HiGS-Calib: A Hierarchical 3D Gaussian Splatting based Targetless Local-Consistent LiDAR-Camera Calibration Method

Tianjun Zhang[1], Lin Zhang[2], *Senior Member, IEEE,* Hesheng Wang[1*], *Senior Member, IEEE*

***Abstract*—3D Gaussian Splatting (3DGS) has emerged as a powerful scene representation, offering geometrically dense and photometrically accurate modeling capabilities that present a promising new paradigm for accurate targetless sensor calibration. Current 3DGS-based LiDAR-camera calibration methods usually highly rely on the joint optimization of the Gaussian model and extrinsics, and thus suffer two critical limitations. On the one hand, the global optimization is usually sensitive to the accumulated localization error of LiDAR. On the other hand, the inaccurate extrinsics may cause oscillations during the joint optimization. Specifically, there is a fundamental dilemma: accurate extrinsic calibration requires accurate scene models, while constructing accurate models itself depends on accurate extrinsics. This dilemma frequently triggers oscillatory optimization trajectories, significantly increasing vulnerability to premature convergence at suboptimal states. To address these challenges, we propose HiGS-Calib, a novel 3DGS calibration pipeline featuring the integration of our proposed Local-Consistent Photometric-Geometric (LCPG) error model and the hierarchical architecture. The LCPG error leverages the spatial consistency within local windows to quantify pose misalignment using only geometric attributes of the 3DGS model, bypassing color-pose reliance constraints. Besides, diverging from joint optimization paradigms, HiGS-Calib implements coarse-to-fine iterative optimization, decoupling scene modeling from extrinsic refinement and thereby achieving stable and accurate calibration. Extensive evaluation demonstrates the significantly improved calibration accuracy and stability of our HiGS-Calib over other state-of-the-art methods. To make our results reproducible, the source code has been released at https://github.com/IRMVLab/HiGS-Calib.**

***Index Terms*—Gaussian splatting, LiDAR-camera calibration, targetless calibration.**

I. Introduction

In autonomous driving [1], [2], robotics [3], [4], and augmented reality [5], [6], robust perception systems typically rely heavily on multi-modal sensor fusion. The synergistic combination of camera sensors, which provide rich visual texture and dense geometry, and LiDARs, which deliver precise, long-range, and weather-robust 3D point measurements, forms a cornerstone of such fusion paradigms [7]–[9]. However, the effectiveness of integrating these heterogeneous data streams fundamentally hinges on accurately determining their spatial alignment, i.e., achieving accurate LiDAR-camera extrinsic calibration. Traditionally, such a calibration process heavily relies on specific targets, which need to be properly placed within the overlapping field of view of both sensors and accurately configured. While effective in controlled environments, such target-dependence imposes considerable operational constraints: it necessitates significant deployment complexity and cost, which hinders frequent recalibration in dynamic settings. Consequently, developing targetless calibration methods has emerged as a critical research focus, aiming to enable flexible, autonomous, and continuous calibration in diverse and uncontrolled environments.

Significant efforts have been dedicated to targetless LiDAR-camera calibration in the past decade, yielding many effective solutions that leverage naturally occurring scene features instead of artificial patterns. Prominent approaches often revolve around establishing cross-modal correspondences based on salient geometric features like edges, or involve formulating joint optimization frameworks that co-register camera images and LiDAR point clouds. However, the inherent sparsity of point clouds and image features limits the calibration accuracy of these methods. Recently, the emergence of powerful scene representation techniques, exemplified by 3D Gaussian Splatting (3DGS) which has drawn great research interest in the past few years [10]–[12], has opened a novel and promising avenue for LiDAR-camera calibration. By facilitating the construction of the photometrically accurate dense 3D models of the scenes, 3DGS inherently enables a more potent integration of visual and geometric information. Calibration algorithms leveraging 3DGS can elegantly exploit the density of both camera pixels and the geometric accuracy of the point cloud from LiDAR, which promises a considerably higher theoretical upper bound on calibration accuracy compared to traditional sparse-feature-based methods.

Recently, some 3DGS-based calibration methods have been proposed, which typically address the extrinsic estimation by jointly optimizing the 3DGS parameters alongside the extrinsics. While leveraging the dense scene representation potential of 3DGS, such joint optimization approaches introduce two aspects of challenges that hinder the robustness and reliability of the calibration:

1) **The global registration paradigm is highly susceptible to accumulative localization errors of the calibration data sequence.** Existing 3DGS-based calibration methods usually employ mature odometry to estimate LiDAR poses and generate global point clouds, subsequently initializing the Gaussian models. These methods

This work was supported in part by the Natural Science Foundation of China under Grant 62225309, Grant U24A20278, Grant 62361166632, Grant U21A20480, and Grant 62503315, the China Postdoctoral Science Foundation under Grant 2025M771737 (Corresponding author: Hesheng Wang).

[1]Tianjun Zhang and Hesheng Wang are with School of Automation and Intelligent Sensing, Shanghai Jiao Tong university and State Key Laboratory of Avionics Integration and Aviation System-of-Systems Synthesis, Shanghai 200240, China (email: {z619850002, wanghesheng}@sjtu.edu.cn).

[2]Lin Zhang is with the School of Computer Science and Technology, Tongji University, Shanghai 201804, China (email: cslinzhang@tongji.edu.cn)

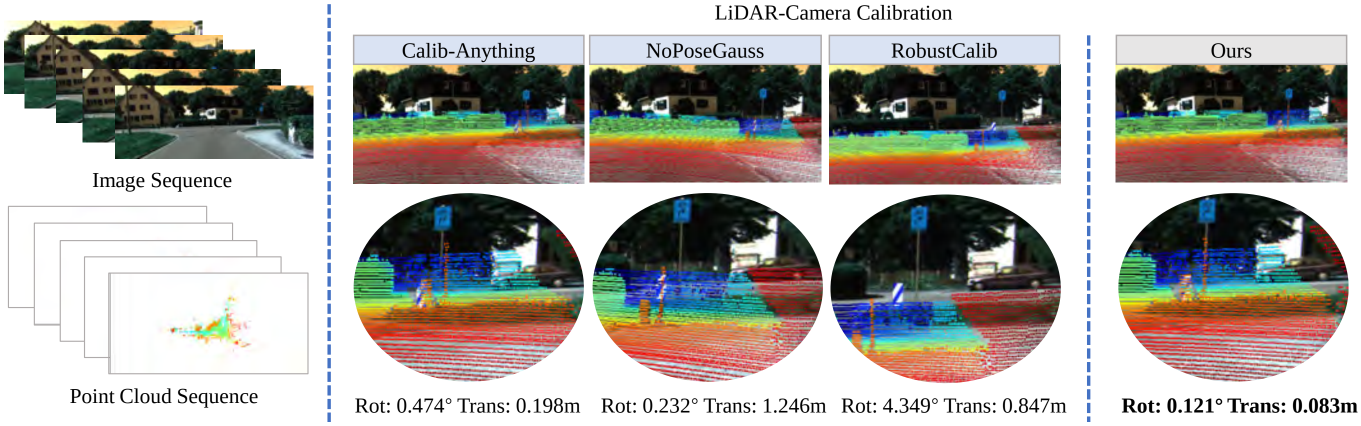


Image Sequence
Point Cloud Sequence
LiDAR-Camera Calibration
Calib-Anything
NoPoseGauss
RobustCalib
Ours
Rot: 0.474° Trans: 0.198m
Rot: 0.232° Trans: 1.246m
Rot: 4.349° Trans: 0.847m
Rot: 0.121° Trans: 0.083m

Fig. 1. **Accurate calibration results from our method.** Focusing on targetless LiDAR-camera calibration, our HiGS-Calib obviously outperforms other compared methods in calibration accuracy.

then jointly optimize models with extrinsics to achieve color registration across global frames. However, the number of calibration frames presents a critical trade-off: insufficient frames lead to calibration overfitting, whereas more frames inevitably introduce inherently accumulative localization errors into the calibration system.

2) **The joint optimization process exhibits inherent convergence instability.** Specifically, since the consistency between extrinsics and the Gaussian model is thoroughly maintained by training, every time the extrinsics are adjusted during joint optimization, the Gaussian model must continuously re-optimize to adapt to the shifted camera poses to further offer the pose-consistent color supervision. Besides, deficient initial poses prevent the Gaussian model from providing spatial-accurate color information, which substantially compounds convergence difficulties for the calibration algorithm. Consequently, the optimization trajectories usually become oscillatory and the training process risks destabilization.

As an attempt to overcome the limitations of existing schemes, we propose a novel hierarchical 3DGS-based LiDAR-camera calibration method, namely HiGS-Calib (**Hi**erarchical **G**aussian **S**platting based **Calib**rator). HiGS-Calib leverages a coarse-to-fine optimization framework seamlessly integrated with our proposed local-consistent projection-based error model. As a targetless calibration model, it can effectively achieve extrinsic calibration with the input of a sequence of images and point clouds collected in natural environments without manual operations. The superior calibration performance of HiGS-Calib is shown in Fig. 1, and the overall architecture of HiGS-Calib is illustrated in Fig. 2. Our contributions can be summarized as follows:

1) A novel projection-based error model, namely LCPG error model (Local-Consistent Photometric and Geometric error) is proposed. Constructed over consecutive frames within a local window, LCPG error can effectively quantify the extrinsic inaccuracy by leveraging only geometric information from the 3DGS model, eliminating dependency on higher-frequency and inaccurate color features. As a projection-based error, LCPG enables calibration to avoid the global accumulated localization error and circumvent the problem of maintaining the color-pose consistency, thereby ensuring optimization robustness and stability throughout the training process.
2) A calibration-oriented hierarchical 3DGS-based optimization framework is presented. Starting at low resolution, such a hierarchical optimization framework follows a coarse-to-fine manner and alternately conducts model-only optimization and pose-geometry joint optimization while progressively scaling the image resolution, thereby effectively stabilizing the extrinsic calibration.
3) A high-accuracy targetless LiDAR-camera calibration algorithm, namely HiGS-Calib, is designed. As a targetless calibration method, HiGS-Calib can autonomously complete the LiDAR-camera calibration with the input of a sequence of images and LiDAR point clouds collected in arbitrary natural scenes. Compared to existing joint-optimization-based methods, HiGS-Calib exhibits significantly improved stability and superior calibration accuracy.
4) Extensive qualitative and quantitative experiments on both representative existing calibration methods and HiGS-Calib are conducted. The experimental results effectively demonstrate the superior calibration accuracy and robustness of our HiGS-Calib.

II. Related Work

A. Targeted LiDAR-camera calibration methods

Targeted LiDAR-camera calibration methods utilize artificial objects with predefined geometries—such as checkerboards, planar targets, or fiducial markers (e.g., ArUco)—to establish precise 2D-3D correspondences between camera pixels and LiDAR points. The pioneering work by Zhang *et al.* [13] solved extrinsics between the camera and the 2D laser range finder using point-to-plane constraints from checkerboard corners in both modalities. Based on Zhang *et al.* 's work [13], Unnikrishnan *et al.* [14] proposed the first calibration framework applicable to cameras and 3D

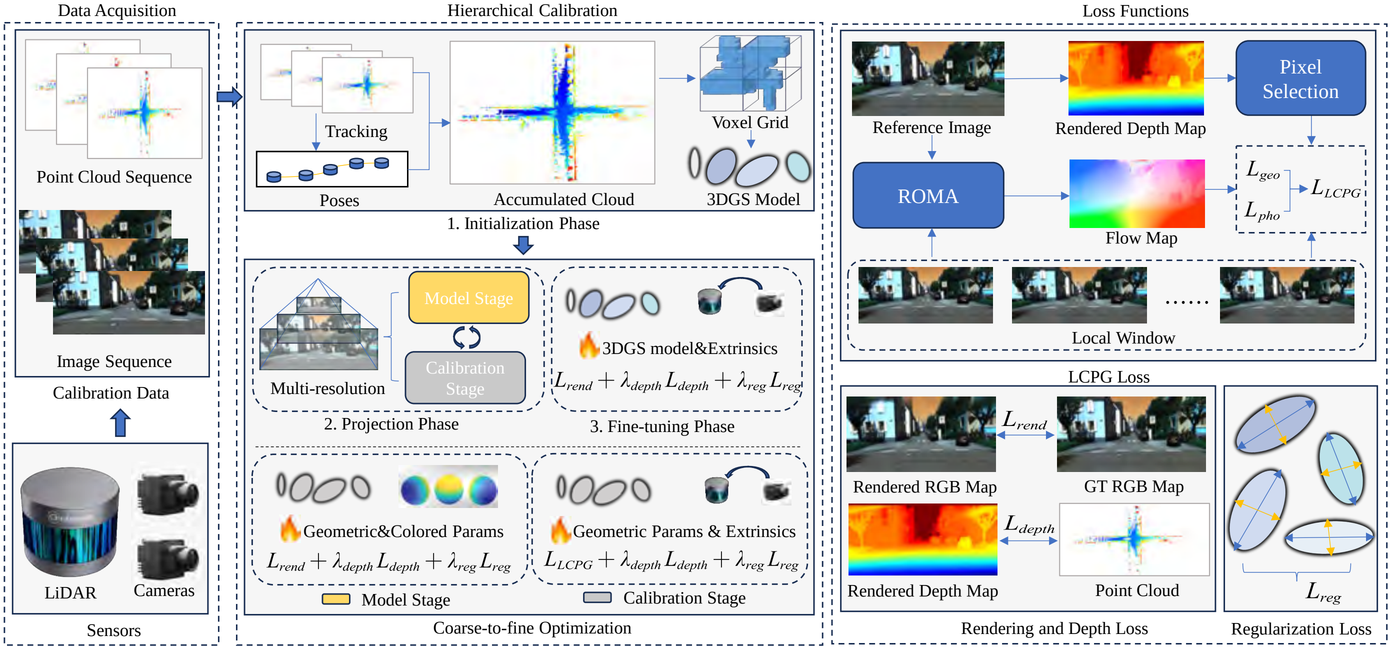


Fig. 2. **The architecture sketch of HiGS-Calib.** Following a hierarchical architecture, HiGS-Calib first initializes the Gaussian model using point clouds and poses recovered from LiDAR odometry. It then proceeds to a two-phase optimization: the projection phase and the fine-tuning phase. During the projection phase optimization, HiGS-Calib employs multi-level optimization, with each level sequentially executing the model stage and the calibration stage optimization. Incorporating our proposed LCPG error, HiGS-Calib achieves superior convergence stability and calibration accuracy.

LiDARs. These two work provided the basic paradigm of targeted LiDAR-camera calibration technology, and subsequent innovations mainly automated the feature extraction and improved the calibration accuracy. Dhall *et al.* [15] enabled automatic 3D corner detection by fusing LiDAR reflections with ArUco marker vertices. Zhou *et al.* [16] simplifies LiDAR-camera calibration using checkerboards by requiring just one board instead of three or more. Guindel *et al.* [17] automated LiDAR-stereo camera calibration without manual intervention, specifically addressing the problem of low sensor resolution and challenging mounting positions. Pusztai *et al.* [18] introduced a practical LiDAR-camera calibration method using common cardboard boxes with known sizes. It can automatically identify box surfaces in LiDAR scans to match with corresponding images, requiring less manual work than previous approaches.

The aforementioned methods are mainly developed for mechanical LiDARs, while ACSC [19] adapted Perspective-n-Point (PnP) solvers for solid-state LiDARs with non-repetitive scanning patterns to calibrate cameras and solid-state LiDARs. Similarly, Dong *et al.* [20] proposed an automatic extrinsic calibration method using pseudo-images and multiple manmade targets, specifically designed for vehicle-mounted solid-state LiDARs. In [21], Yan *et al.* proposed an open-source toolbox with multi-scenario calibration capabilities (including manual, automatic, factory, and online methods) for autonomous vehicles, namely OpenCalib, accompanied by a benchmark dataset for accuracy evaluation. On account of its extensive calibration methodology coverage, OpenCalib has emerged as a widely adopted benchmark implementation for landmark-assisted multi-sensor calibration.

While targeted calibration methods can achieve high accuracy and stability, their significant operational costs and reliance on specific physical targets limit their practical applications in unstructured natural environments or real-time operational scenarios. These inherent constraints have consequently prompted researchers to actively pursue the development of targetless calibration algorithms.

B. Targetless LiDAR-camera calibration methods

Targetless calibration methods eliminate dependencies on artificial markers, enabling flexible deployment in natural or unstructured environments. Among them, geometric-feature-based approaches constitute a major category, which calibrate sensors by extracting and matching explicit geometric features (e.g., points, lines) between cameras and LiDARs. Moghadam *et al.* [22] pioneered line-based calibration by matching natural 3D line segments from LiDAR scans to 2D edges in images, jointly optimizing extrinsic parameters. Liu *et al.* [23] accelerated multi-LiDAR-camera calibration via adaptive voxelization, which efficiently extracts LiDAR feature points (planes/edges) while avoiding redundant k-d tree construction, ultimately reformulating the calibration as a bundle adjustment problem. Koide *et al.* [24] proposed a general toolbox that establishes 2D-3D correspondences using Super-Glue [25] for initial guess, followed by edge-alignment refinement. Such geometric-feature-based methods are straightforward and generally efficient. However, relying heavily on detectable geometric structures, such methods may struggle with feature scarcity in textureless or unstructured environments.

To address the limitations of geometric methods, non-geometric methods leverage cross-modal consistency metrics without explicit feature extraction and matching. Foundational work by Pandey *et al.* [26] utilized mutual information maximization between LiDAR reflectance and camera image intensities. Ishikawa *et al.* [27] introduced intensity variance minimization across image sequences to optimize extrinsics for sparse LiDARs, while Ou *et al.* [28] resolved motion degeneracy (e.g., rotation-limited vehicles) via structural alignment constraints. To further improve the calibration robustness, some data-driven methods have also been proposed. Liu *et al.* [29] designed a Transformer-based network with local-to-global constraints for 6-DoF extrinsics regression. Luo *et al.* [30] exploited the Segment Anything Model (SAM) [31] to generate geometric masks automatically, optimizing extrinsics by maximizing point attribute (intensity, normal vectors) consistency within these masks. These cross-modal consistency approaches avoid the requirement for explicit geometric feature extraction. However, their calibration accuracy is usually limited in practice due to the sparsity of LiDAR point clouds, particularly the low resolution inherent in traditional mechanical LiDARs.

C. Neural rendering based calibration methods

In recent years, differentiable neural rendering techniques have gained significant attention. Representative approaches mainly include Neural Radiance Field (NeRF) [32] and 3DGS [33]. As emerging 3D scene representations, both NeRF [32] and 3DGS [33] possess fine-grained rendering capabilities beyond conventional methods, enabling high-accuracy data matching, and have gradually started to be applied in the field of sensor calibration. As a milestone work, NeRF was proposed by Mildenhall *et al.* [32] in 2020. In [32], the static scene is modeled as a radiance field, and novel views can be synthesized based on volume rendering techniques. Based on NeRF, some neural-rendering-based calibration methods have also been proposed. Zhou *et al.* [34] proposed INF, training separate neural density and color fields while jointly optimizing poses and extrinsics. Building on volumetric representations, Herau *et al.* [35] presented SOAC, partitioning scenes by sensor visibility for robust optimization using only overlapping areas. To address scalability challenges, Yang *et al.* [36] developed UniCal, unifying sensor calibration through differentiable rendering with a surface alignment loss that ensures multi-view consistency across LiDARs and cameras. These approaches can maintain multi-sensor consistency through unified scene representations and complete the calibration tasks without specific calibration targets. However, the suboptimal training and rendering speeds of NeRF usually result in time-consuming calibration processes.

As another mainstream neural rendering technique, 3DGS [33] differs from NeRF [32] by abandoning ray-tracing in favor of rasterization, delivering significantly faster rendering and training speeds. Due to these advantages, 3DGS has emerged as a key research focus and is increasingly being applied across diverse fields [37]–[39]. Similar to NeRF, 3DGS has also been applied to LiDAR-camera calibration. Herau *et al.* developed 3DGS-Calib for quick and accurate multi-sensor spatiotemporal calibration [40]. Such an approach sidesteps slow volume rendering through efficient 3DGS scene modeling, achieving significant speed gains while maintaining accuracy. Schmidt *et al.* [41] proposed a pose-free extension to 3DGS that jointly optimizes 3D scene geometry and camera poses by deriving analytical gradients of photometric residuals and integrating them into a high-performance CUDA pipeline, enabling fast, high-quality novel-view synthesis without accurate initial poses. With moderate adaptation, such a framework could also be extended to the LiDAR-camera calibration task. Separately, Zhou *et al.* [43] adapted 2D Gaussian Splatting [42] for LiDAR-camera systems. Their method reconstructs geometry from LiDAR clouds, then optimizes extrinsics by updating Gaussian colors with photometric loss and geometric constraints. 3DGS-based calibration methods offer significant speed advantages over NeRF-based methods thanks to the efficient rendering. However, the aforementioned existing algorithms [40], [41], [43] rely heavily on the joint optimization of sensor poses and Gaussian models during training. Once camera poses alter, the Gaussian model needs to be retrained to adapt to the current state before it can provide further effective guidance for pose optimization, which often leads to unstable training convergence and robustness challenges in practice.

III. Design Motivations of HiGS-Calib

In this section, the preliminaries of 3D Gaussian Splatting is introduced first. After that, some limitations of existing calibration algorithms are analyzed, and then the design motivations behind HiGS-Calib are presented.

A. 3DGS preliminaries

Represented in 3DGS form, the scene $\mathcal{S}$ can be considered as a group of Gaussian distributions, $\mathcal{S} = \{\boldsymbol{G}_i\}_{i=1}^{N}$, where N is the number of Gaussians. Each Gaussian in the model has multiple attributes, which can be given as,

$$\boldsymbol{G}_i = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \boldsymbol{c}_i, \alpha_i\}, \tag{1}$$

where $\boldsymbol{\mu}_i$ is the expectation of the Gaussian distribution, $\boldsymbol{\Sigma}_i$ is the covariance of the distribution, $\boldsymbol{c}_i$ and α_i are the color and opacity, respectively. To parametrize the distribution $\boldsymbol{G}_i$, the covariance matrix $\boldsymbol{\Sigma}_i$ can be further decomposed as,

$$\boldsymbol{\Sigma}_i = \boldsymbol{R}_i \boldsymbol{s}_i \boldsymbol{s}_i^T \boldsymbol{R}_i^T, \tag{2}$$

where $\boldsymbol{s}_i \in \mathbb{R}^3$ is the scale of the distribution, and $\boldsymbol{R}_i \in \mathbb{R}^{3\times3}$ is the rotation, which can usually be represented as a quaternion $\boldsymbol{q}_i \in \mathbb{R}^4$.

Given a camera with pose of $\boldsymbol{T}_{cw}$, the 3D Gaussian $\boldsymbol{G}_i$ can be projected onto the image plane as $\boldsymbol{G}_i^{2D}$, and the expectation and covariance of $\boldsymbol{G}_i^{2D}$ can be computed by the projection function. Finally, the image $\boldsymbol{I}$ can be rendered through the rasterization pipeline, and the rendered color at $\boldsymbol{p}$ is given as,

$$\boldsymbol{I}(\boldsymbol{p}) = \sum_{i=1}^{N} \boldsymbol{c}_i \alpha_i \boldsymbol{G}_i^{2D}(\boldsymbol{p}) \prod_{j=1}^{i-1} (1 - \alpha_j \boldsymbol{G}_j^{2D}(\boldsymbol{p})). \tag{3}$$

Similarly, the corresponding depth map $\boldsymbol{D}$ can also be rendered as,

$$\boldsymbol{D}(\boldsymbol{p})=\sum_{i=1}^{N} d_i \alpha_i \boldsymbol{G}_i^{2D}(\boldsymbol{p}) \prod_{j=1}^{i-1}(1-\alpha_j \boldsymbol{G}_j^{2D}(\boldsymbol{p})), \tag{4}$$

where d_i is the depth of $\boldsymbol{G}_i$ in the camera coordinate system. The construction of 3DGS models is achieved by training parameters to minimize rendering loss. Specifically, during the training of a standard 3DGS model, RGB images $\mathcal{I}=\{\boldsymbol{I}_t\}_{t=1}^{T}$ captured from different viewpoints and their corresponding camera poses $\mathcal{T}=\{\boldsymbol{T}_t\}_{t=1}^{T}$ are known, while the 3D Gaussian model $\mathcal{G}=\{\boldsymbol{G}_i\}_{i=1}^{N}$ remains unknown. After the 3DGS model $\mathcal{G}$ is initialized, images $\mathcal{I}^{rend}=\{\boldsymbol{I}_t^{rend}\}_{t=1}^{T}$ can be rendered from the original camera viewpoints through the end-to-end rendering pipeline. The photometric loss is then constructed between the rendered images $\mathcal{I}^{rend}$ and the ground-truth images $\mathcal{I}$. By optimizing the parameters of the 3DGS model to minimize this rendering loss, a high-quality 3DGS model is ultimately obtained, enabling free-viewpoint rendering. The objective goal of training a 3DGS model can be given as,

$$\mathcal{G}^*=\arg\min_{\mathcal{G}} Loss\left(\mathcal{I},\mathcal{I}^{rend}\right). \tag{5}$$

In some variant tasks (such as camera calibration), the camera poses $\mathcal{T}$ are also treated as unknown variables to be optimized simultaneously during model training. Such a training-and-rendering pipeline establishes a theoretical foundation for 3DGS-based camera-LiDAR extrinsic calibration, as its fully differentiable pipeline enables joint optimization of Gaussian parameters and sensor poses during training.

B. Design motivations

The most mainstream 3DGS-based methods usually resort extrinsics calibration to the joint optimization of both Gaussian models and extrinsics based on all images and point clouds. The goal of such an optimization can be considered as,

$$\mathcal{G}^*,\boldsymbol{T}_{cl}^*=\arg\max_{\mathcal{G},\boldsymbol{T}_{cl}} P(\mathcal{G},\boldsymbol{T}_{cl}|\mathcal{I},\mathcal{P},\mathcal{T}_l), \tag{6}$$

where $\mathcal{G}$ is the set of all parameters of the Gaussian model, $\boldsymbol{T}_{cl}$ represents the extrinsics from the LiDAR coordinate system to the camera system, $\mathcal{I}$ and $\mathcal{P}$ are sets of all calibration images and point clouds, respectively, and $\mathcal{T}_l$ is the set of LiDAR poses of all calibration frames (including $\boldsymbol{T}_{wl}$ at different timestamps), which are usually estimated by mature LiDAR odometry or SLAM methods [44], [45].

Through comprehensive analysis, we found that in such a joint pose-model optimization process, there actually exist two potential pitfalls:

1) **The joint optimization of all data is usually sensitive to the accumulated error of LiDAR poses.** To achieve robust calibration, 3DGS-based calibration algorithms typically require dozens of frames to fit the Gaussian model. However, LiDAR poses inevitably contain accumulative localization errors that cannot be effectively corrected. The global optimization approach actually implicitly introduces extensive long-term color data associations, which often become unreliable due to accumulated localization errors, ultimately compromising the calibration accuracy.
2) **The inaccurate extrinsics may cause oscillations during joint optimization.** Specifically, during each step of the optimization, the gradient for extrinsics $\boldsymbol{T}_{cl}$ is computed using the current Gaussian model state $\mathcal{G}$, while $\mathcal{G}$ itself is fitted based on the wrong state of $\boldsymbol{T}_{cl}$. Since the extrinsics are inaccurate, the Gaussian model may not offer reliable guidance for the extrinsics' update. Additionally, when poses change during optimization, the color information in the Gaussian model must be retrained to match the new viewpoint. These interdependent updates typically result in oscillatory behavior and may cause extrinsics to converge to the local optimum.

To address the limitations of the aforementioned joint optimization methods, we aim to design a new error model to replace the rendering error of Gaussian models, and further construct a new algorithm for LiDAR-camera calibration. Such a calibration algorithm should minimize long-term data associations in favor of short-term correlations, thereby reducing interference from accumulated localization errors. Moreover, even with inaccurate extrinsics, LiDAR point clouds still provide reliable geometric supervision for the Gaussian model. We can therefore bypass color information and exclusively utilize geometric representations. Decomposing the parameters of the Gaussian model $\mathcal{G}$ into two groups, geometry parameters $\mathcal{G}_g$ and color parameters $\mathcal{G}_c$, the goal of the optimization in Eq. 6 can then be reformulated as,

$$\begin{aligned}
\mathcal{G}^*,\boldsymbol{T}_{cl}^* &= \arg\max_{\mathcal{G},\boldsymbol{T}_{cl}} P_c\cdot P_T\cdot P_g\\
P_g &= P(\mathcal{G}_g|\mathcal{I},\mathcal{P},\mathcal{T}_l)\\
P_T &= P(\boldsymbol{T}_{cl}|\mathcal{G}_g,\mathcal{I},\mathcal{P},\mathcal{T}_l)\\
P_c &= P(\mathcal{G}_c|\boldsymbol{T}_{cl},\mathcal{G}_g,\mathcal{I},\mathcal{P},\mathcal{T}_l).
\end{aligned} \tag{7}$$

In fact, through such a decomposition, we can approximately maximize $P(\mathcal{G},\boldsymbol{T}_{cl}|\mathcal{I},\mathcal{P},\mathcal{T}_l)$ by independently maximizing P_c, P_T and P_g, thereby circumventing the need for global joint optimization. Among them, maximizing P_T enables reliable estimation of the extrinsics $\boldsymbol{T}_{cl}$ using well-constrained variables without relying on the unconverged and inaccurate $\mathcal{G}_c$. This can effectively mitigate convergence oscillation and reduce the computational cost of maintaining color-pose consistency, thereby enhancing calibration stability. Standing on the motivations above, we propose a hierarchical calibration framework (introduced in Sec. IV) to maximize P_c, P_T and P_g iteratively so as to mitigate the global optimization. Besides, a projection-based error model, namely LCPG error model whose minimization is equivalent to maximizing P_T, is also presented (introduced in Sec. V).

IV. Hierarchical LiDAR-camera calibration

In this section, we provide a detailed introduction to our HiGS-Calib calibration algorithm featuring its hierarchical structure. To show our HiGS-Calib more intuitively, the illustration of its hierarchical pipeline is given in Fig. 3. As illustrated in Fig. 3, there are actually three phases in HiGS-Calib, including the initialization phase, the projection phase

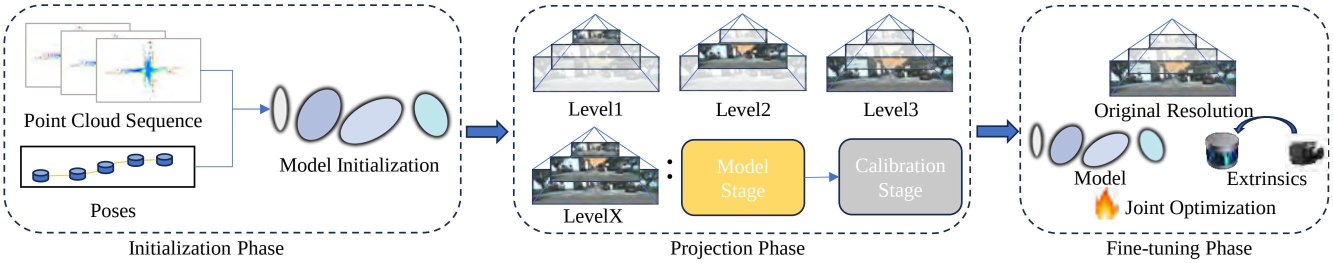


Fig. 3. **Illustration of the hierarchical framework of HiGS-Calib.** HiGS-Calib has three calibration phases: initialization phase, projection phase, and fine-tuning phase. In the projection phase, there are three levels of optimization conducted on different image resolutions, and each level is composed of two stages of optimization: the model stage and the calibration stage.

and the fine-tuning phase. In the projection phase, three levels of optimization are conducted, and in each level, there are two stages: the model stage and the calibration stage. Next, we will introduce our hierarchical framework in detail.

A. Gaussian model initialization

Similar to existing neural-rendering-based calibration methods, given a group of LiDAR point clouds $\mathcal{P} = \{\mathcal{P}_t\}_{t=1}^N$, the corresponding pose $\boldsymbol{T}_{wl}^t$ of each LiDAR frame $\mathcal{P}_t$ can be extracted using established LiDAR odometries or SLAM algorithms [44], [45]. Subsequently, all point clouds are aligned and transformed into the world coordinate system to form the global cloud $\mathcal{P}_w$, which then initializes the 3D Gaussian model. Specifically, all points in $\mathcal{P}_w$ are first assigned to voxels of size v. For any voxels with points, a Gaussian distribution is constructed. If there is only one point in the voxel, the expectation of the distribution is set to the position of the voxel, the scales on all axes are set to v, and the rotations are set to identity rotations. For voxels with more than one point, related attributes of the distribution can be fitted by the internal points. Specifically, taking voxel $\boldsymbol{v}$ as an example, the expectation $\boldsymbol{\mu}_v$ is computed as,

$$\boldsymbol{\mu}_v = \frac{1}{N_v}\sum_{i=1}^{N_v}\boldsymbol{P}_i^v, \tag{8}$$

where N_v is the number of points in voxel $\boldsymbol{v}$, and $\boldsymbol{P}_i^v$ is the i_{th} 3D point in voxel $\boldsymbol{v}$. The covariance matrix $\boldsymbol{\Sigma}_v$ of the Gaussian distribution can be computed as,

$$\boldsymbol{\Sigma}_v = \frac{1}{N_v}\sum_{i=1}^{N_v}(\boldsymbol{P}_i^v - \boldsymbol{\mu}_v)(\boldsymbol{P}_i^v - \boldsymbol{\mu}_v)^T. \tag{9}$$

Performing eigenvalue decomposition on $\boldsymbol{\Sigma}_v = \boldsymbol{R}_v\boldsymbol{D}_v\boldsymbol{R}_v^T$, the rotation of the distribution is set to $\boldsymbol{R}_v$, and the scale $\boldsymbol{s}_v$ can be set to,

$$\boldsymbol{s}_v = Sqrt(Diag(\boldsymbol{D}_v)), \tag{10}$$

where $Diag(\cdot)$ represents the diagonal vector of the internal matrix.

B. Hierarchical calibration pipeline

As a hierarchical calibration pipeline, HiGS-Calib implements three distinct optimization phases: the initialization phase, the projection phase and the fine-tuning phase. The initialization phase has already been introduced in Sec. IV-A. In the projection phase, three levels of optimization are performed on progressively recovered image resolutions. The image resolution scales relative to the original resolution in these levels are 0.25×, 0.50×, and 1.00×, respectively. Since the projection-based LCPG error is introduced during optimization (which will be introduced in detail in Sec. V), such a phase is called the "projection phase". After these three levels of LCPG-based optimization in the "projection phase", since the extrinsics have already been relatively accurate, a standard joint optimization of both extrinsics and the Gaussian model is performed for further fine-tuning, which is called the "fine-tuning phase". Such a coarse-to-fine optimization architecture can significantly enhance the calibration algorithm's stability and robustness against noise, outliers, and local minima.

The optimization goal in the projection phase of HiGS-Calib derives from the objective function specified in Eq. 7 with critical modifications. Specifically, while the initialization phase provides an initial geometric estimation of the Gaussian model (maximizing P_g in Eq. 7), the projection phase focuses on maximizing P_c and P_T. However, LiDAR point clouds usually exhibit sparse granularity and the geometric accuracy of the initialized 3DGS model is usually limited. Consequently, we reformulate Eq. 7 to leverage the geometric refinement capabilities of both LCPG and rendering errors. Specifically, we synchronously optimize the geometric parameters during the optimization on extrinsics and color attributes of the Gaussian model, and the goal at each level of the projection phase can be given as,

$$\begin{aligned} \mathcal{G}^*, \boldsymbol{T}_{cl}^* &= \arg\max_{\mathcal{G}, \boldsymbol{T}_{cl}} P_{cg}\cdot P_{Tg} \\ P_{cg} &= P(\mathcal{G}_c, \mathcal{G}_g|\boldsymbol{T}_{cl}, \mathcal{I}, \mathcal{P}, \mathcal{T}_l) \\ P_{Tg} &= P(\boldsymbol{T}_{cl}, \mathcal{G}_g|\mathcal{I}, \mathcal{P}, \mathcal{T}_l). \end{aligned} \tag{11}$$

In each level of the projection phase, there are two stages of optimization, including the model stage and the calibration stage. In the model stage, the posterior P_{cg} is maximized by minimizing loss L_{cg}, and the calibration optimization stage minimizes loss L_{Tg} to maximize P_{Tg}. The definition of L_{cg} and L_{Tg} will be introduced in Sec. IV-C.

C. Loss functions

Except for the LCPG error based loss L_{LCPG}, which will be introduced in Sec. V, there are also three types of loss involved in the calibration process of HiGS-Calib, including the rendering loss L_{rend}, the depth loss L_{depth} and the regularization loss L_{reg}. Next, taking image I_t as an example, these losses will be introduced one by one.

Rendering loss. Rendering loss is defined as the difference between the calibration image and the corresponding rendered image of the Gaussian model. The rendering loss L_{rend} of image I_t is defined as,

$$L_{rend} = (1 - \lambda_{ssim})|I_t, I_t^{rend}| + \lambda_{ssim}(1 - SSIM(I_t, I_t^{rend})), \tag{12}$$

where I_t^{rend} is the rendered image of the Gaussian model corresponding to I_t, $SSIM(*)$ represents the structure similarity index measure [47], λ_{ssim} is the weight of the SSIM loss.

Depth loss. Depth loss is actually used to constrain the geometric structure of the Gaussian model by LiDAR observations. Specifically, for frame t, the points in the LiDAR observations $\mathcal{P}_t$ are projected onto the virtual image plane with pose $\boldsymbol{T}_{vw}^t$ to get the virtual depth map $D_{virtual}^t$. Take a point $\boldsymbol{P}_t$ in $\mathcal{P}_t$ as an example. Transform $\boldsymbol{P}_t$ into the camera coordinate system of the virtual camera by $\boldsymbol{T}_{vw}^t$ to get $\boldsymbol{P}_t^{virtual}$, and then project $\boldsymbol{P}_t^{virtual}$ onto the virtual image plane by the intrinsics of the camera to get $\boldsymbol{p}_t^{virtual}$. The virtual depth map can be set to,

$$D_{virtual}^t(\boldsymbol{p}_t^{virtual}) = \left[\boldsymbol{P}_t^{virtual}\right]_z, \tag{13}$$

where $[\cdot]_z$ stands for the Z-axis value of the internal point. $D_{virtual}^t$ can then constrain the geometric structure of the Gaussian model by introducing the inverse depth loss,

$$L_{depth} = |1/D_{virtual}^t(M_t), 1/D_{rend}^t(M_t)|, \tag{14}$$

where $D_{rend}^t(M_t)$ is the rendered depth map at $\boldsymbol{T}_{vw}^t$ of the Gaussian model, and M_t is the constraint mask in which only pixels with projections from the point cloud are set to true. For the pose $\boldsymbol{T}_{vw}^t$ of the virtual camera, we leverage the rotation $\boldsymbol{R}_{cw}^t$ from the current camera frame to enhance the surface smoothness, while employing LiDAR-derived translation $\boldsymbol{t}_{lw}^t$ to mitigate occlusion artifacts.

Regularization loss. To prevent excessively elongated distributions in the 3DGS model and enhance calibration robustness, the regularization loss L_{reg} is introduced to our HiGS-Calib, which is given as,

$$L_{reg} = \frac{1}{|\sqsubseteq_t|} \sum_{i \in \sqsubseteq_t} \max(\frac{\max(\boldsymbol{s}_i)}{\min(\boldsymbol{s}_i)} - s_{reg}, 0), \tag{15}$$

where $\sqsubseteq_t$ is the set of indices of Gaussian distributions that are visible in the view-frustum of the current frame, $\boldsymbol{s}_i$ is the scale of the corresponding Gaussian distribution, and s_{reg} is the scale threshold for the regularization loss.

Training loss in different stages. In the model stage of each level of the projection phase, the parameters of the Gaussian model are trained to maximize the posterior P_{cg} in Eq. 11, and extrinsics $\boldsymbol{T}_{cl}$ is set fixed. The loss function L_{model} is given as,

$$L_{model} = L_{rend} + \lambda_{depth} \cdot L_{depth} + \lambda_{reg} \cdot L_{reg}, \tag{16}$$

where λ_{depth} and λ_{reg} are hyperparameters to control the weights of corresponding losses. In the calibration stage of the projection phase, $\boldsymbol{T}_{cl}$ is optimized, and the training loss L_{calib} is given as,

$$L_{calib} = L_{LCPG} + \lambda_{depth} \cdot L_{depth} + \lambda_{reg} \cdot L_{reg}. \tag{17}$$

It's worth noticing that, though the color parameters of the Gaussian model are not explicitly set fixed, the loss function is irrelevant to them and thus they won't be updated in such phases. Besides, in the final fine-tuning phase, the model loss L_{model} is also used as the training loss. Compared with the model stage, the only difference is that $\boldsymbol{T}_{cl}$ is also optimized.

D. Design details

Optimizable poses. The LiDAR-camera extrinsics $\boldsymbol{T}_{cl}$ comprise two components: rotation $\boldsymbol{R}_{cl}$ and translation $\boldsymbol{t}_{cl}$. In HiGS-Calib, the rotation matrix $\boldsymbol{R}_{cl}$ is parameterized as a quaternion $\boldsymbol{q}_{cl}$ to remove internal constraints [48]. Besides, to guarantee numerical stability, $\boldsymbol{q}_{cl}$ is constructed as $\boldsymbol{q}_{cl} = \hat{\boldsymbol{q}}_{cl} \otimes \delta\boldsymbol{q}_{cl}$, where $\otimes$ denotes quaternion multiplication. During training, the base component $\hat{\boldsymbol{q}}_{cl}$ is fixed, while the delta variable $\delta\boldsymbol{q}_{cl}$ is optimized. Each time the rotation is updated, $\hat{\boldsymbol{q}}_{cl}$ is set to $\hat{\boldsymbol{q}}_{cl} \otimes \delta\boldsymbol{q}_{cl}$ and then normalized. After that, the delta variable $\delta\boldsymbol{q}_{cl}$ is set to the identity quaternion.

Delayed pose update. During calibration, pose optimization suffers from noticeable fluctuations as gradients are backpropagated using only one randomly selected image per iteration. Moreover, constructing the LGPG loss requires rendered depth maps from both the current frame and neighboring frames in a local window for pixel selection, while updating these depth maps per iteration would impose prohibitive computational costs. To address these challenges, we propose a delayed update strategy: gradients accumulate after each backward pass but poses remain unchanged, and only after every N_{delay} iterations are poses updated using aggregated gradients. This strategy significantly stabilizes pose optimization. Simultaneously, depth maps needed for the LGPG loss are pre-rendered in bulk after each time of the pose update. Since poses remain fixed between updates, depth maps remain valid during this interval, requiring updates only when poses are refreshed, which also drastically boosts computational efficiency.

V. Local-Consistent Projective Error

In this section, a detailed introduction to our proposed Local-Consistent Photometric and Geometric error model (LCPG) is given, which can effectively quantify the extrinsic inaccuracy by leveraging only geometric information rather than colors from the 3DGS model. Specifically, the specific definition of the LCPG model is offered first. After that, the corresponding pixel selection strategy is introduced.

A. Basic form of LCPG error model

LCPG is an error model that quantifies the extrinsics' inaccuracy by both photometric and geometric projective errors on frames in the local window. From the probabilistic perspective, minimizing the LCPG error equals to maximizing the distribution P_T in Eq. 7. In each iteration of the training process, one calibration image I_t at timestamp t is chosen as

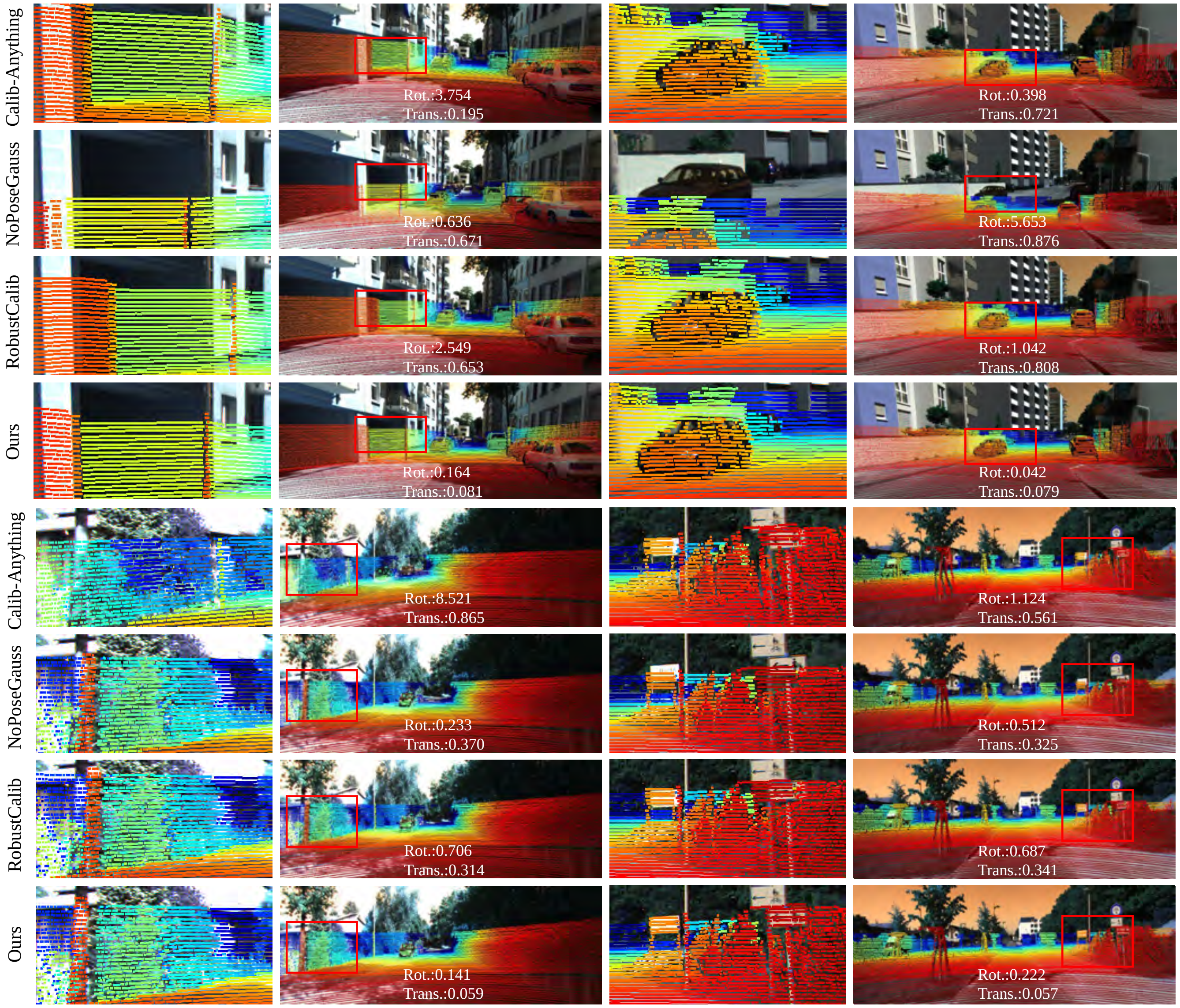


Fig. 4. **Typical samples of the calibration visualization results of the compared methods.** To show the calibration accuracy, the point clouds are projected to images using the extrinsics yielded by the compared methods. In each group of data, the full views are shown on the right and the enlarged local ROIs are on the left. The calibration rotation errors and translation errors are also plotted on the figures. The results of the top two scenes are from the KITTI360 dataset [49], and the bottom two are from the KITTI dataset [50].

the ground-truth image to construct the LCPG error L_{LCPG}^t, which is given as,

$$L_{LCPG}^t = \sum_{s=-W}^{W} L_{pho}(t, t+s) + \lambda_{geo} L_{geo}(t, t+s), \quad (18)$$

where W is the radius of the local window, λ_{geo} is the hyperparameter to control the weight of the geometric error, $L_{pho}(t, t+s)$ and $L_{geo}(t, t+s)$ are photometric and geometric errors between frame I_t and I_{t+s}, respectively.

To construct both the photometric error and the geometric one, the depth map D_t at timestamp t need to be rendered first,

$$D_t = rend(\mathcal{G}, T_{cw}^t), \quad (19)$$

where $rend(\cdot)$ is the depth-map rendering pipeline of 3DGS (given in Eq. 4), and T_{cw}^t is the camera pose at timestamp t, which can be computed by both the LiDAR pose T_{wl}^t and the extrinsics T_{cl}. Based on the depth map D_t, a pixel $\boldsymbol{p}_t$ on I_t can be projected onto image I_{t+s} as $\hat{\boldsymbol{p}}_{t+s}$, which can be given as,

$$\hat{\boldsymbol{p}}_{t+s} = \mathcal{W}(\boldsymbol{p}_t, D_t, \boldsymbol{T}_{cw}^t, \boldsymbol{T}_{cw}^{t+s}, \boldsymbol{K}), \quad (20)$$

where $\boldsymbol{K}$ is the intrinsic matrix of the camera, and $\mathcal{W}(\cdot)$ is the warp function, which first lifts $\boldsymbol{p}_t$ to the world coordinate system and then projects onto the image plane of frame I_{t+s}. Based on this warp function, the photometric error term $l_{pho}(\boldsymbol{p}_t, \hat{\boldsymbol{p}}_{t+s})$ between $\boldsymbol{p}_t$ and $\hat{\boldsymbol{p}}_{t+s}$ can be given as,

$$l_{pho}(\boldsymbol{p}_t, \hat{\boldsymbol{p}}_{t+s}) = |I_t(\boldsymbol{p}_t), I_{t+s}(\hat{\boldsymbol{p}}_{t+s})|, \quad (21)$$

where $|\cdot|$ represents the l_1 loss.

As a differentiable loss, the derivatives of the photometric loss with respect to the pose depend on local image gradients, while the intensity profiles of natural images typically exhibit strong non-convexity. Thus, to make the calibration more stable, we also introduce the flow-based geometric loss. In our LCPG error model, ROMA [46] is applied to extract the optical flow F_t^{t+s} from I_t to I_{t+s}, and the geometric loss term $l_{geo}(\boldsymbol{p}_t, \hat{\boldsymbol{p}}_{t+s})$ between $\boldsymbol{p}_t$ and $\hat{\boldsymbol{p}}_{t+s}$ can be given as,

$$l_{geo}(\boldsymbol{p}_t, \hat{\boldsymbol{p}}_{t+s}) = |\boldsymbol{p}_t + F_t^{t+s}(\boldsymbol{p}_t), \hat{\boldsymbol{p}}_{t+s}|. \tag{22}$$

By summarizing the loss terms of all qualified pixels, the photometric error $L_{pho}(t, t+s)$ and geometric error $L_{geo}(t, t+s)$ can be given as,

$$\begin{aligned} L_{pho}(t, t+s) &= \sum_{\boldsymbol{p}_t \in \mathcal{PS}_t} W_{pho}(\boldsymbol{p}_t) \cdot l_{pho}(\boldsymbol{p}_t, \hat{\boldsymbol{p}}_{t+s}) \\ L_{geo}(t, t+s) &= \sum_{\boldsymbol{p}_t \in \mathcal{PS}_t} l_{geo}(\boldsymbol{p}_t, \hat{\boldsymbol{p}}_{t+s}), \end{aligned} \tag{23}$$

where $\mathcal{PS}_t$ is the set of all qualified pixels on I_t selected by our pixel selection strategy, which is introduced in Sec. V-B, and $W_{pho}(\boldsymbol{p}_t)$ is the gradient compensation weight of $\boldsymbol{p}_t$. Actually, the photometric errors corresponding to pixels with lower gradients are more insensitive to the pose change. Thus, to achieve more accurate and robust calibration, larger weights are assigned to them, and $W_{pho}(\boldsymbol{p}_t)$ is given as,

$$W_{pho}(\boldsymbol{p}_t) = 1 + (1 - G_t(\boldsymbol{p}_t)) = 2 - G_t(\boldsymbol{p}_t), \tag{24}$$

where G_t is the normalized gradient map of I_t.

It's worth mentioning that, ROMA [46] is currently one of the most stable and advanced optical flow algorithms available, and it generally handles moderate illumination variations and motion blur effectively, that's why we chose it. Actually, our method itself does not depend on any specific optical flow algorithm, but is designed as an extensible framework. Should more robust optical flow or feature matching algorithms emerge in the future, it's also feasible to integrate them to further improve the performance of HiGS-Calib.

B. Pixel selection strategy

Given the calibration image I_t, not all pixels but pixels selected by our pixel selection strategy on I_t are used in constructing the LCPG error. Fundamentally, our pixel selection strategy incorporates three technical components: depth validation, flow confidence assessment, and occlusion masking.

Depth validation. To ensure the stability of calibration, we only select pixels to build the LCPG error if their corresponding depth values lie within a practical range. Specifically, any pixel $\boldsymbol{p}_t$ in the qualified pixel set $\mathcal{PS}_t$ must satisfy,

$$D_{min} < D_t(\boldsymbol{p}_t) < D_{max}. \tag{25}$$

Flow confidence assessment. While the ROMA flow network does not produce accurate optical flow estimates for every pixel, it provides confidence maps for its predictions. We leverage these confidence maps to filter pixels, permitting only those exceeding the threshold to contribute to the LCPG error,

$$C_t^{t+s}(\boldsymbol{p}_t) > C_{min}, \tag{26}$$

where C_t^{t+s} is the corresponding confidence map of the flow estimation F_t^{t+s}.

Occlusion masking. While pixel correspondences derived from inter-frame projection in the LCPG are generally correct, mismatches can occur at the boundaries between foreground and background objects due to occlusion. To maintain calibration accuracy, pixels adversely affected by occlusion must be filtered out. Specifically, the $\boldsymbol{p}_t$'s corresponding 3D point $\hat{\boldsymbol{P}}_{t+s}$ in the camera coordinate system of I_{t+s} is given as,

$$\hat{\boldsymbol{P}}_{t+s} = \boldsymbol{T}_{cw}^{t+s}(\boldsymbol{T}_{cw}^{t})^{-1} D_t(\boldsymbol{p}_t)\boldsymbol{K}^{-1}\boldsymbol{p}_t, \tag{27}$$

where D_t is the corresponding rendered depth map of I_t, and the homogeneous-cartesian conversion is implied. Any qualified pixel $\boldsymbol{p}_t$ in $\mathcal{PS}_t$ must satisfy,

$$\left[\hat{\boldsymbol{P}}_{t+s}\right]_z < (1 + r_o) D_{t+s}(\hat{\boldsymbol{p}}_{t+s}), \tag{28}$$

where r_o is the occlusion tolerance ratio, and D_{t+s} is the corresponding rendered depth map of I_{t+s}.

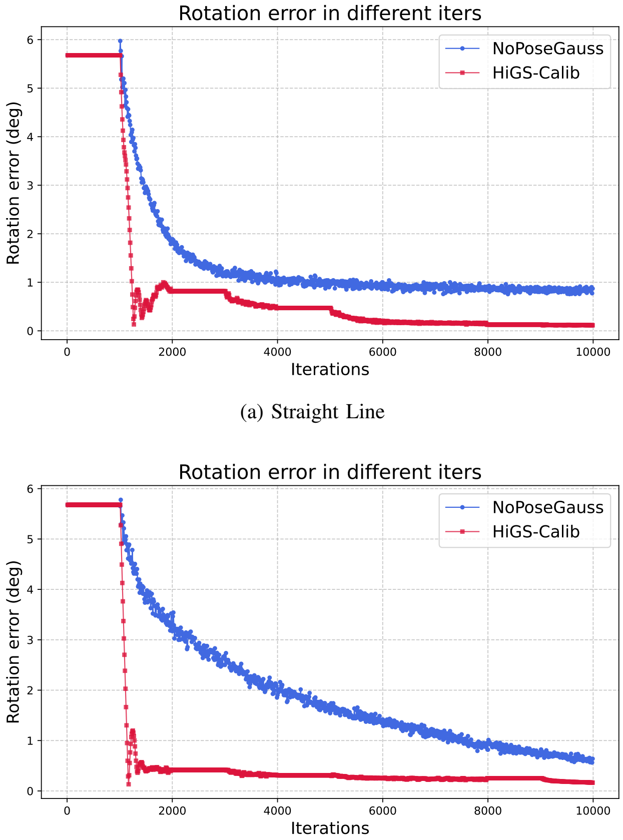
VI. Experimental Results

A. Experimental setup

Dataset. To ensure the rationality and fairness of the comparative experiments, we adopted the dataset settings consistent with [40]. Specifically, five typical data sequences were selected from the KITTI360 dataset [49]: Straight Line, Small Zigzag, Small Rotation, Large Zigzag, and Large Rotation. Furthermore, considering the high image quality in KITTI360, we complemented our evaluation with another representative dataset, KITTI [50]. KITTI features relatively blurred images, which enhances the generalizability and authenticity of our experimental findings. Within the KITTI dataset, also five typical sequences covering diverse scene types were selected, including: Rural Area, Narrow Road, Wide Road, Open Road, and Slope Road. In both datasets, without specific declaration, both the left camera and the right camera are used in the experiments, and the reported errors are the mean errors of the cameras involved.

Implementation details. For the implementations of the LCPG error introduced in Sec. V, the geometric error weight λ_{geo} is set to 0.1, the local window radius W is set to 2, and totally 30 frames are used for calibration. In terms of the pixel selection strategy, the depth thresholds D_{min} and D_{max} are set to $0.1m$ and $50m$, respectively, the flow confidence threshold C_{min} is set to 0.5, and the occlusion tolerance ratio r_o is set to 0.2.

On the aspect of our hierarchical optimization framework introduced in Sec. IV, the resolution for the voxel size in the initialization phase is set to $0.1m$. HiGS-Calib's three levels of optimization in the projection phase undergo 2,000, 2,000, and 4,000 iterations, respectively. For each level, the first 1,000 iterations constitute the model stage, while subsequent iterations form the calibration stage. Optimization in the fine-tuning phase totals 2,000 iterations. For the definition of the loss function, the SSIM weight λ_{ssim} is set to 0.2, the scale threshold s_{reg} is set to 10, and the weight λ_{depth} for depth loss and the weight λ_{reg} for the regularization loss are set to 10 and 0.01, respectively. The delayed iteration number N_{delay} for pose update in HiGS-Calib is set to 30. Besides, in most experiments (except for robust analysis), the most commonly



(a) Straight Line

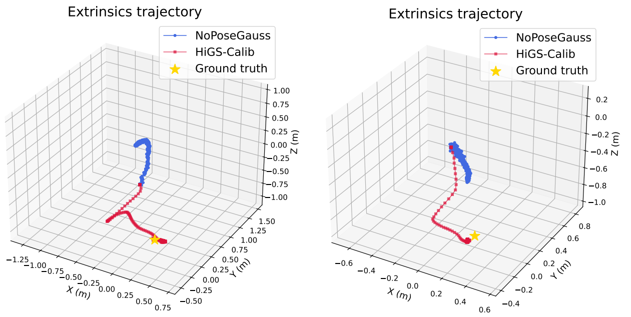
(b) Small Zigzag

Fig. 5. **Illustrations of the rotation errors along with the evolvement of calibration.** The curves plotted in red represent the results of HiGS-Calib, while blue curves represent for NoPoseGauss [41].

utilized initialization method, "from-LiDAR-initialization", is adopted to initialize the extrinsics, in which the cameras are initially oriented with approximate yaw angles of 0°, 90°, 180° and 270°, and the translations of the extrinsics are simply initialized to zero. As a relatively rough initialization method, especially on KITTI360 dataset, using "from-LiDAR-initialization" can also effectively evaluate the robustness of comparative calibration methods.

HiGS-Calib is implemented using Python with PyTorch [51]. All experiments are conducted on a common laptop equipped with an Intel(R) Core(TM) i7-13620H processor and an NVIDIA GeForce RTX 4060 GPU.

Comparative methods. To demonstrate HiGS-Calib's calibration performance, we compared it against three representative open-source approaches with necessary extensions: Calib-Anything [30], NoPoseGauss [41] and RobustCalib [43]. Calib-Anything [30] and RobustCalib [43] were evaluated using their official implementations. For NoPoseGauss [41], rather than optimizing all camera poses as in the official implementation, we modeled camera poses as a combination of LiDAR poses and LiDAR-camera extrinsics, while LiDAR poses are set fixed and only extrinsics are optimized.



(a) Straight Line (b) Small Zigzag

Fig. 6. **Illustrations of the trajectories of the calibrated cameras along with the evolvement of calibration.** The trajectories corresponding to different methods are plotted in different colors, and the ground-truth positions of the cameras are marked as yellow stars.

B. *Qualitative experiments*

Extrinsics calibration accuracy. To intuitively demonstrate the superior extrinsic calibration accuracy of our proposed HiGS-Calib, we performed calibration experiments according to the settings described in Sec. VI-A on multiple data sequences and compared its performance with other comparative methods. Specifically, we constructed projection maps by projecting LiDAR points onto images based on the calibration results, which can clearly reveal the calibration accuracy of the compared algorithms. Some typical examples of projection maps are shown in Fig. 4. As illustrated in Fig. 4, the projection maps generated by our HiGS-Calib exhibit the best registration between point clouds and images compared to other competitors, qualitatively demonstrating the advantageous calibration accuracy achieved by our HiGS-Calib.

Convergence stability comparison. To illustrate the fast and steady convergence of our hierarchical calibration pipeline, HiGS-Calib, we tracked the calibration process of HiGS-Calib, and recorded the rotation errors and the trajectories of calibrated cameras during the calibration process. Some typical samples of the rotation errors along with the training evolvement are plotted in Fig. 5, and corresponding trajectories of the cameras are plotted in Fig. 6. As a comparison, the corresponding results of NoPoseGauss [41], which is also an optimization-based compared method similar to HiGS-Calib, are also offered. From Fig. 6, it can be clearly seen that the rotation errors decrease obviously faster in the calibration process of HiGS-Calib compared to NoPoseGauss [41], and the final calibration accuracy of HiGS-Calib is also much better than NoPoseGS [41], which qualitatively verifies the outstanding calibration stability and robustness of HiGS-Calib. It's worth mentioning that, the observed plateaus in calibration error—during the 0-1,000, 2,000-3,000, and 4,000-5,000 iteration intervals—occur since HiGS-Calib is performing optimization in the model stages during these periods. During the model stages, only the 3DGS model is optimized, while the camera extrinsic parameters remain fixed.

C. *Quantitative experiments*

Extrinsics calibration accuracy. Under the experimental settings introduced in Sec. VI-A, we conducted the calibration

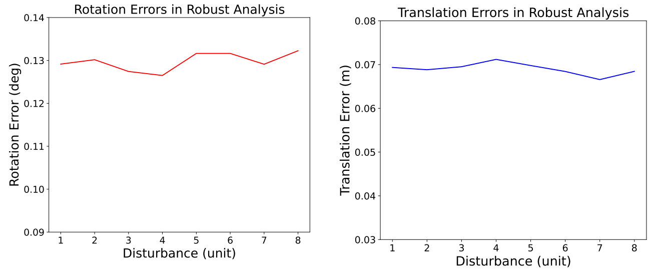
TABLE I
ROTATION ERRORS (deg) AND TRANSLATION ERRORS (M) OF COMPARED LIDAR-CAMERA CALIBRATION SCHEMES.

	Scenes	Calib-Anything [30]		NoPoseGauss [41]		RobustCalib [43]		3DGS-Calib* [40]		**HiGS-Calib**	
		Rot.↓	Trans.↓	Rot.↓	Trans.↓	Rot.↓	Trans.↓	Rot.↓	Trans.↓	Rot.↓	Trans.↓
KITTI360	Straight Line	0.453	0.314	1.011	1.409	5.230	0.873	–	–	**0.097**	**0.034**
	Small ZigZag	3.880	0.168	0.530	0.667	5.020	0.717	–	–	**0.189**	**0.085**
	Small Rotation	0.355	0.408	3.225	0.951	2.192	0.809	–	–	**0.063**	**0.113**
	Large ZigZag	1.977	0.594	0.473	1.177	3.719	0.751	–	–	**0.115**	**0.082**
	Large Rotation	0.474	0.198	0.232	1.246	4.349	0.847	–	–	**0.142**	**0.098**
	Mean	1.428	0.337	1.094	1.090	4.102	0.799	0.450	0.096	**0.121**	**0.083**
	L&R Std	0.046	0.187	0.815	0.117	1.710	0.038	–	–	**0.030**	**0.012**
KITTI	Rural Area	4.497	0.988	0.256	0.449	1.317	0.384	–	–	**0.160**	**0.053**
	Narrow Road	2.021	0.455	0.368	0.492	1.702	0.524	–	–	**0.214**	**0.054**
	Wide Road	2.846	0.786	0.385	0.465	1.019	0.282	–	–	**0.165**	**0.055**
	Open Road	1.333	0.845	0.501	0.469	1.411	0.449	–	–	**0.218**	**0.055**
	Slope Road	1.714	1.017	0.399	0.461	0.313	0.219	–	–	**0.218**	**0.053**
	Mean	2.482	0.818	0.382	0.467	1.152	0.372	–	–	**0.195**	**0.054**
	L&R Std	2.043	0.469	0.037	0.172	0.682	0.151	–	–	**0.014**	**0.004**

* 3DGS-Calib is not open-sourced and relevant results are from the original publication [40].

process of our HiGS-Calib and the other three competitors to evaluate their calibration accuracy. The calibration accuracy is measured by the rotation errors and translation errors among the calibrated extrinsics and the ground-truth ones. Relevant quantitative experimental results are summarized in Table I. It's worth mentioning that, except for the compared methods mentioned in Sec. VI-A, another 3DGS-based calibration method, 3DGS-Calib [40] is also taken as a comparative method. Since 3DGS-Calib is not open-sourced, we only gave the evaluation metrics of 3DGS-Calib on KITTI360 [49] following the results reported in [40]. "L&R" in Table I represents for the standard deviation of the calibration errors between the left view and right view. From Table I, it can be clearly seen that our HiGS-Calib consistently outperforms all other competitors on almost all data sequences in both the calibration accuracy and stability, which quantitatively corroborates the superior calibration accuracy and stability of HiGS-Calib.

Robustness to initial extrinsics. Actually, using the "from-LiDAR-initialization" on the KITTI360 dataset [49], the initial rotation error and translation error are about 6° and $1m$, respectively. Such a level of error already presents a significant robustness challenge for most existing algorithms. To evaluate the calibration robustness of HiGS-Calib to initial extrinsics more comprehensively, we also tested its calibration accuracy on the KITTI360 dataset [49] under varying degrees of initialization errors. Specifically, we applied different magnitudes of pose disturbance to the ground-truth extrinsics to generate initial values and then conducted the calibration. Each disturbance unit is equivalent to rotating the camera by 1° and also applying a translation of 5cm on each of the three axes. The left-view camera was adopted to conduct the evaluation. The evaluation results under different disturbance settings are presented in Fig. 7. As shown in Fig. 7, even when the disturbance reaches 8 units (resulting in a rotation error of approximately 14°), HiGS-Calib maintains stable correction performance, and different initialization points bring negligible influence on the final calibration accuracy (within 0.01° and 1cm), which convincingly demonstrates the robustness of HiGS-Calib to initial extrinsics.



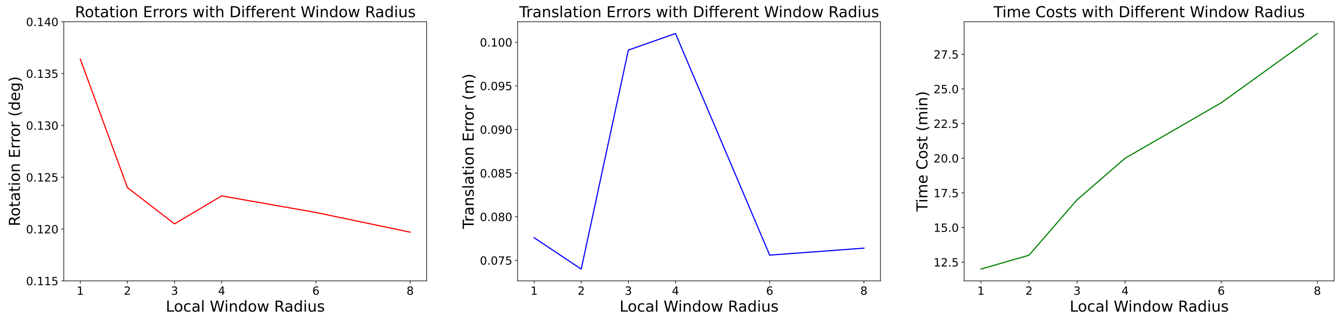
(a) Rotation Error (b) Translation Error

Fig. 7. **The rotation and translation calibration errors of HiGS-Calib under different pose disturbance settings in the robust analysis.** The rotation errors are plotted as the red curve in (a), while the blue curve in (b) represents the translation calibration errors.

Time cost analysis. For the purpose of time complexity analysis, the average calibration time consumed by each of the compared algorithms to complete calibration was recorded. Specifically, CalibAnything [30], NoPoseGauss [41], RobustCalib [43] and our HiGS-Calib takes about 19min, 11min, 20min and 13min to finish the calibration, respectively. As an offline calibration algorithm, our HiGS-Calib shows comparatively favorable calibration speed relative to existing SOTA methods, and its calibration time of just 13 minutes on common laptop computational environments can meet requirements for most offline calibration scenarios. Given that HiGS-Calib's accuracy significantly surpasses other competitors, we believe that our HiGS-Calib delivers the best overall performance.

D. Ablation studies

Ablation studies on LCPG error. Compared to relying entirely on photometric-based rendering loss, introducing LCPG error significantly enhances the robustness and stability of the calibration pipeline. To more convincingly validate our perspective, ablation experiments focusing on LCPG error were conducted. Specifically, we evaluated the calibration performance of HiGS-Calib and its three variants on the KITTI360



(a) Rotation Error

(b) Translation Error

(c) Time Cost

Fig. 8. **Illustrations of the calibration performance of HiGS-Calib under different window radius W settings.** The results of rotation errors, translation errors, and calibration time costs are plotted in (a), (b) and (c), respectively.

TABLE II
ABLATION STUDIES ON LCPG ERROR.

Method	Rotation error (deg)↓	Translation error (m)↓
w/o LCPG	2.583	0.993
w/o L-W	0.499	0.189
w/o P-S	0.149	0.122
HiGS-Calib	**0.124**	**0.074**

TABLE III
ABLATION STUDIES ON THE HIERARCHICAL PIPELINE IN HIGS-CALIB.

Method	Rotation error (deg)↓	Translation error (m)↓
w/o L_{depth}	1.130	0.946
w/o L_{reg}	0.129	0.077
w/o H-O	0.141	0.102
w/o D-U	0.137	**0.074**
HiGS-Calib	**0.124**	**0.074**

dataset [49], including: (1) w/o LCPG: without using LCPG error, only relying on rendering photometric loss; (2) w/o L-W: without using local windows, constructing the LCPG error solely based on the current frame and its adjacent single frame; and (3) w/o P-S: without performing pixel selection. The left camera was adopted and experimental results are presented in Table II. As shown in Table II, the calibration performance of HiGS-Calib significantly outperforms the other three variants, effectively demonstrating the importance of LCPG error in improving the accuracy of HiGS-Calib, as well as the necessity of each component in the design of LCPG.

Ablation studies on the hierarchical pipeline. Beyond the introduction of the LCPG error, the hierarchical pipeline design of HiGS-Calib is also essential to ensure calibration stability. Hence, we further conducted the ablation analysis on HiGS-Calib's calibration pipeline. Specifically, ablation experiments were performed on the KITTI360 dataset [49] to evaluate the calibration accuracy of HiGS-Calib against four variant methods, including: (1) w/o L_{depth}: without depth loss supervision; (2) w/o L_{reg}: without regularization loss supervision; (3) w/o H-O: without hierarchical optimization structure, optimizing only at the original resolution (i.e., the highest level of optimization); and (4) w/o D-U: without the delayed pose update strategy. Experimental results are shown in Table III. As demonstrated in Table III, HiGS-Calib consistently outperforms all other variants, providing compelling evidence for the rationality of its pipeline design.

Ablation studies on the match window size. LCPG is a photometric projective error constructed within a local window, making the window radius W a critical parameter for the LCPG loss. We evaluated the calibration performance of HiGS-Calib under different window radius settings on the KITTI360 dataset (based on the left-view camera). Evaluation metrics included rotation error, translation error, and calibration time consumption. The experimental results are shown in Fig. 8. From Fig. 8, it can be seen that when the window radius is set to 1, HiGS-Calib exhibits significantly lower rotational accuracy compared to larger window radius configurations. Meanwhile, a larger window radius leads to longer calibration time for the algorithm. Actually, when the window radius exceeds 2, the marginal improvement in accuracy becomes limited while the computational cost increases noticeably. Therefore, to balance calibration accuracy and efficiency, we set the window radius to 2. In practical applications, users may adjust the window radius based on their specific needs to achieve optimal overall performance.

VII. LIMITATIONS AND FUTURE WORK

Our HiGS-Calib incorporates the LCPG loss and hierarchical optimization strategy, demonstrating superior robustness compared to other existing comparative methods. Experimental validation also confirms that HiGS-Calib maintains satisfactory robustness to initial extrinsic parameters. Nevertheless, as a 3DGS-based calibration pipeline, excessively poor initialization may still cause optimization to converge to local minima, leading to calibration failure. This represents an inherent limitation common to existing co-visibility matching-based optimization algorithms. A viable solution involves integrating HiGS-Calib with hand-eye calibration algorithms. By leveraging initialization-robust methods to provide high-quality initial estimates for HiGS-Calib, both robust and accurate extrinsic calibration may be achieved. Besides, HiGS-Calib currently lacks support for calibrating asynchronous LiDAR-camera sensor suites. This limitation could be addressed by implementing a continuous-time representation framework based on B-spline curves. Except for the aforementioned limitations, our method currently does not support handling dynamic scenes, which is consistent with most existing calibration methods based on 3DGS. Addressing dynamic scenes remains a challenging problem for 3DGS-based calibration algorithms and is an important direction for our future research. Future work will focus on these areas to further enhance HiGS-Calib's capabilities. Additionally, we will pursue broader framework refinement to expand applicability across diverse sensor configurations.

VIII. Conclusion

In this paper, we studied a practical problem, 3DGS-based camera-LiDAR calibration, and proposed a novel solution, namely HiGS-Calib. Unlike existing 3DGS-based calibration methods that are highly dependent on the photometric rendering loss, we proposed a more robust error model, called the LCPG error, and integrated it into HiGS-Calib. As a photometric-geometric error based on inter-frame projections, LCPG can effectively quantify the extrinsic error by leveraging only geometric information rather than colors from the 3DGS model, and mainly focuses on the local structure inaccuracy of the calibration frames to mitigate the interference from global accumulated localization errors. Building upon the proposed LCPG error, we further developed a hierarchical multi-level pipeline for LiDAR-camera calibration, which can achieve robust extrinsic calibration through cascaded alternating optimization of the scene representation and extrinsic parameters. The experimental results effectively corroborate the superior performance of I-DACS. Although our HiGS-Calib demonstrates significant performance advantages over current state-of-the-art methods, it still exhibits moderate dependence on initial extrinsics and currently lacks support for asynchronous sensor calibration. We will prioritize these aspects in future efforts toward methodological refinement.

References

[1] T. Liang, H. Xie, K. Yu, Z. Xia, Z. Lin, *et al.* "BEVFusion: A simple and robust lidar-camera fusion framework," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 10421-10434.

[2] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, *et al.*, "BEVFormer: Learning bird's-eye-view representation from LiDAR-camera via spatiotemporal transformers," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 47, no. 3, pp. 2020-2036, 2025.

[3] J. S. Berrio, M. Shan, S. Worrall, and E. Nebot, "Camera-LiDAR integration: Probabilistic sensor fusion for semantic mapping," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7637-7652, 2022.

[4] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "CamVox: A low-cost and accurate LiDAR-assisted visual SLAM system," in *Proc. IEEE Int'l Conf. Robot. Automat.*, 2021, pp. 5049-5055.

[5] A. Chen, X. Wang, K. Shi, S. Zhu, B. Fang, *et al.*, "ImmFusion: Robust mmWave-RGB fusion for 3D human body reconstruction in all weather conditions," in *Proc. IEEE Int'l Conf. Robot. Automat.*, 2023, pp. 2752-2758.

[6] Z. Gu, J. Ma, Y. Huang, H. Wei, Z. Chen, *et al.*, "HGSFusion: Radar-camera fusion with hybrid generation and synchronization for 3d object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2025, pp. 3185-3193.

[7] J. Wang, F. Li, Y. An, X. Zhang, and H. Sun, "Toward robust LiDAR-camera fusion in BEV space via mutual deformable attention and temporal aggregation," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 34, no. 7, pp. 5753-5764, 2024.

[8] Y. Yang, J. Liu, T. Huang, Q. L. Han, G. Ma, and B. Zhu, "RaLiBEV: Radar and LiDAR BEV fusion learning for anchor box free object detection systems," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 35, no. 5, pp. 4130-4143, 2025.

[9] A. Zhu, Y. Xiao, C. Liu, and Z. Cao, "Robust LiDAR-camera alignment with modality adapted local-to-global representation," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 33, no. 1, pp. 59-73, 2023.

[10] J. Zhang, X. Li, H. Zhong, Q. Zhang, Y. Cao, *et al.*, "HumanRef-GS: Image-to-3D human generation with reference-guided diffusion and 3D Gaussian splatting," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 35, no. 7, pp. 6867-6880, 2025.

[11] Z. Guo, W. Zhou, L. Li, M. Wang, and H. Li, "Motion-aware 3D Gaussian splatting for efficient dynamic scene reconstruction," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 35, no. 4, pp. 3119-3133, 2025.

[12] Z. Liu, J. Su, G. Cai, Y. Chen, B. Zeng, *et al.*, "GeoRGS: Geometric regularization for real-time novel view synthesis from sparse inputs," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 34, no. 12, pp. 13113-13126, 2024.

[13] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *Proc. IEEE Int'l Conf. Intell. Robots and Syst.*, 2004, pp. 2301-2306.

[14] R. Unnikrishnan and M. Hebert, "Fast extrinsic calibration of a laser rangefinder to a camera," *Carnegie Mellon University*, 2005.

[15] A. Dhall, K. Chelani, V. Radhakrishnan, and K.M. Krishna, "LiDAR-camera calibration using 3D-3D point correspondences," *arXiv preprint, arXiv:1705.09785*, 2017.

[16] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3D LiDAR using line and plane correspondences," in *Proc. IEEE/RSJ Int'l Conf. Intell. Robots and Syst.*, 2018, pp. 5562-5569.

[17] C. Guindel, J. Beltrán, D. Martín, and F. García, "Automatic extrinsic calibration for lidar-stereo vehicle sensor setups," in *Proc. IEEE Int'l Conf. Intell. Transp. Syst.*, 2017, pp. 1-6.

[18] Z. Pusztai and L. Hajder, "Accurate calibration of LiDAR-camera systems using ordinary boxes," in *Proc. IEEE Int'l Conf. Computer Vis. Workshops*, 2017, pp. 394-402.

[19] J. Cui, J. Niu, Z. Ouyang, Y.He, and D. Liu, "ACSC: Automatic calibration for non-repetitive scanning solid-state LiDAR and camera systems," *arXiv preprint, arXiv:2011.08516*, 2020.

[20] Y. Dong, Y. Liu, L. Li, H. Deng, J. Tang, and J. Li, "Automatic LiDAR–camera extrinsic calibration using pseudoimage and multiple targets," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1-13, 2024.

[21] G. Yan, Z. Liu, C. Wang, C. Shi, and P. Wei, *et al.* "Opencalib: A multi-sensor calibration toolbox for autonomous driving," *arXiv preprint, arXiv:2205.14087*, 2022.

[22] P. Moghadam, M. Bosse, and R. Zlot, "Line-based extrinsic calibration of range and image sensors," in *IEEE Int'l Conf. Robot. Automat.*, 2013, pp. 3685–3691.

[23] X. Liu, C. Yuan, and F. Zhang, "Targetless extrinsic calibration of multiple small FoV LiDARs and cameras using adaptive voxelization," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1-12, 2022.

[24] K. Koide, S. Oishi, M. Yokozuka, and A. Banno, "General, single-shot, target-less, and automatic lidar-camera extrinsic calibration toolbox," in *Proc. IEEE Int'l Conf. Robot. Automat.*, 2023, pp. 11301–11307.

[25] P. E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning feature matching with graph neural networks," in *Proc. IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, 2020, pp. 4937-4946.

[26] G. Pandey, J. McBride, S. Savarese, and R. Eustice, "Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information," in *Proc. AAAI Conf. Artif. Intell.,* 2012, pp. 2053-2059.

[27] R. Ishikawa, S. Zhou, Y. Sato, T. Oishi, and K. Ikeuchi, "Lidar-camera calibration using intensity variance cost," in *Proc. IEEE Int'l Conf. Robot. Automat.*, pp. 10688–10694, 2024.

[28] H. Ou, H. Cai, and J. Wang, "Targetless lidar-camera calibration via cross-modality structure consistency," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 1, pp. 2636-2648, 2024.

[29] F. Liu, Y. Cao, X. Cheng, and X. Wu, "Transformer-based local-to-global lidar-camera targetless calibration with multiple constraints," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1-13, 2024.

[30] Z. Luo, G. Yan, X. Cai, and B. Shi, "Zero-training LiDAR-camera extrinsic calibration method using segment anything model," in *Proc. IEEE Int'l Conf. Robot. Automat.*, 2024, pp. 14472-14478.

[31] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, *et al.*, "Segment anything," in *Proc. IEEE/CVF Int'l Conf. Computer Vis.*, 2023, pp. 3992-4003.

[32] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, and R. N. R. Ramamoorthi, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 405–421.

[33] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139:1-14, 2023.

[34] S. Zhou, S. Xie, R. Ishikawa, K. Sakurada, M. Onishi, and T. Oishi, "INF: Implicit neural fusion for LiDAR and camera," in *Proc. IEEE/RSJ Int'l Conf. Intell. Robots and Syst.*, 2023, pp. 10918-10925.

[35] Q. Herau, N. Piasco, M. Bennehar, L. Roldao, D. Tsishkou, *et al.*, "SOAC: Spatio-temporal overlap-aware multi-sensor calibration using neural radiance fields," in *IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, 2024, pp. 15131-15140.

[36] Z. Yang, G. Chen, H. Zhang, K. Ta, and I. A. Bârsan, *et al.*, "Unical: Unified neural sensor calibration," in *Proc. Eur. Conf. Computer Vis.*, 2024, pp.327-345.

[37] G. Hess, C. Lindström, M. Fatemi, C. Petersson, and L. Svensson, "SplatAD: Real-time lidar and camera rendering with 3d Gaussian splatting for autonomous driving," in *Proc. Computer Vis. Pattern Recognit.*, 2025, pp. 11982-11992.

[38] Z. Qian, S. Wang, M. Mihajlovic, A. Geiger, and S. Tang, "3DGS-Avatar: Animatable avatars via deformable 3d Gaussian splatting," in *Proc. IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, 2024, pp. 5020-5030.

[39] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting SLAM," in *Proc. IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, 2024, pp. 18039-18048.

[40] Q. Herau, M. Bennehar, A. Moreau, N. Piasco, L. Roldão, *et al.*, "3DGS-Calib: 3D Gaussian splatting for multimodal spatioTemporal calibration," in *Proc. IEEE/RSJ Int'l Conf. Intell. Robots and Syst.*, 2024, pp. 8315-8321.

[41] C. Schmidt, J. Piekenbrinck, and B. Leibe, "Look Gauss, No Pose: Novel view synthesis using Gaussian splatting without accurate pose initialization," in *Proc. IEEE/RSJ Int'l Conf. Intell. Robots and Syst.*, 2024, pp. 8732-8739.

[42] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2D Gaussian splatting for geometrically accurate radiance fields," in *Proc. ACM SIGGRAPH Conf.*, 2024, pp. 1–11.

[43] S. Zhou, S. Xie, R. Ishikawa, and T. Oishi, "Robust LiDAR-camera calibration with 2D Gaussian splatting," *IEEE Robot. Automat. Let.*, vol. 10, no. 5, pp. 4674-4681, 2025.

[44] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int'l Conf. Intell. Robots and Syst.*, 2018, pp. 4758-4765.

[45] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053-2073, 2022.

[46] J. Edstedt, Q. Sun, G. Bökman, M. Wadenbäck, and M. Felsberg, "RoMa: Robust dense feature matching," in *Proc. IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, 2024, pp. 19790-19800.

[47] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.,* vol. 13, no. 4, pp. 600-612, 2004.

[48] R. Mukundan, "Quaternions," in *Adv. Methods in Computer Graph.: With examples in OpenGL*, 2012, pp. 77-112.

[49] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," in *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 45, no. 3, pp. 3292-3310, 2023.

[50] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Computer Vis. Pattern Recognit.*, 2012, pp. 3354-3361.

[51] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint, arXiv:1912.01703*, 2019.

Tianjun Zhang received his B.Sc. and Ph.D. degree from the School of Software Engineering, Tongji University, Shanghai, China, in 2019 and 2024, respectively. Starting from Dec. 2024, he worked as a postdoctoral at the Department of Automation, Shanghai Jiao Tong University, China. His research interests include collaborative SLAM, computer vision, and sensor calibration.

Lin Zhang (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2003 and 2006, respectively. He received the Ph.D. degree from the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, in 2011. From March 2011 to August 2011, he was a Research Associate with the Department of Computing, The Hong Kong Polytechnic University. In Aug. 2011, he joined the School of Software Engineering, Tongji University, Shanghai, China. He is currently a Full Professor at the School of Computer Science and Technology, Tongji University. His current research interests include environment perception of intelligent vehicle, pattern recognition, computer vision, and perceptual image/video quality assessment. He serves as an Associate Editor for IEEE Robotics and Automation Letters, and Journal of Visual Communication and Image Representation. He was awarded as a Young Scholar of Changjiang Scholars Program, Ministry of Education, China.

Hesheng Wang (Senior Member, IEEE) received the B.Eng. degree in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2002, and the M.Phil. and Ph.D. degrees in automation and computer-aided engineering from The Chinese University of Hong Kong, Hong Kong, in 2004 and 2007, respectively. He is currently a Distinguished Professor with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China. His current research interests include visual servoing, intelligent robotics, computer vision, and autonomous driving. Dr. Wang is an Associate Editor of Robotic Intelligence and Automation and the International Journal of Humanoid Robotics, a Senior Editor of the IEEE/ASME Transactions on Mechatronics, an Editor-in-chief of Robot Learning. He served as an Associate Editor of the IEEE Transactions on Robotics from 2015 to 2019, an IEEE Transactions on Automation Science and Engineering from 2021 to 2023, a Technical Editor of the IEEE/ASME Transactions on Mechatronics from 2020 to 2023, an Editor of Conference Editorial Board (CEB) of IEEE Robotics and Automation Society from 2022 to 2024. He was the General Chair of IEEE ROBIO 2022 and IEEE RCAR 2016, and the Program Chair of IEEE ROBIO 2014 and IEEE/ASME AIM 2019. He will be the General Chair of IEEE/RSJ IROS 2025.