dataset, FAST-LIVO2 [32]. KITTI-360 includes a spinning LiDAR, two front-facing perspective cameras, and two side-mounted fisheye cameras. Following prior work [23], we select three representative scenes based on motion patterns: Straight, Small zigzag, and Small rotation. To further examine more diverse motion patterns, we additionally include Large zigzag and Large rotation. WAYMO features a top-mounted spinning LiDAR and five perspective cameras covering the front and sides. FAST-LIVO2 consists of a solid-state LiDAR paired with a single perspective camera, with both sensors facing

TABLE III: NVS results on KITTI-360 [29] and WAYMO [30]. Metrics are averaged over 10 runs with color-coded rankings.

Datasets	Dataset Calib.			CalibAnything [22]			INF [15]			RobustCalib [24]			3DGS-Calib* [23]			Ours			
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	
KITTI-360	Straight	26.29	0.85	0.08	23.56	0.78	0.15	25.45	0.83	0.11	25.43	0.83	0.10	25.59	0.83	0.10	26.47	0.86	0.08
	Small zigzag	27.13	0.89	0.07	23.96	0.83	0.12	24.21	0.83	0.13	26.04	0.87	0.09	26.48	0.88	0.09	27.38	0.90	0.07
	Large zigzag	26.87	0.86	0.09	18.03	0.64	0.32	26.27	0.85	0.10	25.31	0.83	0.13	22.89	0.78	0.17	27.06	0.87	0.09
	Small rotation	24.96	0.81	0.13	13.58	0.50	0.52	23.50	0.77	0.17	18.52	0.65	0.32	24.26	0.79	0.15	25.00	0.81	0.13
	Large rotation	25.77	0.83	0.11	22.00	0.72	0.20	22.80	0.76	0.17	18.56	0.65	0.31	24.55	0.80	0.14	26.06	0.84	0.10
	Mean	26.20	0.85	0.10	20.22	0.69	0.26	24.45	0.81	0.14	22.77	0.77	0.19	24.52	0.80	0.14	26.39	0.85	0.09
WAYMO	Scene 81	27.81	0.88	0.13	25.66	0.84	0.18	28.08	0.88	0.12	27.11	0.86	0.14	28.64	0.89	0.11	29.06	0.89	0.11
	Scene 226	23.78	0.77	0.18	17.59	0.61	0.40	22.10	0.75	0.22	23.99	0.78	0.18	23.70	0.77	0.18	24.97	0.80	0.15
	Scene 362	25.99	0.87	0.11	17.87	0.72	0.33	20.89	0.78	0.24	26.79	0.88	0.10	26.65	0.88	0.09	27.08	0.89	0.09
	Mean	25.86	0.84	0.14	20.37	0.72	0.31	23.69	0.80	0.19	25.96	0.84	0.14	26.33	0.85	0.13	27.04	0.86	0.11

TABLE IV: NVS and calibration accuracy on FAST-LIVO2 [32]. Results using the dataset calibration are included for reference.

Scenes	Dataset Calib.			Ours			$R(\circ)\downarrow$	$t(\text{cm})\downarrow$
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow		
Building	22.13	0.755	0.172	22.16	0.754	0.171	0.56 ± 0.01	14.5 ± 0.37
Landmark	23.81	0.656	0.223	23.99	0.660	0.220	0.15 ± 0.01	8.71 ± 0.08
Sculpture	19.47	0.550	0.224	19.56	0.553	0.224	0.46 ± 0.02	10.1 ± 0.41
Mean	21.81	0.654	0.207	21.90	0.656	0.205	0.39 ± 0.18	11.1 ± 2.51

forward. For evaluation, we select three test scenes from each dataset that contain fewer dynamic objects: WAYMO (81, 226, 362) and FAST-LIVO2 (Building, Landmark, Sculpture). For each scene, we select approximately 80 sequential images per camera, using every second frame as a training view.

B. Experimental Setup

1) *Initialization*: To evaluate our method on three datasets, we follow the *from-LiDAR initialization* protocol [17], which provides coarse camera poses derived from LiDAR odometry and approximate camera rotations. For instance, in KITTI-360, the four cameras are roughly aligned with yaw angles of 0° (front), 90° (left), and -90° (right). This initialization is challenging, as translation errors can reach up to 1.2m, substantially degrading calibration accuracy. As a result, existing targetless calibration methods often fail to converge under this setting, as described in Sec. IV-D.

2) *Baselines*: We compare our method against four baselines, along with the dataset-provided calibration, resulting in a total of five comparison methods. Baselines are selected based on two criteria: (i) the availability of publicly released implementations to ensure consistent and reproducible evaluation, and (ii) relevance to targetless LiDAR-camera calibration. We adopt these criteria because several closely related approaches [16], [17], [23] have not released their source code, making direct and reproducible comparison difficult. As an exception, we include 3DGS-Calib [23], which is the most closely related method to our work. Since its official implementation is unavailable, we re-implement it following the paper and include it as our primary baseline, denoted with an asterisk (*). Implementation details are provided in Sec. IV-C.2. Finally, we include the *dataset calibration* officially released with the dataset as a reference for comparison.

C. Implementation Details

1) *TLC-Calib*: We use a two-layer MLP with ReLU activation and 32 hidden units to regress Gaussian attributes. The parameters are set to $K=5$, $\beta=5000$, and $\sigma=10$ for the number of auxiliary Gaussians, the proportionality factor of N_{target} , and the scale regularization threshold. Training is conducted for 30K iterations using AdamW, with a weight decay of 10^{-2} for the first 15K iterations. Each camera has a separate optimizer, with learning rates of 2×10^{-3} for rotation and 5×10^{-3} for translation, using a cosine annealing scheduler that decays the learning rate to $0.1\times$ the initial value. The loss combines D -SSIM and scale terms, weighted by $\lambda_{D\text{-SSIM}}=0.2$ and $\lambda_{\text{scale}}=1.0$. With this configuration, the method remains below 8GB of VRAM and uses the *same hyperparameters* across all datasets. All experiments are averaged over 10 random seeds and are conducted on a single RTX 4090 GPU.

2) *Baselines*: For 3DGS-Calib* [23], our primary baseline, we follow the original implementation with only minor adjustments. We use the standard hash-grid encoding with scene contraction and lightweight MLP heads with a hidden dimension 64. Based on direct communication with the original authors, we adopt pose learning rates of 1.5×10^{-2} for translation and 1.5×10^{-3} for rotation. Due to memory requirements, this baseline is evaluated on an RTX 5090 GPU. For baselines that assume a single camera setup [22], [24], we run the method independently for each camera and aggregate the results across cameras for evaluation.

D. Evaluation of Calibration Accuracy

Calibration accuracy is evaluated by comparing the estimated extrinsics with the *dataset calibration*. We report the success rate (SR, %), rotation error ($^\circ$), and translation error (cm), where a calibration is considered successful if the rotation and translation errors are within 1° and 20 cm, respectively [24]. Table I summarizes the results on the KITTI-360 [29] dataset. Our method achieves state-of-the-art performance across all evaluated scenes, attaining a 100% SR. In contrast, baseline methods succeed only in limited motion scenarios, with the strongest baseline, INF [15], reaching at most 74.5% SR.

KITTI-360 includes both front-view perspective and side-mounted fisheye cameras, requiring consistent calibration across wide viewing angles and severe distortions. Nevertheless, our method maintains high accuracy across all

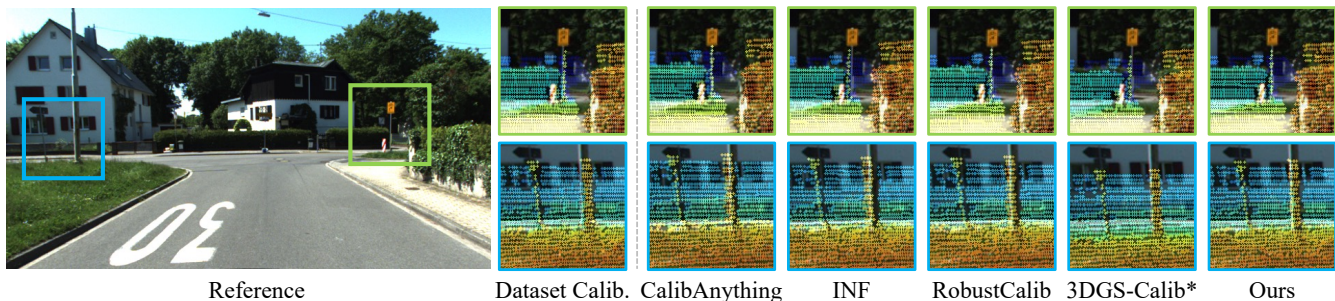


Fig. 4: Qualitative evaluation of LiDAR-camera alignment on KITTI-360 [29]. LiDAR points are projected onto images using the calibration results estimated by each baseline method. Point colors indicate 3D distances from the LiDAR, ranging from red (near) to blue (far).

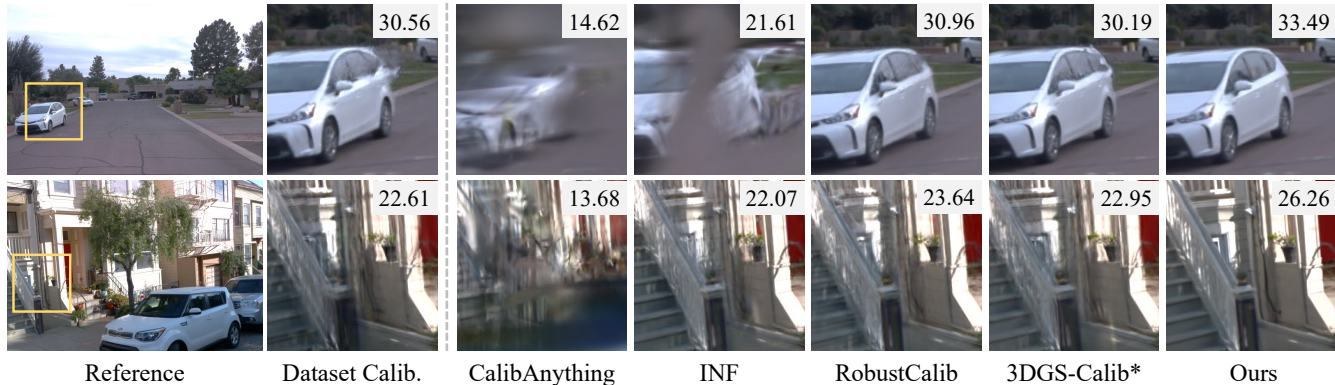


Fig. 5: Qualitative comparison of NVS results on WAYMO [30]. Key improvements are highlighted with yellow boxes, and cropped patches show zoomed-in regions for clarity. The PSNR of each rendered image is shown in the top-right corner.

camera types, improving rotation and translation accuracy by 68.3% and 62.5%, respectively. Moreover, the average rotation error of our method remains around 0.1° , effectively reducing large projection deviations caused by small angular misalignments. Figure 4 qualitatively visualizes the alignment quality, showing that our method preserves accurate alignment even for distant objects. This robustness stems from jointly optimizing camera poses within a shared scene representation that enforces structural consistency across views. In contrast, baseline methods [15], [22], [23], [24] often exhibit unstable convergence depending on hyperparameter choices and scene complexity. Table II further reports the training time comparison. Our method is the second fastest among all approaches, with an average training time of ~ 0.18 hours. Additional calibration results on the FAST-LIVO2 [32] dataset are provided in Tab. IV. Finally, WAYMO [30] provides less reliable calibration and pose information, as reported in recent work [33]. We therefore evaluate calibration quality on this dataset using Novel View Synthesis (NVS) performance as a complementary indicator to assess extrinsic alignment.

E. Evaluation of Novel View Synthesis

Recent studies [18] have shown that accurate camera poses are critical for achieving high-quality NVS. Accordingly, in addition to explicit pose accuracy metrics, we evaluate NVS performance to assess the impact of calibration accuracy on rendering consistency. For a fair comparison, we employ a unified 3DGS-based NVS pipeline [4], using LiDAR points for initialization. All methods are evaluated

within this identical pipeline, ensuring that NVS differences arise solely from the quality of the estimated LiDAR-to-camera extrinsics. As shown in Tab. III, our method achieves the highest rendering quality across all scenes on both KITTI-360 and WAYMO. On the WAYMO dataset, where the provided calibration is less reliable [33], baseline methods such as RobustCalib [24] and 3DGS-Calib [23] already outperform the dataset calibration in NVS quality. In contrast, on KITTI-360, which provides highly accurate ground-truth data, our method is the only approach that consistently surpasses the dataset calibration in NVS performance, indicating improved extrinsic estimation beyond the provided reference. Figure 5 qualitatively illustrates this advantage, showing that our calibration produces sharper reconstructions and more consistent rendering compared to both baseline methods and the dataset calibration. To further assess the sensor generalization capability of our approach, we additionally evaluate it on a solid-state LiDAR setup, which has not been explored by prior calibration methods. As reported in Tab. IV, consistent with previous results, our method slightly outperforms the dataset calibration.

F. Ablation Study

1) *Robustness to Noisy Initialization*: We analyze the robustness of our method to noisy initialization by progressively increasing the magnitude of pose perturbations. As shown in Fig. 6, we evaluate on four different difficulty levels: Easy, Medium, Hard, and Extreme, where the initial extrinsics are perturbed by up to $(5^\circ, 0.5\text{ m})$, $(10^\circ, 1.0\text{ m})$, $(15^\circ, 1.5\text{ m})$, and $(20^\circ, 2.0\text{ m})$, respectively. For each

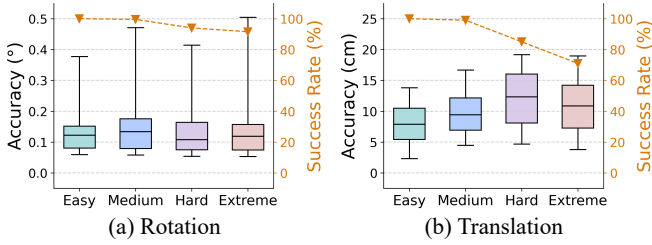


Fig. 6: Robustness to initialization perturbations. We evaluate four difficulty levels: *Easy*, *Medium*, *Hard*, and *Extreme*. For each level, we report the success rate (right axis) and the calibration error (left axis) over successful runs for (a) rotation and (b) translation.

TABLE V: Ablation study on model components and voxel density. Calibration accuracy and training time are reported for varying ϵ and K , with AVC selecting voxel sizes automatically.

R-O	Methods			Accuracy		Time(s) \downarrow
	$\mathcal{L}_{\text{scale}}$	AVC	K / ϵ	$R(^{\circ})\downarrow$	$t(\text{cm})\downarrow$	
	\checkmark	\checkmark	5 / -	1.94	64.7	647
\checkmark		\checkmark	5 / -	0.24	11.2	624
\checkmark	\checkmark		5 / 0.5	0.15	16.4	591
\checkmark	\checkmark		5 / 0.3	<u>0.13</u>	<u>9.13</u>	<u>612</u>
\checkmark	\checkmark		5 / 0.1	0.14	9.74	673
\checkmark	\checkmark	\checkmark	20 / -	0.12	9.99	729
\checkmark	\checkmark	\checkmark	10 / -	<u>0.13</u>	9.71	680
\checkmark	\checkmark	\checkmark	5 / -	<u>0.13</u>	8.86	625

level, we report the success rate and the calibration error over successful runs for both rotation and translation. As the perturbation magnitude increases, the success rate gradually decreases, with the most noticeable drop under the *Extreme* setting.

However, across all difficulty levels, the median rotation and translation errors of successful runs remain consistently low, indicating stable convergence once optimization succeeds. In practical scenarios, initial calibration errors typically fall within the *Easy* or *Medium* ranges. Under these conditions, our method achieves near-perfect success rates with consistently low calibration errors. These results show that our approach is well suited for real-world targetless calibration, where initial poses are often imprecise but rarely subject to large perturbations.

2) *Model Components and Voxel Density*: Table V analyzes the impact of model components and voxel density on calibration accuracy and training time. Removing rig optimization (R-O) results in large rotation and translation errors, confirming that joint optimization of camera extrinsics is essential. Scale regularization further improves optimization stability and consistently reduces calibration error. Increasing the number of auxiliary Gaussians K to $2\times$ or $4\times$ the default value ($K=5$) reduces the relative contribution of anchor Gaussians, leading to degraded translation accuracy. With manually fixed voxel sizes ϵ , calibration performance becomes sensitive to hyperparameter choices, and overly fine voxelization increases training time with limited accuracy gains. In contrast, adaptive voxel control (AVC presented in Sec. III-E) achieves comparable or better accuracy without requiring manual tuning of ϵ . Overall, our method exhibits limited sensitivity to voxel density, with only minor perfor-

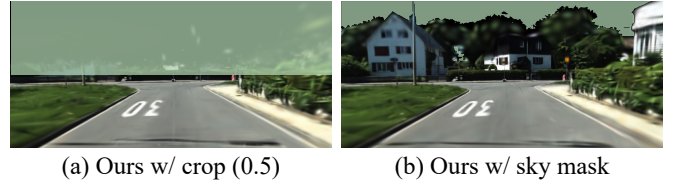


Fig. 7: Rendering results of our method with (a) image cropping (0.5) and (b) sky masking. The mask is overlaid for visualization.

TABLE VI: Effect of masking strategies on calibration and NVS metrics. Calibration accuracy and NVS performance under different masking settings.

Models	Accuracy		Novel View Synthesis		
	$R(^{\circ})\downarrow$	$t(\text{cm})\downarrow$	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Ours w/ crop (0.7)	1.64	72.9	22.16	0.751	0.196
Ours w/ crop (0.5)	0.59	40.9	25.87	0.841	0.106
Ours w/ crop (0.3)	0.14	22.1	<u>26.27</u>	0.851	0.097
Ours w/ sky mask	0.12	<u>17.5</u>	26.26	<u>0.852</u>	<u>0.096</u>
Ours full	<u>0.13</u>	8.86	26.39	0.854	0.094

TABLE VII: Ablation study of extrinsic parameter generalization on KITTI-360 [29]. Calibration optimized on *Scene A* is applied to *Scene B* and *Scene C*, and evaluated using NVS metrics.

Scenes	Dataset Calib.			Ours		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Seq. 9A	26.87	0.864	0.092	27.06	0.867	0.091
Seq. 9B	26.21	0.866	0.089	26.37	0.868	0.089
Seq. 9C	24.01	0.829	0.120	24.08	0.830	0.119
Seq. 10A	24.96	0.810	0.130	25.00	0.813	0.129
Seq. 10B	24.04	0.814	0.124	24.25	0.817	0.123
Seq. 10C	23.74	0.832	0.100	23.95	0.836	0.098

mance degradation observed under extreme settings.

3) *Effect of Masking LiDAR-Unobserved Areas*: Prior 3DGS-based targetless calibration methods [23], [24] restrict photometric supervision to LiDAR-observed regions in order to maintain geometric consistency, thereby discarding image regions that are not directly supported by LiDAR measurements (Fig. 2). To analyze how this design choice affects optimization behavior and calibration accuracy, we conduct an ablation study that explicitly masks LiDAR-unobserved image regions. Table VI summarizes calibration accuracy and NVS performance under different masking strategies. Both image cropping and sky masking remove photometric supervision from LiDAR-unobserved regions, reducing the amount of available image information and degrading pose optimization. In particular, calibration performance consistently deteriorates as the cropping ratio increases, and even a relatively mild cropping ratio (0.3) leads to a noticeable performance drop compared to the full model. Similarly, sky masking [34] also underperforms the full model, as it restricts a comparable amount of photometric information to mild image cropping. In contrast, the full model achieves the best calibration accuracy and NVS quality by retaining photometric supervision over the entire image without explicit masking. Figure 7 illustrates the masking strategies used in this ablation study, while Fig. 2(c) shows that our method successfully reconstructs upper image regions without masking.

4) *Extrinsic Generalization*: To evaluate extrinsic generalization, we apply the extrinsics optimized on `Scene A` to novel `Scene B` and `Scene C` without recalibration. A key indicator of calibration quality is whether the estimated extrinsics remain effective when transferred across sequences captured by the same vehicle. Here, `Seq. 9A` (Large zigzag) and `Seq. 10A` (Small rotation) serve as source scenes, while `9B/C` and `10B/C` are their corresponding targets within the same KITTI-360 sequence. For clarity, source scenes are indicated in `gray`. As shown in Tab. VII, our method achieves consistently higher NVS metrics on the target scenes than the default dataset calibration, demonstrating strong cross-scene calibration robustness.

V. CONCLUSION

In this paper, we introduced TLC-Calib, a targetless LiDAR-camera calibration that leverages a neural scene representation without scene-specific hyperparameters. Our approach jointly optimizes the scene representation and sensor poses using neural Gaussians, which mitigate viewpoint overfitting, avoid poor local minima, and improve overall optimization stability. Experiments on two public driving datasets and one handheld dataset demonstrate that TLC-Calib achieves superior pose accuracy, rendering quality, and generalization compared to existing methods. Similar to prior targetless calibration approaches, our method assumes synchronized sensors, and relies on reasonably accurate LiDAR odometry for initialization. Extending the framework to handle temporal misalignment, and joint multi-LiDAR calibration remains an important direction for future work.

REFERENCES

- [1] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96. Association for Computing Machinery, 1996, p. 31–42.
- [2] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96. Association for Computing Machinery, 1996, p. 43–54.
- [3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Eur. Conf. Comput. Vis.*, 2020, pp. 405–421.
- [4] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D gaussian splatting for real-time radiance field rendering," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.
- [5] C. Zhao, S. Sun, R. Wang, Y. Guo, J.-J. Wan, Z. Huang, X. Huang, Y. V. Chen, and L. Ren, "TCLC-GS: Tightly Coupled LiDAR-Camera Gaussian Splatting for Autonomous Driving," in *Eur. Conf. Comput. Vis.*, 2024, p. 91–106.
- [6] Z. Chen, J. Yang, J. Huang, R. de Lutio, J. M. Esturo, B. Ivanovic, O. Litany, Z. Gojic, S. Fidler, M. Pavone, L. Song, and Y. Wang, "OmniRe: Omni Urban Scene Reconstruction," in *Int. Conf. Learn. Represent.*, 2025.
- [7] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 3, 2004, pp. 2301–2306.
- [8] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *IEEE Int. Conf. Robot. Automat.*, 2012, pp. 3936–3943.
- [9] F. M. Mirzaei, D. G. Kottas, and S. I. Roumeliotis, "3D LiDAR-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization," *Int. J. Robot. Res.*, vol. 31, no. 4, pp. 452–467, 2012.
- [10] J. Levinson and S. Thrun, "Automatic online calibration of cameras and lasers," in *Robot. Sci. Syst.*, vol. 2, no. 7, 2013.
- [11] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2007, pp. 4164–4169.
- [12] M. Á. Muñoz-Bañón, F. A. Candelas, and F. Torres, "Targetless camera-lidar calibration in unstructured environments," *IEEE Access*, vol. 8, pp. 143 692–143 705, 2020.
- [13] N. Schneider, F. Piewak, C. Stiller, and U. Franke, "RegNet: Multimodal sensor registration using deep neural networks," in *IEEE Intell. Vehi. Symp.*, 2017, pp. 1803–1810.
- [14] X. Lv, B. Wang, Z. Dou, D. Ye, and S. Wang, "LCCNet: LiDAR and camera self-calibration using cost volume network," in *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, 2021, pp. 2888–2895.
- [15] S. Zhou, S. Xie, R. Ishikawa, K. Sakurada, M. Onishi, and T. Oishi, "INF: Implicit neural fusion for lidar and camera," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2023, pp. 10 918–10 925.
- [16] Q. Herau, N. Piasco, M. Bennehar, L. Roldão, D. Tsishkou, C. Migniot, P. Vasseur, and C. Démonceaux, "MOISST: Multimodal Optimization of Implicit Scene for SpatioTemporal Calibration," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2023, pp. 1810–1817.
- [17] Z. Yang, G. Chen, H. Zhang, K. Ta, I. A. Bãrsan, D. Murphy, S. Manivasagam, and R. Urtasun, "UniCal: Unified Neural Sensor Calibration," in *Eur. Conf. Comput. Vis.*, 2024, pp. 327–345.
- [18] E. Brachmann, J. Wynn, S. Chen, T. Cavallari, A. Monszpart, D. Turmukhambetov, and V. A. Prisacariu, "Scene coordinate reconstruction: Posing of image collections via incremental learning of a relocalizer," in *Eur. Conf. Comput. Vis.*, 2024, pp. 421–440.
- [19] X. Zhang, S. Zhu, S. Guo, J. Li, and H. Liu, "Line-based automatic extrinsic calibration of LiDAR and camera," in *IEEE Int. Conf. Robot. Automat.*, 2021, pp. 9347–9353.
- [20] Y. Zhu, C. Li, and Y. Zhang, "Online camera-lidar calibration with sensor semantic information," in *IEEE Int. Conf. Robot. Automat.*, 2020, pp. 4970–4976.
- [21] P. Rotter, M. Klemiato, and P. Skruch, "Automatic calibration of a lidar-camera system based on instance segmentation," *Remote Sensing*, vol. 14, no. 11, p. 2531, 2022.
- [22] Z. Luo, G. Yan, X. Cai, and B. Shi, "Zero-training LiDAR-Camera Extrinsic Calibration Method Using Segment Anything Model," in *IEEE Int. Conf. Robot. Automat.*, 2024, pp. 14 472–14 478.
- [23] Q. Herau, M. Bennehar, A. Moreau, N. Piasco, L. Roldão, D. Tsishkou, C. Migniot, P. Vasseur, and C. Démonceaux, "3DGS-Calib: 3D Gaussian Splatting for Multimodal SpatioTemporal Calibration," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2024, pp. 8315–8321.
- [24] S. Zhou, S. Xie, R. Ishikawa, and T. Oishi, "Robust LiDAR-Camera Calibration With 2D Gaussian Splatting," *IEEE Robot. Automat. Lett.*, 2025.
- [25] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross, "EWA splatting," *IEEE Trans. Vis. Comput. Graph.*, vol. 8, no. 3, pp. 223–238, 2002.
- [26] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024, pp. 18 039–18 048.
- [27] Z. Chen, Y. Xu, S. Yuan, and L. Xie, "ig-lid: An incremental gicp-based tightly-coupled lidar-inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 9, no. 2, pp. 1883–1890, 2024.
- [28] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "Kiss-icp: In defense of point-to-point icp-simple, accurate, and robust registration if done the right way," *IEEE Robot. Automat. Lett.*, vol. 8, no. 2, pp. 1029–1036, 2023.
- [29] Y. Liao, J. Xie, and A. Geiger, "Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3292–3310, 2022.
- [30] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2446–2454.
- [31] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai, "Scaffold-gs: Structured 3d gaussians for view-adaptive rendering," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024, pp. 20 654–20 664.
- [32] C. Zheng, W. Xu, Z. Zou, T. Hua, C. Yuan, D. He, B. Zhou, Z. Liu, J. Lin, F. Zhu, *et al.*, "Fast-LIVO2: Fast, direct lidar-inertial-visual odometry," *IEEE Trans. Robot.*, 2024.
- [33] Q. Herau, N. Piasco, M. Bennehar, L. Roldão, D. Tsishkou, B. Liu, C. Migniot, P. Vasseur, and C. Démonceaux, "Pose Optimization for Autonomous Driving Datasets using Neural Rendering Models," *arXiv preprint arXiv:2504.15776*, 2025.
- [34] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," *Adv. Neural Inform. Process. Syst.*, vol. 34, pp. 12 077–12 090, 2021.