

# EasyCalib: A Novel Target for High-Accuracy Fully Automatic Extrinsic Calibration of Camera and LiDAR

Zhiyu Zhou<sup>1b</sup>, *Student Member, IEEE*, Zhi Gao<sup>1b</sup>, *Member, IEEE*, Min Cao<sup>1b</sup>, Jingshi Wang<sup>1b</sup>,  
Yong Li<sup>1b</sup>, *Member, IEEE*, Zhipeng Lin<sup>1b</sup>, Wenbin Yang<sup>1b</sup>, Xiaonan Wang<sup>1b</sup>, and Qiyuan Zhang<sup>1b</sup>

**Abstract**—Precise extrinsic calibration is a fundamental requirement for enabling robust camera–light detection and ranging (LiDAR) fusion applications. Existing target-based methods suffer from practical limitations: they require manual annotation, precise target dimensions, and initial extrinsic estimates, demand complex target placement, and have limited applicability across different sensor types. To the best of our knowledge, we are not aware of a prior approach that simultaneously achieves full automation, avoids reliance on precise target dimensions and accurate initialization, and generalizes across both mechanical and nonrepetitive-scan solid-state LiDARs. In this article, we propose EasyCalib, a novel artificial target for high-accuracy, fully automatic extrinsic calibration for almost all types of camera–LiDAR systems. Specifically, we design dedicated point cloud segmenters for mechanical and solid-state LiDARs to automatically extract EasyCalib. By leveraging the co-visible corner points and central edge features of EasyCalib in both the image and point cloud domains, we achieve a coarse-to-fine, initialization-free, and highly accurate extrinsic calibration. Moreover, since the proposed calibration pipeline does not require the target to be strictly orthogonal and tolerates moderate assembly misalignment, and since the target dimensions are not involved in the calibration computation, EasyCalib requires

neither complex manufacturing procedures nor precise prior knowledge of its size.

**Index Terms**—Calibration target, extrinsic calibration, light detection and ranging (LiDAR) and camera.

## I. INTRODUCTION

AMONG various sensors, light detection and ranging (LiDAR) and camera, both of which have their own strengths and weaknesses, show surprising complementarity. In recent years, LiDAR–camera fusion has emerged as a promising research direction, significantly enhancing robot localization, perception, and mapping in challenging environments [17], [18], [19], [20], [21]. The extrinsic calibration of these sensors, a key component in data fusion, has also been extensively studied, achieving high accuracy and resulting in mature software packages [4], [5], [8].

Despite this progress, existing methods still face significant challenges in practical applications. Targetless methods [2], [3], [4], [5], [22] rely on weak scene features such as depth-discontinuity edges, appearance statistics, or independent motions, which makes them highly dependent on accurate initial extrinsic parameters. These methods often fail under conditions with poor lighting, low texture, or odometry drift, leading to unstable results. Target-based methods reduce the reliance on initialization but introduce other limitations. Planar or perforated boards [8], [9], [10], [11] require careful placement in multiple poses, dense point clouds to detect features, and manual selection of point cloud regions. Sphere-based approaches [12], [13], [14] demand high-precision manufacturing and multiple observations, and any error in the target dimensions can result in systematic calibration bias. Furthermore, most existing targets are designed for mechanical spinning LiDARs and do not adapt well to semi-solid-state or hybrid LiDARs with nonuniform scan patterns. Many current pipelines still require manual annotation, rely on known target dimensions, assume good initial extrinsic parameters, need multiple viewpoints to gather sufficient constraints, or fail to achieve stable subpixel accuracy under diverse lighting conditions and sensor types.

To address these limitations, we introduce EasyCalib, a self-designed 3-D trihedral target composed of three sector-shaped acrylic plates with distinct red, green, and blue colors (Fig. 1). EasyCalib enables a fully automatic coarse-to-fine calibration pipeline. First, we propose a scan-line-based clustering method

Received 9 March 2026; revised 3 April 2026; accepted 14 April 2026. Date of publication 28 April 2026; date of current version 8 May 2026. This work was supported in part by the National Key Laboratory Project of Information System Engineering under Grant WZC20265290404, in part by the Key Research Project of China State Railway Group under Grant N2025T007-C (JB), in part by the Guangxi Major Science and Technology Project under Grant AA24263005, and in part by the Fundamental Research Funds for the Central Universities-Key Interdisciplinary Team Development Project under Grant 2042026kf0018. The Associate Editor coordinating the review process was Dr. Irida Shallari. (*Corresponding author: Zhi Gao.*)

Zhiyu Zhou and Zhi Gao are with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China (e-mail: zhouzhiyu@whu.edu.cn; gaozhinus@gmail.com).

Min Cao is with Wuhan Guanggu Zoyon Science and Technology Company Ltd., Wuhan 430223, China (e-mail: market@zoyon.com.cn).

Jingshi Wang is with the School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai 200240, China, and also with Jiangsu Automation Research Institute, Lianyungang, Jiangsu 222061, China (e-mail: wang\_jingshi@foxmail.com).

Yong Li is with Shanghai Aerospace Control Technology Institute, Shanghai 201109, China, and also with Shanghai Integrated Innovation Center for Manned Lunar Exploration, Shanghai 201109, China (e-mail: marsleely@alumni.sjtu.edu.cn).

Zhipeng Lin is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong, SAR, China (e-mail: zplin@link.cuhk.edu.hk).

Wenbin Yang is with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China (e-mail: wenbin\_yang@whu.edu.cn).

Xiaonan Wang and Qiyuan Zhang are with ZG Technology Company Ltd., Wuhan 430000, China (e-mail: xiaonan.x.wang@hexagon.com; qiyuanzhang@whu.edu.cn).

Digital Object Identifier 10.1109/TIM.2026.3687975

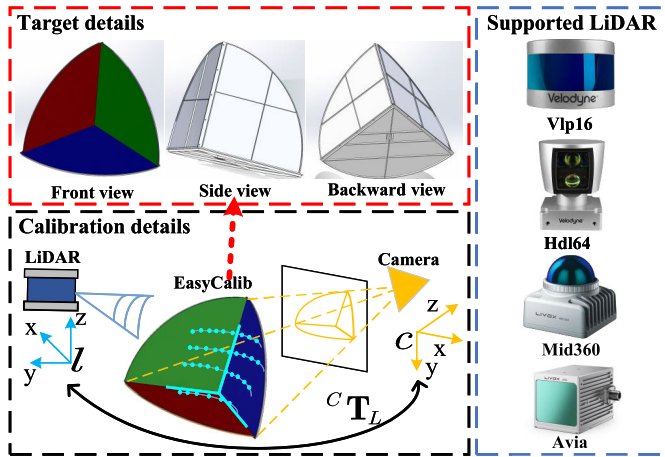


Fig. 1. EasyCalib-based extrinsic calibration for LiDAR and the camera. Our core idea is to extract line and centroid features of EasyCalib from two sensors, thus obtaining accurate extrinsic parameters by aligning these features. The red box highlights the structure details of EasyCalib.

to automatically localize the target in the LiDAR point cloud. Second, we extract compact and reliable features from both modalities: in 3-D, we perform plane segmentation to recover the target planes and their intersection cues; in 2-D, we use gradient-direction statistics on the trinarized edge map to localize the center-corner feature. Third, we obtain a stable initialization by solving a perspective- $n$ -point (PnP) problem with the extracted 3-D–2-D point correspondences. Finally, we refine the extrinsics via a point-to-line optimization that enforces pixel-level alignment between projected 3-D lines and observed 2-D line features. Importantly, this refinement stage does not rely on strict target orthogonality or additional geometric priors; it only minimizes reprojection residuals between matched features, thus remaining tolerant to deviations from ideal orthogonal assembly. Extensive experiments demonstrate the robustness and accuracy of our method and its adaptability across different LiDAR–camera systems. Overall, our contributions are as follows.

- 1) We design a novel calibration target, EasyCalib, consisting of three sector-shaped, color-coded planar plates. The distinctive trihedral geometry and RGB layout provide robust cues for reliable detection and feature extraction in both LiDAR and camera data, without requiring strictly orthogonal assembly.
- 2) We develop a fully automatic coarse-to-fine calibration pipeline. By combining scan-line clustering, robust feature extraction, PnP-based initialization, and a pixel-level point-to-line refinement, the system performs accurate extrinsic calibration without manual intervention or an initial extrinsic estimate.
- 3) Extensive experiments show that EasyCalib achieves higher accuracy and stronger robustness than representative target-based and targetless methods and generalizes well to diverse LiDAR–camera configurations.

## II. RELATED WORK

In this section, we briefly review target-based and targetless calibration methods and highlight their respective advantages and limitations.

### A. Targetless LiDAR–Camera Extrinsic Calibration

Targetless methods eliminate the need for preset artificial markers by directly aligning naturally occurring features visible to both LiDAR and camera sensors to optimize extrinsic parameters. These methods are typically categorized into feature- and motion-based approaches. Yuan et al. [4] and Liu et al. [5] developed a widely used targetless calibration toolkit that detects depth discontinuity edges in point clouds and matches them with visual texture edges in images to form point-line correspondences for extrinsic optimization. However, these approaches are highly sensitive to initial parameter accuracy and susceptible to noise and occlusion during feature extraction and matching, often leading to calibration failure. Pandey et al. [2] and Koide et al. [3] proposed a mutual information-based method leveraging reflectivity intensity similarity. Although these approaches can achieve pixel-level precision, they remain highly sensitive to the accuracy of initial parameters. An alternative motion-based strategy [22] estimates trajectories of LiDAR and camera separately and aligns them to compute the extrinsic parameters. Although this method does not require an initial value or feature matching, its calibration accuracy is often limited due to unpredictable odometry tracking failures. To achieve higher-accuracy extrinsic results and ensure superior sensor fusion performance, target-based methods are generally more stable and reliable compared to targetless approaches.

### B. Target-Based LiDAR–Camera Extrinsic Calibration

Compared to the unreliability of targetless methods, target-based approaches provide a more robust, accurate, and environment-independent solution for extrinsic calibration. Nevertheless, current target-based approaches face several drawbacks (limited automation, manufacturing errors, and restricted applicability across sensor types). Table I summarizes the strengths and weaknesses of the main target types; we discuss them below.

1) *Nontrihedron-Based Methods*: The planar board is the most commonly used target: LiDAR reliably detects its planar geometry, and cameras can easily use attached markers (e.g., checkerboards or QR codes) for robust detection. Zhou et al. [8] and Park et al. [9] performed semi-automatic calibration by matching the board's planar and line features across LiDAR and camera. However, these methods are limited to mechanical LiDARs and still require time-consuming manual annotation of the LiDAR point cloud. Moreover, these methods require a reliable initial LiDAR–camera extrinsic (typically from manual calibration) for accurate feature alignment. Velo2Cam [10] and FastCalib [11] add perforations to the checkerboard to obtain 3-D hole-center features. However, these methods rely on dense point clouds and fail with low-channel LiDAR, which cannot reliably detect the hole feature of the point cloud. Both planar and perforated boards provide

TABLE I  
COMPARISON OF REPRESENTATIVE LiDAR–CAMERA EXTRINSIC CALIBRATION METHODS

| Method                       | Target/Marker                    | Human Effort      | Prior Knowledge   |                        | LiDAR Support |   |   |   |
|------------------------------|----------------------------------|-------------------|-------------------|------------------------|---------------|---|---|---|
|                              |                                  | Manual annotation | Initial extrinsic | Known target dimension | A             | B | C | D |
| Scaramuzza et al. (2007) [1] | Target-less (natural scenes)     | ✓                 |                   | –                      | ✓             |   |   |   |
| Pandey et al. (2021) [2]     | Target-less (mutual information) |                   | ✓                 | –                      |               |   |   | ✓ |
| Koide et al. (2023) [3]      | Target-less (mutual information) |                   | ✓                 | –                      |               |   |   | ✓ |
| Yuan et al. (2021) [4]       | Target-less (edge feature)       |                   | ✓                 | –                      |               |   |   | ✓ |
| Liu et al. (2022) [5]        | Target-less (edge feature)       |                   | ✓                 | –                      |               |   |   | ✓ |
| SGCalib (2024) [6]           | Target-less (semantic)           |                   |                   | –                      |               |   |   | ✓ |
| Dhall et al. (2017) [7]      | Planar board                     | ✓                 |                   | ✓                      | ✓             |   |   |   |
| Zhou et al. (2018) [8]       | Planar board                     | ✓                 | ✓                 | ✓                      | ✓             | ✓ |   |   |
| Park et al. (2014) [9]       | Planar board                     | ✓                 | ✓                 | ✓                      | ✓             | ✓ |   |   |
| Beltrán et al. (2022) [10]   | Perforated board                 | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| FAST-Calib (2025) [11]       | Perforated board                 | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| Tóth et al. (2020) [12]      | Sphere                           | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| Kümmerle et al. (2018) [13]  | Sphere                           | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| Pereira et al. (2016) [14]   | Sphere                           | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| Gong et al. (2013) [15]      | Trihedron                        | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| Choi et al. (2024) [16]      | Trihedron                        | ✓                 |                   | ✓                      |               |   | ✓ | ✓ |
| <b>EasyCalib (Ours)</b>      | Trihedron                        |                   |                   |                        | ✓             | ✓ | ✓ | ✓ |

<sup>A</sup> Low-channel mechanical LiDAR (e.g., Velodyne VLP-16).

<sup>B</sup> High-channel mechanical LiDAR (e.g., Velodyne HDL-64).

<sup>C</sup> Non-repetitive-scan solid-state LiDAR (e.g., Livox Avia and Mid360).

<sup>D</sup> Dense point cloud required (i.e., performance depends on sufficiently dense returns on the target/structures).

– indicates not applicable (no physical target is used, hence no target dimension is required).

geometric constraints that lie on the same plane, so multiple observations from different viewpoints are typically required to obtain sufficient constraints. Spherical targets enable easy detection of sphere centers in both image and point cloud for 2-D–3-D point correspondences, but require at least four LiDAR–camera data pairs [12], [13], [14]. Moreover, these methods rely on accurate target dimensions for extrinsic estimation and optimization, which raises target manufacturing complexity and may introduce systematic calibration errors. In this work, we design a novel target, EasyCalib, which overcomes the limitations of existing targets, enabling fully automatic, robust, and high-accuracy extrinsic calibration for almost all types of LiDAR–camera systems.

2) *Trihedron-Based Methods*: Orthogonal trihedral structures have been widely adopted for camera–LiDAR calibration due to their rich geometric cues. Gong et al. [15] exploited natural trihedral corners in real-world environments (e.g., building corners) and recovered the corner pose independently in the camera and LiDAR frames using motion constraints, from which the camera–LiDAR relative pose is obtained. However, Gong’s method is sensitive to LiDAR point-cloud sparsity as well as illumination and texture conditions on the observed surfaces. Moreover, since it relies primarily on external pose estimates without enforcing pixel-level feature alignment, it is difficult to achieve high-precision calibration. Choi et al. [16] proposed a distinctive trihedral target: an orthogonal three-sided box with a checkerboard pattern attached to each face. Similar in spirit to Gong et al. [15], their method performs extrinsic calibration indirectly by estimating the poses of the LiDAR and the camera with respect to the target and then deriving the relative transformation. Thanks to the known checkerboard dimensions and the mutually orthogonal planes, Choi’s approach can obtain accurate relative poses without

relying on motion constraints. However, it has several clear limitations. First, like Gong’s method, it does not enforce pixel-level feature alignment, which restricts the achievable precision. Second, it requires users to manually specify the coordinate-axis orientations of the LiDAR and camera frames, which is inconvenient and can introduce user-dependent errors in practice.

### III. OUR METHOD

We here discuss the details of the proposed calibration method, and the pipeline is shown in Fig. 2. First, we designed customized point cloud clustering algorithms for different types of LiDARs and identified the segments of EasyCalib using geometric regularities. The 2-D and 3-D corner features of EasyCalib are then leveraged to achieve fully automatic and robust initialization of extrinsic parameters. Fine-tuned point-line optimization is performed using the centerline features of EasyCalib to achieve subpixel-level accuracy.

#### A. Manufacturing Complexity of EasyCalib

EasyCalib imposes very low fabrication requirements. For common calibration targets such as planar boards [8], [9] and spheres [12], [13], [14], achieving high calibration accuracy requires precisely known board length and width and the sphere radius, as well as strict flatness of the plane and roundness of the sphere. This is because their geometric and dimensional parameters directly participate in the calibration computation, so any manufacturing error introduces systematic bias and degrades the final accuracy. In contrast, EasyCalib does not require precise geometry or dimension information as prior knowledge for extrinsic calibration. Actually, EasyCalib does not require the three planar boards to be strictly

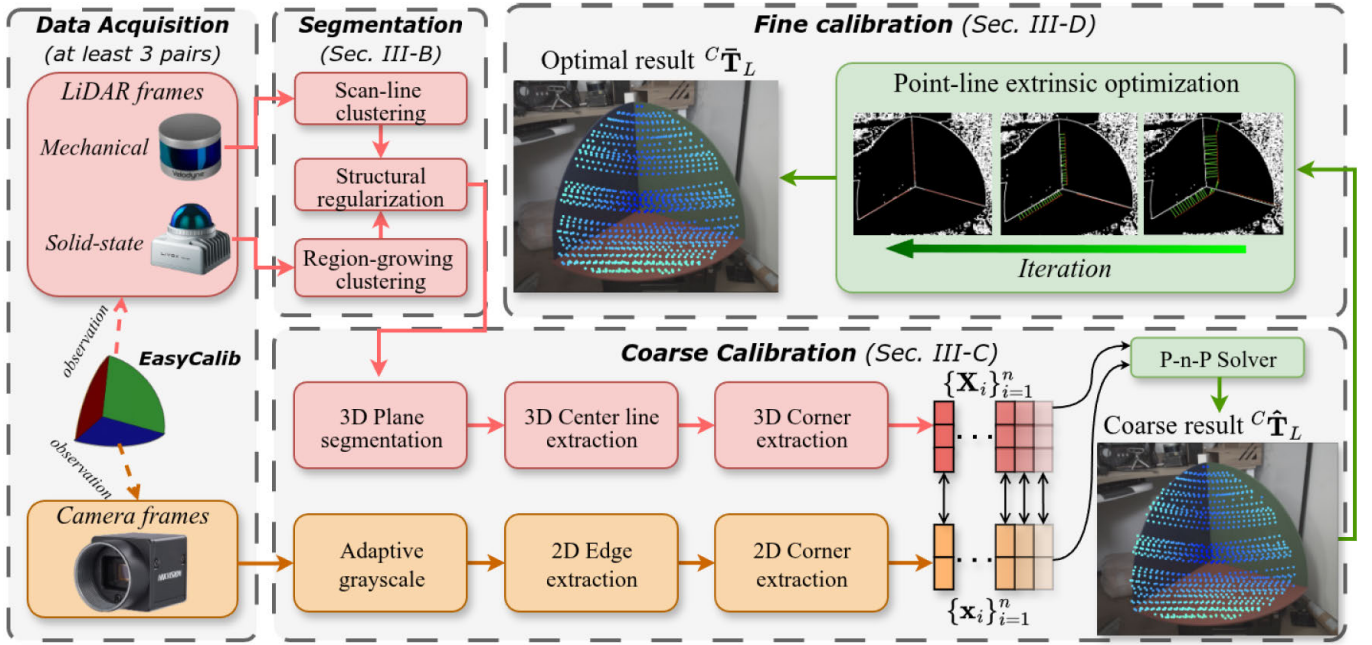


Fig. 2. Overview of the proposed EasyCalib calibration pipeline. Given at least three synchronized LiDAR–camera data pairs, we first segment the EasyCalib target from the LiDAR point cloud via scan-line clustering and structural regularization (Section III-B). We then extract 3-D plane/center-line/corner features from the target and 2-D edge/center-corner features from the trinarized image (Section III-C). A PnP solver provides a coarse extrinsic initialization, which is further refined by a point-to-line reprojection optimization to achieve subpixel alignment (Section III-D).

orthogonal; it only needs to highlight the central intersection lines and corner features. The shape of the planar boards is also not restricted, as the sector shape is merely used to better emphasize the central line features, and in practice, it can be of any shape. Regarding the size of EasyCalib, it is important to note that if the size is too small, the number of points falling on the planar boards in the point cloud may be insufficient, leading to failure in extracting planar features. Therefore, the size should be adjusted to ensure that enough points fall on the planar boards within the effective scanning range of the LiDAR. In our practical application, we set the radius of the three sector-shaped boards of EasyCalib to 38.8 cm, which ensures their proper operation within a LiDAR scanning range of 15 m.

### B. Two-Stage Target Segmentation in Point Cloud

To avoid time-consuming manual labeling, we developed a two-stage segmentation module for the calibration target. First, we perform global clustering on the point cloud using a customized algorithm for different types of LiDAR. Second, we apply a structural regularization step to identify the cluster that contains the calibration target.

1) *Point Cloud Clustering*: For mechanical LiDAR (e.g., Velodyne VLP-16), inspired by [23], we propose an adaptive scan-line clustering algorithm. As shown in Fig. 3(a), we first build a 2-D index matrix over the point cloud based on each point's horizontal and vertical scan angles (rows = scan lines, columns = 4000). We then keep only one point per index cell by averaging all points within each cell, which both deduplicates the cloud and removes outlier noise.

For each row, we determine whether they belong to the same cluster or not by sequentially calculating the spatial

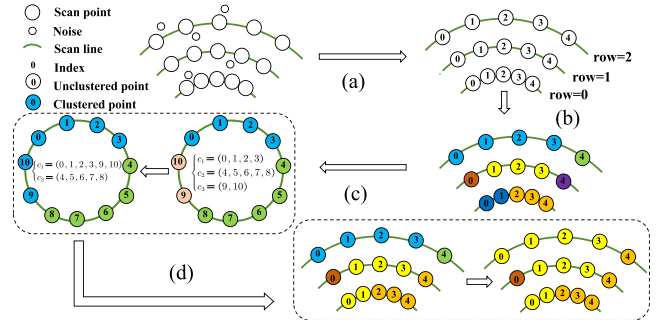


Fig. 3. Single-frame scan-line clustering for sparse LiDAR point cloud. (a) Structured indexing groups points by scan line and assigns an ordered index along each line. (b) Intraline clustering forms 1-D clusters by connecting neighboring points whose spacing is below an adaptive threshold. (c) Endpoint merging connects adjacent 1-D clusters within the same scan line when their endpoints are compatible. (d) Interline clustering merges clusters across neighboring scan lines to produce final 3-D clusters.

distance between points in the row with neighboring indices [as shown in Fig. 3(b)]. Scan points corresponding to the first and last columns of the 2-D index matrix may also be neighboring in space. Therefore, we need to calculate the spatial distance between the endpoint and the first point. If they are neighboring, the clusters to which they, respectively, belong are considered as one cluster [details can be seen in Fig. 3(c)]. Eventually, as shown in Fig. 3(d), for each scan point in row  $i$  ( $i \geq 1$ ), its Euclidean distance to point with the same column on row  $0 \sim (i - 1)$  is calculated, and if they are adjacent (the distance is less than a given threshold  $\sigma$ ), the scan point will be attributed to the cluster with more points. In addition, the target point-cloud density decreases as

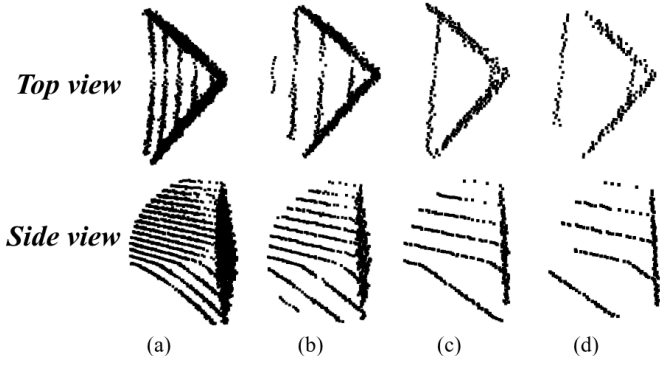


Fig. 4. Point-cloud sparsity of the LiDAR target observations as the scanning distance increases. (a) 1, (b) 2, (c) 3, and (d) 4 m.

the LiDAR scanning distance increases (as shown in Fig. 4). Therefore, we use (1) to dynamically adjust the threshold

$$\sigma = 2r \tan \frac{\theta_v}{2} \quad (1)$$

where  $r$  is the scanning distance and  $\theta_v$  is the vertical angle between the scan point and the horizontal plane, measured from the sensor center.

For nonrepetitive LiDARs (e.g., Livox Mid-360 and Avia), where the point cloud is uniformly distributed in space, we use a region-growing spatial clustering method to obtain more stable results.

2) *Structural Regularization*: Once point cloud clustering is complete, we apply a tailored regularization step to identify the cluster containing EasyCalib. First, we extract all planar features contained in each cluster. For clusters that yield exactly three planes, let the plane normals be  $\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3$  and the plane center points be  $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3$ . We define the cluster center as  $\mathbf{p}_c = (\mathbf{p}_1 + \mathbf{p}_2 + \mathbf{p}_3)/3$ . We then compute two consistency metrics

$$E_{\text{ang}} = \max_{i < j} |\mathbf{n}_i^T \mathbf{n}_j|, \quad E_{\text{dist}} = \max_i |(\mathbf{p}_i - \mathbf{p}_c)^T \mathbf{n}_i|. \quad (2)$$

A cluster is selected as the calibration target if

$$E_{\text{ang}} < \tau_{\text{ang}} \quad \text{and} \quad E_{\text{dist}} < \tau_{\text{dist}} \quad (3)$$

where, in practice, we set  $\tau_{\text{ang}} = 0.2$  and  $\tau_{\text{dist}} = 0.08$  m, which were selected based on extensive empirical evaluation. We further validate the robustness of these thresholds via a sensitivity study in Section IV-B.

### C. Coarse Extrinsic Calibration

The initial extrinsic parameters between the LiDAR and the camera rely on the extraction of 2-D–3-D correspondences. Thanks to the structural design of EasyCalib, its center points exhibit prominent geometric and textural features in both the point cloud and the image. Here, we provide a detailed description of the automatic extraction method for the center corner points of EasyCalib.

1) *For Point Cloud*: After isolating the calibration target (see Section III-B), its main structural features are extracted by planar fitting. Owing to the trihedral structure of the EasyCalib target, this process robustly identifies three mutually

TABLE II  
TARGET SEGMENTATION SR UNDER DIFFERENT THRESHOLD SETTINGS. EASY: SIMULATION SCENES; HARD: REAL-WORLD CLUTTERED SCENES. EACH SETTING IS EVALUATED OVER 50 TRIALS IN EACH SCENARIO (EASY/HARD). BEST RESULTS ARE HIGHLIGHTED AS FIRST, AND SECOND-BEST AS SECOND

| Adaptive $\sigma$ | $\tau_{\text{ang}}$ | $\tau_{\text{dist}}$ (m) | SR <sub>Easy</sub> (%) | SR <sub>Hard</sub> (%) | SR <sub>All</sub> (%) |
|-------------------|---------------------|--------------------------|------------------------|------------------------|-----------------------|
|                   | 0.10                | 0.08                     | 98                     | 90                     | 94                    |
|                   | 0.30                | 0.08                     | 98                     | 90                     | 94                    |
|                   | 0.20                | 0.05                     | 100                    | 88                     | 94                    |
|                   | 0.20                | 0.12                     | 98                     | 92                     | 95                    |
|                   | 0.20                | 0.08                     | 100                    | 90                     | 95                    |
| ✓                 | 0.10                | 0.08                     | 100                    | 90                     | 95                    |
| ✓                 | 0.30                | 0.08                     | 98                     | 92                     | 95                    |
| ✓                 | 0.20                | 0.05                     | 100                    | 90                     | 95                    |
| ✓                 | 0.20                | 0.12                     | 98                     | 92                     | 95                    |
| ✓                 | 0.20                | 0.08                     | 100                    | 92                     | 96                    |

orthogonal planes. Denoting the extracted planes as  $\Pi_1, \Pi_2$ , and  $\Pi_3$ , their equations are formulated as

$$\Pi_i : \mathbf{n}_i^T \mathbf{x} + d_i = 0, \quad i = 1, 2, 3 \quad (4)$$

where  $\mathbf{n}_i$  and  $d_i$  represent the unit normal vector and the signed distance to the origin of the  $i$ th plane, respectively.

To obtain the center lines of the target, the pairwise intersection of planes  $\Pi_i$  and  $\Pi_j$  is computed as

$$L_{ij} = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{x} = \mathbf{p}_{ij} + \lambda (\mathbf{n}_i \times \mathbf{n}_j), \lambda \in \mathbb{R}\} \quad (5)$$

where  $\mathbf{p}_{ij}$  is a point lying on both planes and  $\mathbf{n}_i \times \mathbf{n}_j$  gives the direction vector orthogonal to both normal vectors. The 3-D center corner of EasyCalib is obtained by minimizing the sum of squared Euclidean point-to-line distances to the three intersection lines

$$\mathbf{X}_i = \arg \min_{\mathbf{x}} [\text{dist}^2(\mathbf{x}, L_{12}) + \text{dist}^2(\mathbf{x}, L_{23}) + \text{dist}^2(\mathbf{x}, L_{13})] \quad (6)$$

where  $\text{dist}^2(\mathbf{x}, L)$  denotes the squared Euclidean point-to-line distance in 3-D space. This formulation yields a geometrically stable 3-D reference point for establishing 2-D–3-D correspondences.

2) *For Image*: As illustrated in Fig. 2, we first apply adaptive color-based trinarization to the input image, exploiting the distinctive RGB pattern of the EasyCalib board. Each pixel is classified into one of three color classes based on RGB component dominance

$$k^* = \arg \max_{k \in \{R, G, B\}} \frac{\rho_k}{\rho_R + \rho_G + \rho_B} \quad (7)$$

where  $\rho_k$  represents the intensity of the  $k$ th color channel. This normalized ratio effectively handles illumination variations by focusing on relative color dominance rather than absolute intensity values. The resulting trinarized label map is collapsed into a single-channel intensity image by mapping the three classes to  $\{0, 127, 255\}$ , enhancing the board's structural saliency while suppressing background clutter.

Edge detection is then performed using the Canny operator [24]. Before this step, a light Gaussian smoothing ( $\sigma = 1.0$ ) is applied to mitigate pixel-level noise, and the Canny thresholds

TABLE III

CALIBRATION ERROR [TRANSLATION (CM) AND ROTATION (DEG)] OF DIFFERENT METHODS UNDER THREE SENSOR SETUPS (A–C) WITH THREE TARGET PLACEMENTS (A–C) ON THE SIMULATION DATASET. BEST RESULTS ARE HIGHLIGHTED AS FIRST, AND SECOND-BEST AS SECOND

| Method             | Sensor setup A |       |       |       |       |       | Sensor setup B |       |       |       |       |       | Sensor setup C |       |       |       |       |       |
|--------------------|----------------|-------|-------|-------|-------|-------|----------------|-------|-------|-------|-------|-------|----------------|-------|-------|-------|-------|-------|
|                    | $t_x$          | $t_y$ | $t_z$ | $r_x$ | $r_y$ | $r_z$ | $t_x$          | $t_y$ | $t_z$ | $r_x$ | $r_y$ | $r_z$ | $t_x$          | $t_y$ | $t_z$ | $r_x$ | $r_y$ | $r_z$ |
| <b>Placement A</b> |                |       |       |       |       |       |                |       |       |       |       |       |                |       |       |       |       |       |
| FastCalib [11]     | 18.47          | 18.83 | 21.92 | 1.63  | 1.27  | 1.49  | 12.76          | 13.87 | 18.73 | 1.37  | 0.96  | 1.66  | 10.94          | 11.33 | 16.18 | 1.18  | 0.88  | 1.29  |
| Sphere [12]        | ×              | ×     | ×     | ×     | ×     | ×     | ×              | ×     | ×     | ×     | ×     | ×     | ×              | ×     | ×     | ×     | ×     | ×     |
| Trihedron [16]     | 21.09          | 18.74 | 11.23 | 1.71  | 1.18  | 1.34  | 12.61          | 14.55 | 10.52 | 1.22  | 1.03  | 1.72  | 13.24          | 11.21 | 9.79  | 1.29  | 0.84  | 1.35  |
| EasyCalib (Ours)   | 3.32           | 3.08  | 12.63 | 0.12  | 0.05  | 0.07  | 1.90           | 2.30  | 11.37 | 0.10  | 0.04  | 0.14  | 2.10           | 1.80  | 10.93 | 0.11  | 0.06  | 0.10  |
| <b>Placement B</b> |                |       |       |       |       |       |                |       |       |       |       |       |                |       |       |       |       |       |
| FastCalib [11]     | 19.64          | 18.12 | 24.27 | 1.78  | 1.36  | 1.58  | 13.93          | 14.26 | 19.08 | 1.49  | 1.07  | 1.83  | 11.86          | 12.14 | 17.46 | 1.26  | 0.97  | 1.39  |
| Sphere [12]        | ×              | ×     | ×     | ×     | ×     | ×     | ×              | ×     | ×     | ×     | ×     | ×     | ×              | ×     | ×     | ×     | ×     | ×     |
| Trihedron [16]     | 19.21          | 18.43 | 12.04 | 1.92  | 1.31  | 1.46  | 14.07          | 14.12 | 11.08 | 1.28  | 1.13  | 1.89  | 11.76          | 12.37 | 10.19 | 1.39  | 0.92  | 1.44  |
| EasyCalib (Ours)   | 3.51           | 3.27  | 13.38 | 0.13  | 0.06  | 0.08  | 2.01           | 2.39  | 12.29 | 0.11  | 0.05  | 0.15  | 2.19           | 1.93  | 11.53 | 0.12  | 0.07  | 0.11  |
| <b>Placement C</b> |                |       |       |       |       |       |                |       |       |       |       |       |                |       |       |       |       |       |
| FastCalib [11]     | 5.82           | 4.73  | 6.11  | 0.24  | 0.17  | 0.22  | 3.75           | 3.62  | 4.19  | 0.21  | 0.16  | 0.27  | 2.92           | 2.53  | 2.84  | 0.19  | 0.14  | 0.23  |
| Sphere [12]        | 2.97           | 2.58  | 3.14  | 0.11  | 0.05  | 0.04  | 1.41           | 1.87  | 2.18  | 0.12  | 0.06  | 0.08  | 1.72           | 1.14  | 1.27  | 0.15  | 0.09  | 0.12  |
| Trihedron [16]     | 12.80          | 11.50 | 13.90 | 0.46  | 0.33  | 0.42  | 9.50           | 9.10  | 10.70 | 0.40  | 0.29  | 0.45  | 8.10           | 7.24  | 8.45  | 0.34  | 0.26  | 0.31  |
| EasyCalib (Ours)   | 3.06           | 2.61  | 3.21  | 0.08  | 0.01  | 0.01  | 1.36           | 1.91  | 2.25  | 0.06  | 0.01  | 0.19  | 1.68           | 1.17  | 1.21  | 0.13  | 0.08  | 0.16  |

For methods that are not applicable to Sensor Setup A, we slightly move the sensor and use LiDAR odometry to accumulate denser point clouds to support their operation.  
 × indicates that the method fails to produce a valid calibration result under the corresponding setting.

are adaptively selected based on the median gradient magnitude to maintain edge continuity.

Following edge extraction, the 2-D center corner is identified by analyzing the gradient direction distribution within a circular neighborhood  $\mathcal{N}(\mathbf{u}, r)$  centered at each candidate edge pixel  $\mathbf{u} = (x, y)^T$ . The neighborhood radius is set proportional to the image resolution to ensure consistent behavior across different resolutions:  $r = \lceil \kappa \min(W, H) \rceil$ , where  $W \times H$  denotes the image size. In all experiments, the scale factor is fixed to  $\kappa = 0.01$ . For each edge pixel  $\mathbf{u}' = (x', y')^T$  in  $\mathcal{N}(\mathbf{u}, r)$ , the relative direction is computed as

$$\theta(\mathbf{u}', \mathbf{u}) = \arctan 2(y' - y, x' - x) \quad (8)$$

and accumulated into a direction histogram over  $[-\pi, \pi)$ . We detect the three dominant peaks by nonmaximum suppression in the histogram and retain the top-3 peaks with a minimum angular separation. A pixel is selected as the center corner  $\mathbf{x}_i$  if these peaks are approximately equally spaced (around  $120^\circ$ ) and cover a sufficient angular span, which indicates the intersection of three structural edges. If fewer than three reliable peaks are found (e.g., due to occlusion or color degradation), the candidate is rejected to avoid false positives. The complete procedure is summarized in Algorithm 1.

3) *PnP-Based Coarse Extrinsic Estimation*: Given  $n$  pairs of 2-D–3-D correspondences  $\{\mathbf{x}_i, \mathbf{X}_i\}_{i=1}^n$ , the initial camera–LiDAR extrinsic parameters are estimated by solving a PnP optimization problem

$${}^C\hat{\mathbf{T}}_L = \arg \min_{{}^C\mathbf{T}_L} \sum_{i=1}^n \|\mathbf{x}_i - \Pi({}^C\mathbf{T}_L \mathbf{X}_i)\|_2^2 \quad (9)$$

where  $\Pi(\cdot)$  denotes the camera projection function that maps a 3-D point  $\mathbf{p} = [X, Y, Z]^T$  in the camera frame to the image plane,  $\Pi(\mathbf{p}) = [u, v]^T$ , for example, under a pinhole model  $u = f_x X/Z + c_x$  and  $v = f_y Y/Z + c_y$ . To improve robustness against potential outliers, we adopt the RANSAC-PnP framework [25], [26] to obtain a reliable initial estimate  ${}^C\hat{\mathbf{T}}_L$ .

#### Algorithm 1 2-D Center Corner Test by the Direction Histogram

**Input:** Edge map  $\mathbf{E}$  (Canny; edge pixels are 255), candidate pixel  $\mathbf{u} = (x, y)$ , radius  $r$

**Output:** true if  $\mathbf{u}$  is a center-corner, otherwise false

```

1:  $\mathcal{D} \leftarrow \emptyset$ 
2: for all  $\mathbf{u}' = (x', y') \in \mathcal{N}(\mathbf{u}, r)$  do
3:   if  $\|\mathbf{u}' - \mathbf{u}\| > r$  or  $\mathbf{E}(\mathbf{u}') \neq 255$  then
4:     continue
5:   end if
6:    $\mathcal{D} \leftarrow \mathcal{D} \cup \{\arctan 2(y' - y, x' - x)\}$ 
7: end for
8: if  $|\mathcal{D}| < 10$  then
9:   return false
10: end if
11: Build a circular histogram  $h$  with  $B=9$  bins over  $[-\pi, \pi)$  from  $\mathcal{D}$ .
12: Detect peak bins  $\mathcal{P}$  by local maxima with  $h[i] > 0.2|\mathcal{D}|$ .
13: if  $|\mathcal{P}| \neq 3$  then
14:   return false
15: end if
16: Sort the three peak bins  $i_1 < i_2 < i_3$ , with counts  $c_1, c_2, c_3$ .
17: Compute circular intervals  $d_1 = (i_2 - i_1) \bmod B$ ,  $d_2 = (i_3 - i_2) \bmod B$ ,  $d_3 = (i_1 - i_3) \bmod B$ .
18:  $d_{\min} \leftarrow \min(d_1, d_2, d_3)$ ,  $d_{\max} \leftarrow \max(d_1, d_2, d_3)$ ,  $p \leftarrow \frac{c_1 + c_2 + c_3}{|\mathcal{D}|}$ .
19: return  $\left( \frac{d_{\min}}{d_{\max}} > 0.8 \right) \wedge (p > 0.9)$ .
```

#### D. Fine Extrinsic Calibration

As shown in the optimization part of Fig. 2, we first project the point cloud centerline features onto the corresponding image using  ${}^C\hat{\mathbf{T}}_L$ . It is well known that, given an accurate initial extrinsic estimate, the image edge features and the projected point cloud edge features should be closely aligned. Therefore, for each edge 3-D point  $\mathbf{P}_i$ , we search for a spatially nearest image edge point  $\mathbf{q}_i$  for it and compute its local normal vector  $\mathbf{n}_i$  according to the domain of  $\mathbf{q}_i$ . For each set of edge

point pairs, we build a point-to-line residual

$$\mathbf{e}_i = \mathbf{n}_i^T (\Pi({}^C\mathbf{T}_L \mathbf{P}_i) - \mathbf{q}_i). \quad (10)$$

Then, we construct a weighted least-squares problem using the point-to-line residual  $e_i$  in (14). To enable Lie algebra-based optimization on  $\text{SE}(3)$ , we parameterize the extrinsic transformation by a 6-D twist vector  $\xi = (\phi^T \mathbf{t}^T)^T \in \mathbb{R}^6$ , where  $\phi \in \mathbb{R}^3$  and  $\mathbf{t} \in \mathbb{R}^3$  denote rotation and translation and write  ${}^C\mathbf{T}_L(\xi) = \exp(\xi^\wedge) \in \text{SE}(3)$ . We adopt the left-perturbation update  ${}^C\mathbf{T}_L \leftarrow \exp(\delta\xi^\wedge) {}^C\mathbf{T}_L$ .

1) *Residual Definition*: For each LiDAR point  $\mathbf{P}_i$  (in the LiDAR frame), its coordinates in the camera frame are

$$\mathbf{p}_i = [X_i \ Y_i \ Z_i]^T = {}^C\mathbf{T}_L(\xi) \mathbf{P}_i. \quad (11)$$

Let  $\Pi(\cdot)$  denote the camera projection function. We define

$$\mathbf{f}_i(\xi) = \Pi(\mathbf{p}_i) = \Pi({}^C\mathbf{T}_L(\xi) \mathbf{P}_i) \in \mathbb{R}^2 \quad (12)$$

and the point-to-line residual (signed distance in the image) as

$$e_i = \mathbf{n}_i^T (\mathbf{f}_i(\xi) - \mathbf{q}_i) \quad (13)$$

where  $\mathbf{n}_i \in \mathbb{R}^2$  is the unit normal of the corresponding image line and  $\mathbf{q}_i \in \mathbb{R}^2$  is any point on that line

$${}^C\bar{\mathbf{T}}_L = \arg \min_{{}^C\mathbf{T}_L} \sum_i \frac{1}{2} e_i^T \mathbf{w}_i^{-1} e_i \quad (14)$$

where  ${}^C\bar{\mathbf{T}}_L$  is the optimal extrinsic estimate. In this work, we set  $\mathbf{w}_i = \mathbf{I}$ .

2) *Analytical Jacobian via Chain Rule*: The Jacobian of  $e_i$  w.r.t.  $\xi$  is

$$\mathbf{J}_{p_i} = \frac{\partial e_i}{\partial \xi} = \mathbf{n}_i^T \frac{\partial \mathbf{f}_i}{\partial \xi} = \mathbf{n}_i^T \mathbf{J}_{f_i} \quad (15)$$

with

$$\mathbf{J}_{f_i} = \frac{\partial \mathbf{f}_i}{\partial \xi} = \underbrace{\frac{\partial \Pi(\mathbf{p}_i)}{\partial \mathbf{p}_i}}_{\mathbf{J}_\Pi} \underbrace{\frac{\partial \mathbf{p}_i}{\partial \xi}}_{\mathbf{J}_{\text{SE}(3)}}. \quad (16)$$

Assuming a pinhole model (principal point omitted for clarity),  $\Pi([X, Y, Z]^T) = [f_x X/Z, f_y Y/Z]^T$ , thus,

$$\mathbf{J}_\Pi = \begin{bmatrix} \frac{f_x}{Z_i} & 0 & -\frac{f_x X_i}{Z_i^2} \\ 0 & \frac{f_y}{Z_i} & -\frac{f_y Y_i}{Z_i^2} \end{bmatrix}. \quad (17)$$

Under the left perturbation  ${}^C\mathbf{T}_L \leftarrow \exp(\delta\xi^\wedge) {}^C\mathbf{T}_L$ , the first-order variation of  $\mathbf{p}_i$  is  $\delta\mathbf{p}_i \approx [\delta\phi]_\times \mathbf{p}_i + \delta\mathbf{t}$ , which yields the standard  $\text{SE}(3)$  Jacobian

$$\mathbf{J}_{\text{SE}(3)} = \frac{\partial \mathbf{p}_i}{\partial \xi} = [-[\mathbf{p}_i]_\times \ \mathbf{I}_{3 \times 3}] \quad (18)$$

where  $[\cdot]_\times$  denotes the skew-symmetric matrix.

Combining (16)–(18), we obtain the closed-form  $\mathbf{J}_{f_i} \in \mathbb{R}^{2 \times 6}$

$$\mathbf{J}_{f_i} = \begin{bmatrix} -\frac{f_x X_i Y_i}{Z_i^2} & f_x + \frac{f_x X_i^2}{Z_i^2} & -\frac{f_x Y_i}{Z_i} & \frac{f_x}{Z_i} & 0 & -\frac{f_x X_i}{Z_i^2} \\ -f_y & -\frac{f_y Y_i^2}{Z_i^2} & \frac{f_y X_i Y_i}{Z_i^2} & \frac{f_y X_i}{Z_i} & 0 & \frac{f_y}{Z_i} - \frac{f_y Y_i}{Z_i^2} \end{bmatrix}. \quad (19)$$

Finally, we solve (14) using the Gauss–Newton algorithm and obtain the refined extrinsic parameters  ${}^C\bar{\mathbf{T}}_L$  with subpixel accuracy.

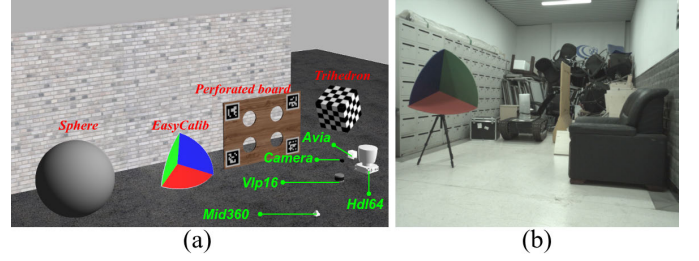


Fig. 5. (a) Our simulation environment built on Gazebo. (b) Calibration scene in the real world.

#### IV. EXPERIMENTAL RESULTS

To comprehensively validate the applicability of EasyCalib, we conducted experiments in both simulated and real-world environments. Fig. 5 illustrates the experimental configurations in both simulation and real-world scenarios. We categorize the sensors into three setups: a mechanical LiDAR (Velodyne VLP-16) and two nonrepetitive-scan LiDARs (Livox Mid-360 and Livox Avia). For the camera, we use a standard pinhole model in all setups. For other camera types, such as fisheye or wide-angle cameras, the images can be readily rectified to a pinhole model using a generic distortion correction model; therefore, we do not conduct separate experiments with different camera configurations.

##### A. Evaluation Metrics

We evaluate the calibration accuracy differently in simulation and real-world experiments due to the availability of ground-truth extrinsics.

1) *Simulation (With Ground Truth)*: In simulation, the ground-truth LiDAR–camera extrinsic transformation  ${}^C\mathbf{T}_L^{gt}$  is known. Given an estimated extrinsic  ${}^C\bar{\mathbf{T}}_L$ , we report the componentwise translation errors

$$\Delta \mathbf{t} = [\Delta t_x \ \Delta t_y \ \Delta t_z]^T = {}^C\bar{\mathbf{t}}_L - {}^C\mathbf{t}_L^{gt} \quad (20)$$

and the overall translation error  $e_t = \|\Delta \mathbf{t}\|_2$ .

For rotation, we compute the relative rotation  $\Delta \mathbf{R} = ({}^C\mathbf{R}_L^{gt})^T {}^C\bar{\mathbf{R}}_L$ . We report the componentwise rotation errors  $(\Delta r_x, \Delta r_y, \Delta r_z)$  by converting  $\Delta \mathbf{R}$  to Euler-angle increments (roll, pitch, yaw), and the overall rotation error

$$e_r = \arccos \left( \frac{\text{tr}(\Delta \mathbf{R}) - 1}{2} \right). \quad (21)$$

2) *Real-World (Without Reliable Ground Truth)*: In real-world experiments, we do not directly quantify absolute calibration accuracy in the same way as in simulation. Instead, the real-world evaluation is intended to assess the repeatability, stability, and practical consistency of the estimated extrinsics under repeated trials. Specifically, we run the calibration multiple times under the same setting (e.g., different data pairs, random subsets, or repeated captures) and analyze the variation of the estimated 6-D poses. Let  $\xi_k = [\phi_k^T \ \mathbf{t}_k^T]^T$  denote the extrinsic estimate from trial  $k$ , where  $\phi_k$  represents the

roll-pitch-yaw angles and  $\mathbf{t}_k$  the translation. We compute the sample mean and standard deviation for each component

$$\mu_j = \frac{1}{K} \sum_{k=1}^K \xi_{k,j}, \quad \sigma_j = \sqrt{\frac{1}{K-1} \sum_{k=1}^K (\xi_{k,j} - \mu_j)^2} \quad (22)$$

where  $j \in \{t_x, t_y, t_z, r_x, r_y, r_z\}$  and  $K$  is the number of trials. We report  $\sigma_{t_x}, \sigma_{t_y}, \sigma_{t_z}$  and  $\sigma_{r_x}, \sigma_{r_y}, \sigma_{r_z}$  to quantify the stability of the estimated extrinsics in real-world scenarios.

3) *Target Segmentation Evaluation*: We evaluate the target segmentation using the success rate (SR). A frame is counted as successful if the cluster returned by the segmentation stage corresponds to the calibration target; otherwise, it is counted as a failure. The SR is defined as

$$\text{SR} = \frac{N_{\text{succ}}}{N_{\text{all}}} \quad (23)$$

where  $N_{\text{all}}$  is the total number of test frames and  $N_{\text{succ}}$  is the number of frames in which the target cluster is correctly selected. In practice, we determine success by verifying the selected cluster against the ground-truth target association in simulation, and by manual inspection (or a target visibility check) for a small subset of real-world frames.

### B. Target Segmentation Evaluation

We evaluate the robustness of the proposed target segmentation using the SR. A trial is counted as successful if the pipeline selects the correct target cluster and yields valid 2-D–3-D correspondences for subsequent coarse calibration. We consider two scenarios: easy (simulation) and hard (real-world clutter). Each setting is evaluated over 50 trials per scenario.

Table II summarizes SR under different combinations of the structural-regularization thresholds ( $\tau_{\text{ang}}, \tau_{\text{dist}}$ ) and whether the adaptive clustering threshold  $\sigma$  is enabled. Overall, SR remains high across a wide range of ( $\tau_{\text{ang}}, \tau_{\text{dist}}$ ), indicating that the segmentation is not overly sensitive to these parameters. In the easy scenario, most settings achieve  $\geq 98\%$  SR. In the hard scenario, SR slightly drops due to clutter and trihedral-like distractors, yet the performance differences among threshold settings remain small.

Enabling adaptive  $\sigma$  yields the best overall performance and improves robustness, particularly in the hard scenario. Fig. 6 further illustrates that adaptive thresholding produces more consistent clustering across different target distances, whereas fixed thresholding is more prone to fragmentation at longer ranges. Failures in real-world clutter mainly occur under severe occlusion/partial scans or when strong background structures induce ambiguous trihedral candidates, in which case the method tends to reject uncertain detections rather than output unreliable correspondences.

### C. Coarse Calibration Performance

1) *Corner Feature Extraction*: We evaluate the proposed 2-D corner feature detection method under diverse imaging conditions, including normal/low-light environments and varying degrees of occlusion. As shown in Fig. 7, our method reliably localizes the center-corner feature in all tested cases.

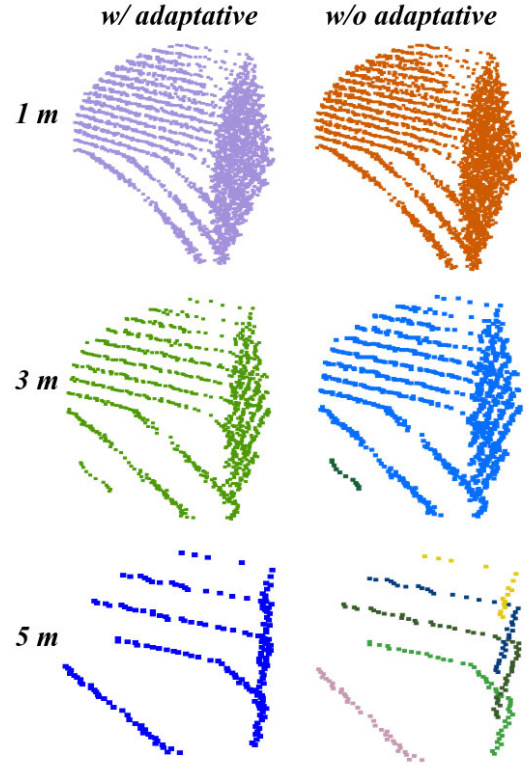


Fig. 6. Target clustering results for a mechanical LiDAR at different distances. Left column: results using adaptive thresholding; right column: results without adaptive thresholding. The same color denotes the same cluster.

The corresponding gradient-direction histograms exhibit three dominant peaks with approximately even angular separation, which is consistent with the expected trihedral edge intersection and thus provides an intuitive verification of detection correctness. Fig. 8 further presents the extracted 3-D edge and planar features for different LiDAR types. Despite the substantially sparser measurements of the low-channel Velodyne VLP-16, the proposed pipeline can still stably recover the target planes and their intersection edges, yielding consistent 3-D corner features. This robustness stems from our geometry-aware segmentation and structural regularization, which leverage the trihedral constraints to suppress outliers and compensate for reduced point density. Overall, these results demonstrate that the proposed feature extraction is not only resilient to challenging visual conditions but also transferable across LiDAR sensors ranging from sparse mechanical scanners to denser solid-state devices.

2) *Feature Matching Evaluation*: We evaluate the quality of coarse calibration by the match ratio, defined as the percentage of 3-D edge points that can be matched to 2-D image edges after projecting with the coarse extrinsic (nearest-neighbor in the image with a fixed pixel tolerance). Fig. 9 shows the match ratio distributions over 20 trials for each target placement and three LiDARs.

Overall, Mid-360 and Avia consistently achieve higher match ratios than the sparser VLP-16, while VLP-16 remains stable with only a slightly larger variance. Placements with depth diversity (“4 diff depths” and “3 diff depths”)

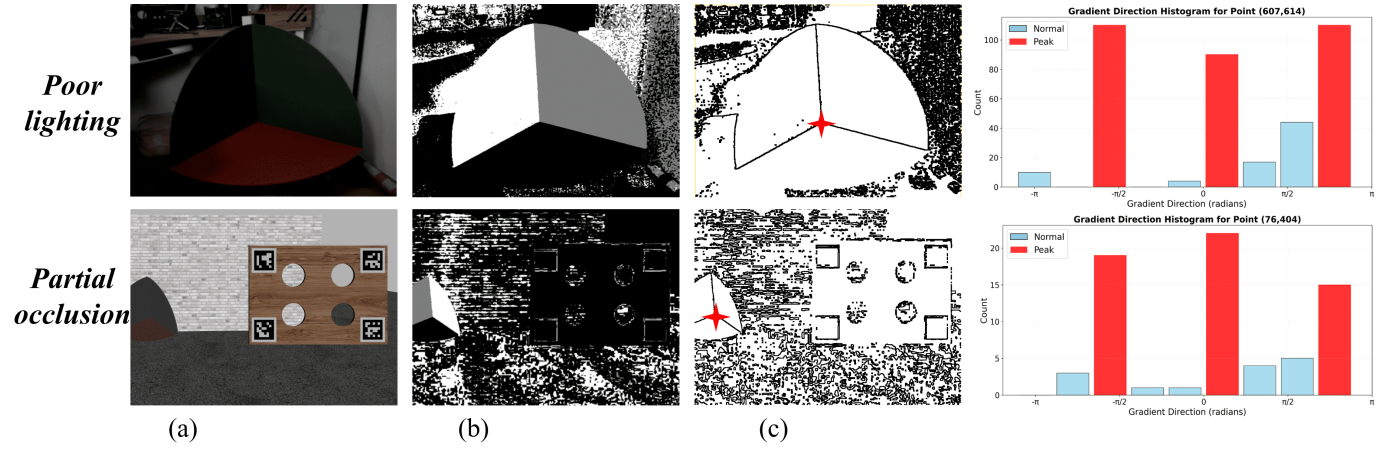


Fig. 7. Qualitative results and intermediate outputs of the proposed 2-D corner feature detection under different imaging conditions, including low-light and occluded scenes. The table on the right shows the gradient-direction statistics of the successfully detected center corner feature. (a) Camera image. (b) Trinarization. (c) Edge detection and 2-D center corner  $\star$  detection.

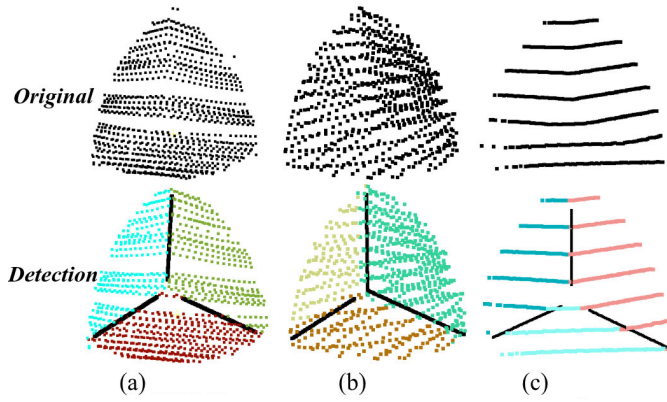


Fig. 8. 3-D edge feature extraction results for different LiDAR types. Points with the same color correspond to the same planar feature. (a) Mid360. (b) Avia. (c) Vlp16.

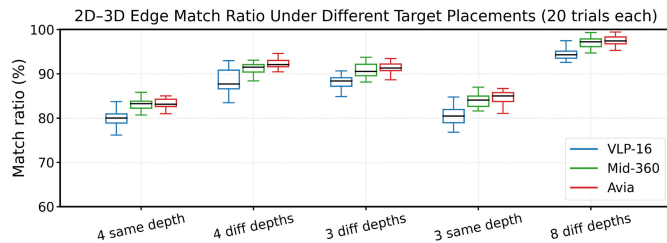


Fig. 9. Feature matching performance after coarse calibration under different target placement settings. The boxplots show the match ratio (matched 2-D-3-D edge correspondences over all 3-D edge points) over 20 trials per setting for three LiDAR types (VLP-16, Mid-360, and Avia). Boxes indicate the interquartile range, the center line denotes the median, and whiskers show the nonoutlier range.

outperform same-depth placements, confirming that depth variation provides stronger geometric constraints for PnP initialization. The “8 diff depths” setting yields the best performance (above 95%), indicating that increasing both the number of observations and depth diversity further improves coarse alignment robustness.

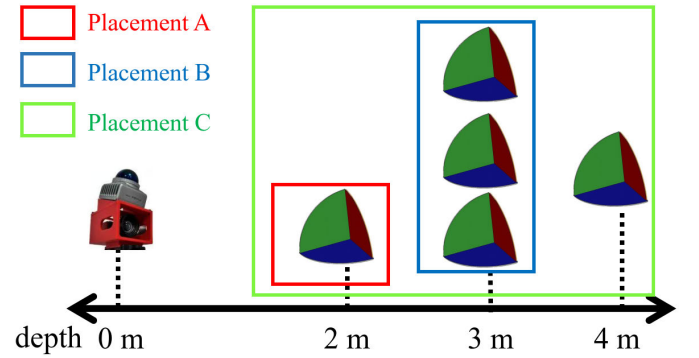


Fig. 10. Different target placement settings used to evaluate the performance of various calibration methods. EasyCalib is shown here for illustration, and the same settings are applied to the other methods.

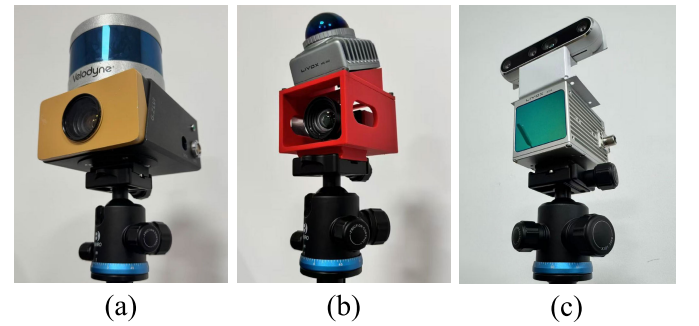


Fig. 11. Our sensor setups. (a) Velodyne VLP-16 LiDAR and MER2630-60U3C camera (3088  $\times$  2064). (b) Livox MID-360 LiDAR and MV-CU013-A0UC camera (1280  $\times$  1024). (c) Livox Avia LiDAR and real sense D455 camera (1280  $\times$  720).

3) *Potential Failure Cases*: In practice, the only factor that can cause EasyCalib to fail is an unsuccessful *coarse calibration* stage. Since coarse calibration relies on solving a PnP problem, the pipeline may fail when the extracted 2-D-3-D correspondences are unreliable, resulting in an incorrect initial extrinsic estimate and preventing the subsequent fine optimization from converging to the correct solution. This

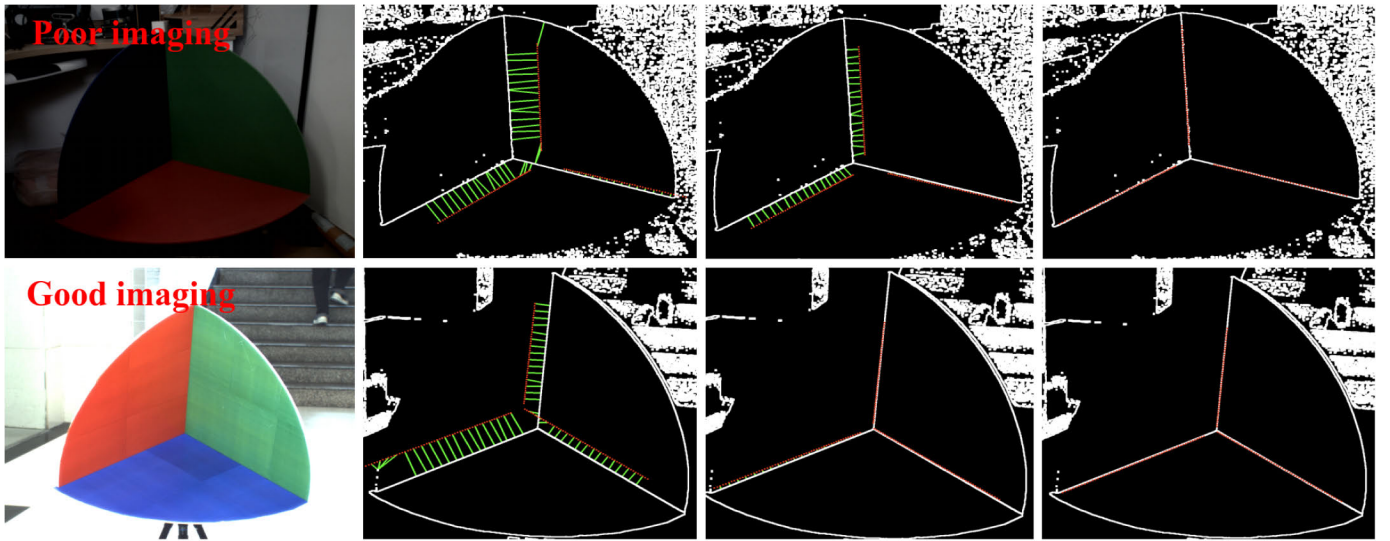


Fig. 12. Point-to-line residual iteration process. White: 2-D edges; red: 3-D centerline features projected onto the image plane; green: residual directions. The first column shows camera images captured under two different lighting conditions. The second column shows the point-to-line residuals constructed from the initial estimate. The fourth column shows the final converged result.

typically happens in two cases: 1) *severe target occlusion*, where the center-corner feature is blocked such that the characteristic three-color distribution around the corner cannot be observed, leading to unreliable 2-D corner detection and PnP failure; and 2) *extremely sparse LiDAR sampling on the target*, which is more likely for low-channel Velodyne-type LiDARs—when fewer than two scan lines intersect the target planes, plane segmentation becomes unreliable, degrading the extraction of 3-D edge and center-corner features and thus causing the coarse calibration to fail.

#### D. Fine Calibration Performance

1) *Simulation Results*: Table III reports the fine-calibration accuracy under three sensor setups and three target placements (as shown in Fig. 10). Placement A uses a single target, Placement B places multiple targets at the same depth, and Placement C places multiple targets at different depths. Overall, EasyCalib achieves the best accuracy on most DoFs across all sensor setups, confirming the effectiveness of the proposed coarse-to-fine pipeline and the pixel-level refinement.

A notable exception appears in the depth-related translation component  $t_z$  under Placements A and B. In these two settings, both FastCalib [11] and EasyCalib exhibit larger  $t_z$  errors compared with the trihedron baseline [16], which is attributed to limited depth constraints: a single target (Placement A) or coplanar targets at the same depth (Placement B) reduce the observability of the depth component and make the solution more sensitive to correspondence noise and initialization. Despite this inherent limitation, EasyCalib still provides substantially lower errors than FastCalib in the remaining translation and rotation components, leading to better overall calibration quality. In contrast, the trihedron-based method consistently yields higher errors across most components. This is expected since it mainly relies on cross-modal pose alignment of geometric structures without enforcing pixel-level reprojection consistency, which limits its ability to achieve

fine, subpixel alignment. Finally, Placement C (multiple targets with different depths) offers stronger geometric constraints and improves depth observability, where EasyCalib maintains the best overall performance across all sensor setups.

We also compare against the sphere-based method [12]. Although the sphere approach can achieve competitive accuracy in Placement C and even attains the best results in some DoFs, it requires multiple observations to reliably estimate the extrinsics (e.g., several sphere poses). Therefore, under Placements A and B, where the available data pairs do not satisfy the minimum requirement, the sphere method fails to produce valid calibration results. Finally, Placement C (multiple targets with different depths) offers stronger geometric constraints and improves depth observability, where EasyCalib maintains the best overall performance across all sensor setups.

2) *Real-World Results*: As shown in Fig. 11, three real-world sensor setups were evaluated: a mechanical Velodyne VLP-16 LiDAR (setup A), a semi-solid-state Livox Mid-360 LiDAR (setup B), and a Livox Avia LiDAR (setup C). Each LiDAR was paired with a pinhole camera (for setup C, only the monocular stream of the D455 camera was used). Since accurate ground-truth extrinsics are unavailable in real scenes, we assess repeatability by running EasyCalib  $K = 20$  times under the same setting and reporting the standard deviation of each 6-DoF component, as summarized in Table IV.

Overall, EasyCalib exhibits strong consistency across all setups, with rotation standard deviations remaining below  $0.1^\circ$  and translation standard deviations typically within subcentimeter to a few centimeters. As expected, multiple-depth target placements yield more stable results than single-target placements, since the added depth diversity provides stronger geometric constraints and improves observability. In contrast, the single placement shows noticeably larger variance in the depth translation component  $\sigma_{t_z}$ , indicating that the depth direction is more sensitive when the constraints are limited.

TABLE IV

REAL-WORLD REPEATABILITY OF EASYCALIB (WITHOUT RELIABLE GROUND TRUTH). WE REPORT THE SAMPLE STANDARD DEVIATION OF THE ESTIMATED EXTRINSICS OVER  $K = 20$  REPEATED TRIALS UNDER DIFFERENT SETTINGS. SMALLER VALUES INDICATE BETTER REPEATABILITY. BEST RESULTS ARE HIGHLIGHTED AS FIRST, AND SECOND-BEST AS SECOND

| Target placement |          | Imaging quality |      | Sensor setup |   |   | Rotation std (deg) |                |                | Translation std (cm) |                |                |
|------------------|----------|-----------------|------|--------------|---|---|--------------------|----------------|----------------|----------------------|----------------|----------------|
| Single           | Multiple | Poor            | Good | A            | B | C | $\sigma_{r_x}$     | $\sigma_{r_y}$ | $\sigma_{r_z}$ | $\sigma_{t_x}$       | $\sigma_{t_y}$ | $\sigma_{t_z}$ |
| ✓                |          | ✓               |      | ✓            |   |   | 0.070              | 0.068          | 0.066          | 0.63                 | 0.66           | 5.32           |
| ✓                |          |                 | ✓    | ✓            |   |   | 0.067              | 0.065          | 0.059          | 0.57                 | 0.61           | 4.24           |
|                  | ✓        | ✓               |      | ✓            |   |   | 0.064              | 0.061          | 0.057          | 0.59                 | 0.57           | 0.48           |
|                  | ✓        |                 | ✓    | ✓            |   |   | 0.053              | 0.052          | 0.061          | 0.55                 | 0.58           | 0.46           |
| ✓                |          | ✓               |      |              | ✓ |   | 0.065              | 0.062          | 0.058          | 0.60                 | 0.59           | 2.18           |
| ✓                |          |                 | ✓    |              | ✓ |   | 0.061              | 0.054          | 0.051          | 0.54                 | 0.51           | 2.08           |
|                  | ✓        | ✓               |      |              | ✓ |   | 0.047              | 0.043          | 0.049          | 0.47                 | 0.52           | 0.42           |
|                  | ✓        |                 | ✓    |              | ✓ |   | 0.036              | 0.035          | 0.038          | 0.40                 | 0.50           | 0.41           |
| ✓                |          | ✓               |      |              |   | ✓ | 0.058              | 0.055          | 0.051          | 0.55                 | 0.54           | 4.02           |
| ✓                |          |                 | ✓    |              |   | ✓ | 0.052              | 0.049          | 0.042          | 0.47                 | 0.46           | 2.94           |
|                  | ✓        | ✓               |      |              |   | ✓ | 0.053              | 0.046          | 0.038          | 0.46                 | 0.47           | 0.37           |
|                  | ✓        |                 | ✓    |              |   | ✓ | 0.049              | 0.037          | 0.026          | 0.35                 | 0.35           | 0.23           |

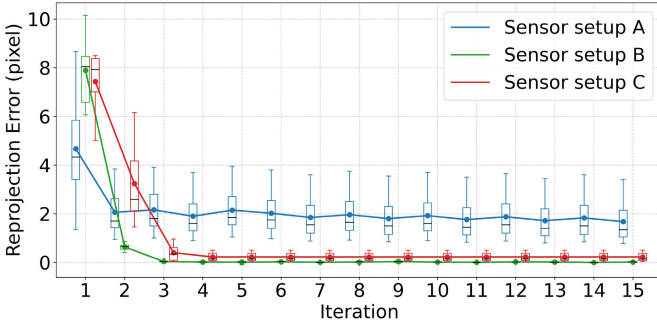


Fig. 13. Convergence process of point-to-line reprojection error of different sensor setups.

3) *Robustness Analysis*: To validate the convergence behavior of EasyCalib in the fine-calibration stage, we track the point-to-line reprojection residuals  $e_i$  in (10) under different sensor setups. Fig. 12 shows that our method can produce stable feature-matching results and quickly align these features under different lighting conditions. As shown in Fig. 13, the optimization rapidly decreases the residual and typically reaches a stable level within 3–4 Gauss–Newton iterations, thanks to the accurate coarse initialization (Section III-C). We nevertheless plot a larger number of iterations to visualize the postconvergence trend and confirm that the residual remains stable without oscillation. In implementation, the solver uses a standard stopping criterion: it terminates when the relative change of the objective (or the update norm  $|\Delta\xi|$ ) falls below a preset tolerance, or when a maximum iteration number is reached.

We also note the larger residual magnitude in Setup A. This is mainly due to the higher image resolution of the MER2630-60U3C camera used in Setup A ( $3088 \times 2064$ ),

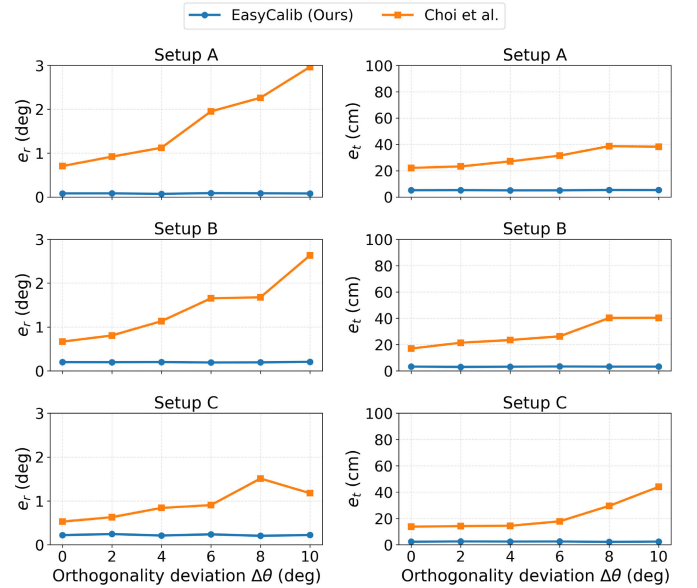


Fig. 14. Orthogonality sensitivity in simulation. We introduce angular deviations  $90^\circ \pm \Delta\theta$  to the trihedron interplane angles and compare  $e_t$  and  $e_r$  under three sensor setups. EasyCalib remains stable, while Choi et al. [16] degraded notably as  $\Delta\theta$  increases.

which yields reprojection residuals in pixel units that are naturally larger than those of Setups B/C with lower-resolution cameras (see Fig. 11 for camera specifications). Importantly, despite the different scales, all setups exhibit the same fast convergence behavior and stable postconvergence residuals.

4) *Sensitivity to Trihedron Orthogonality*: We further evaluate the robustness to imperfect target manufacturing/placement by intentionally breaking the orthogonality of the trihedron in simulation. For each deviation level  $\Delta\theta$ , we randomly perturb

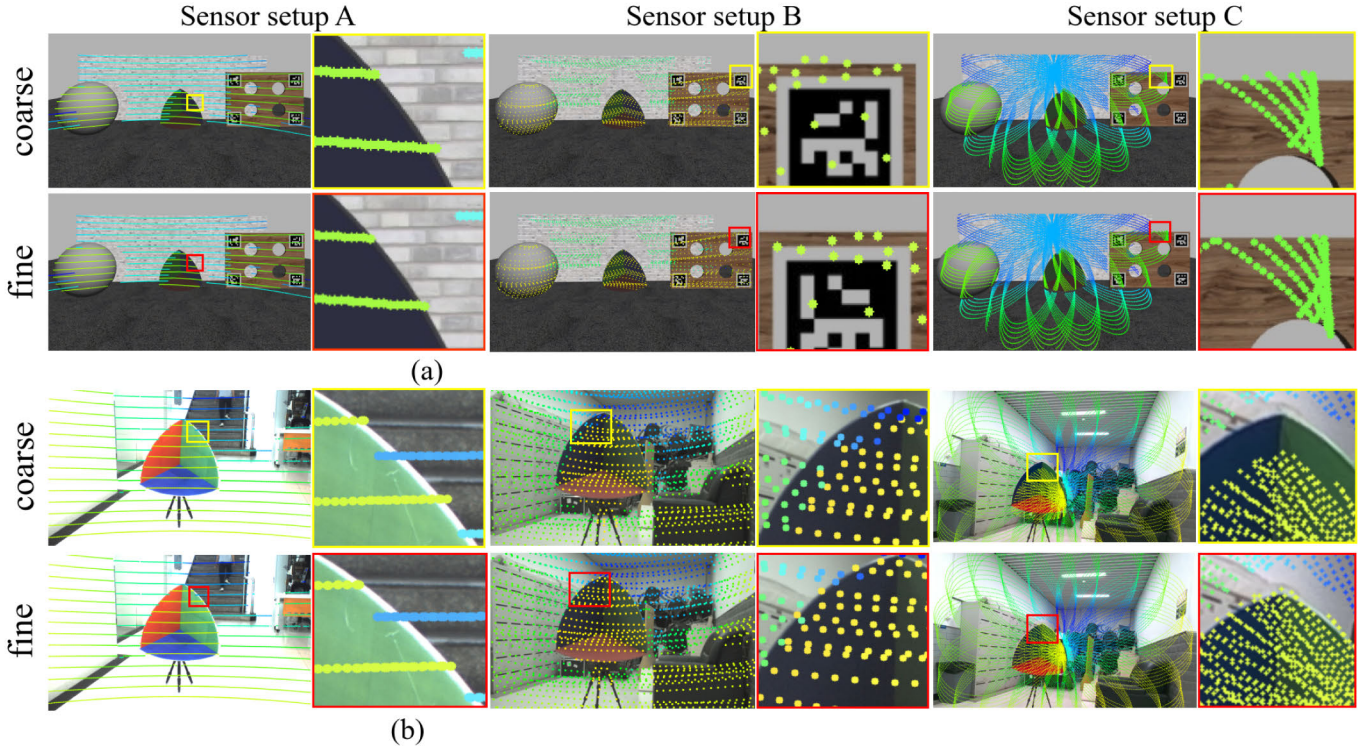


Fig. 15. Visualization of calibration result projections under different sensor setups. We show the projection results of the coarse and fine calibration on both simulated and real-world data, with the projected points colored by depth. Zoomed-in view for details. (a) Reprojection results on our simulation datasets. (b) Reprojection results on our self-recorded datasets.

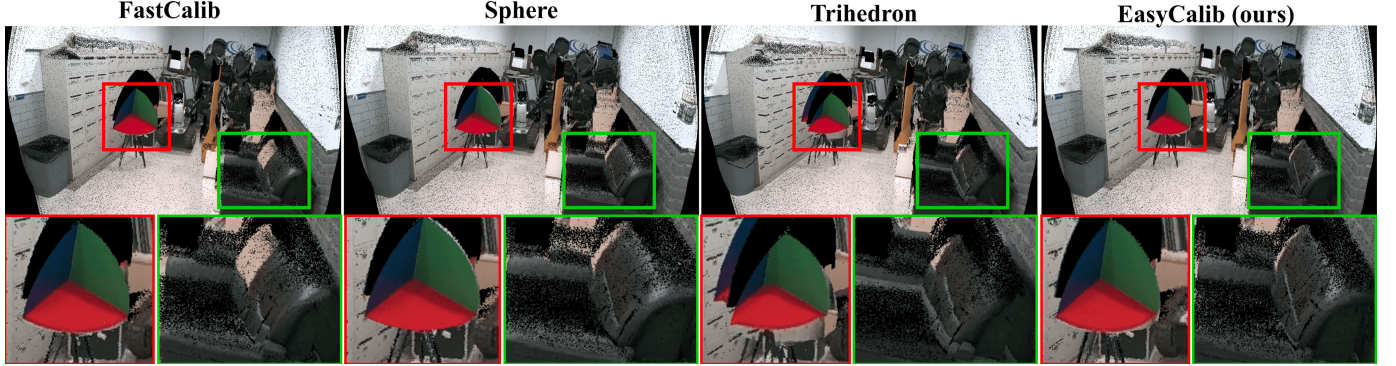


Fig. 16. Point cloud colored using the estimated extrinsics from FastCalib [11], Sphere [12], Trihedron [16], and EasyCalib (from left to right). The red box and green box highlight the same region for better comparison.

the dihedral angles between each pair of planes (i.e.,  $90^\circ \pm \Delta\theta$ ) while keeping all other settings unchanged and report the translation and rotation errors ( $e_t$  and  $e_r$ ) with respect to the ground-truth extrinsics. Fig. 14 summarizes the results for both EasyCalib and the trihedron-based method of Choi et al. [16] under three sensor setups.

As  $\Delta\theta$  increases, Choi et al. [16] exhibited a clear degradation trend in both  $e_r$  and  $e_t$  across all setups, with larger fluctuations at high nonorthogonality. This is expected because its initialization relies on the assumption of an orthogonal trihedron to align the LiDAR and camera poses; once the target deviates from orthogonality, the estimated cross-modal pose alignment becomes biased, leading to rapidly increased errors. In contrast, EasyCalib remains largely insensitive to orthogonality violations:  $e_r$  stays at a low level and  $e_t$  remains

bounded (within 1 m), showing only mild nonmonotonic variations rather than a consistent upward trend. This behavior indicates that EasyCalib does not critically depend on strict orthogonality and can tolerate angular deviations in practice, which is beneficial under manufacturing tolerances and non-ideal target placement. Finally, although the absolute error magnitudes differ slightly across sensor setups due to different sensor characteristics, the same trend holds consistently: EasyCalib maintains stable performance while Choi et al. [16] is highly sensitive to nonorthogonality.

#### E. Calibration Result Evaluation

1) *Reprojection*: As shown in Fig. 15, we can observe that after coarse calibration, the point cloud projections already roughly align with the image edge features, although some

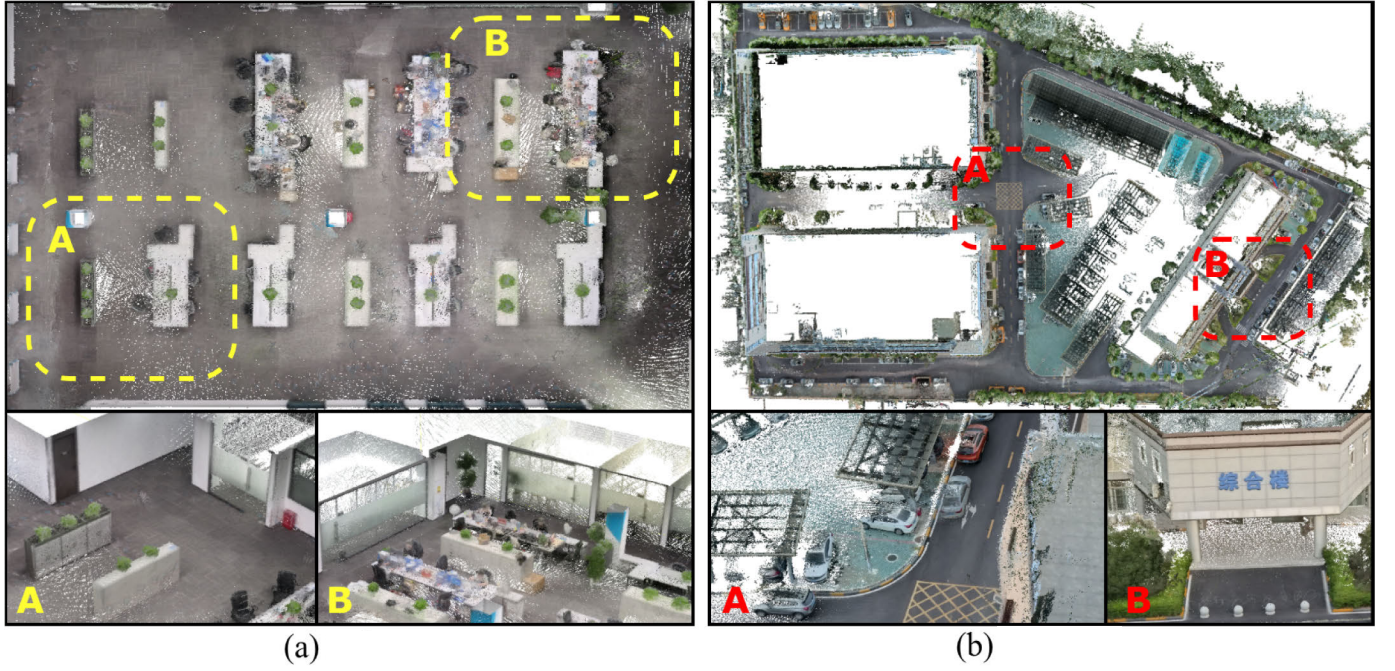


Fig. 17. We build (a) indoor and (b) outdoor colored point-cloud maps using FastLIVO2 [27] with the extrinsic parameters estimated by EasyCalib. The highlighted regions show fine-grained details: in indoor scenes, the edges of walls, cabinets, and plants are clearly reconstructed; in outdoor scenes, detailed structures such as vehicles and building facades are well preserved. These results demonstrate the high accuracy and practical applicability of our method.

noticeable deviations remain (e.g., slight overshoots beyond the edge contours or minor misalignments in certain regions). Nevertheless, this demonstrates the reliability of the coarse estimate. After fine calibration, the projections adhere much more closely to the image edges, indicating that our method effectively refines the calibration accuracy.

2) *RGB-Colored Mapping*: To minimize additional sources of error in the evaluation, we exploit the nonrepetitive scanning property of solid-state LiDARs and obtain a dense environmental point cloud by precisely positioning the Livox Avia. We then colorize the point cloud using the extrinsics estimated by different calibration methods. As shown in Fig. 16, EasyCalib produces noticeably better alignment in boundary regions, especially in detail-rich areas such as marker edges and sofa contours. Although minor color bleeding can still be observed around some edges, EasyCalib remains superior to the compared methods.

Fig. 17 further presents an indoor RGB-colored point cloud map built by FastLIVO2 [27] using the extrinsics estimated by EasyCalib. Benefiting from the high-accuracy extrinsic calibration, the alignment between the point cloud and images is significantly improved, enabling clear reconstruction of fine structures in the map, such as wall boundaries, cabinet edges, and plants. These results further demonstrate the practicality of EasyCalib and its potential for real-world deployment.

#### F. Runtime Analysis

We tested the runtime performance of EasyCalib on a NUC equipped with an Intel i7-1360P CPU and 16 GB RAM. Table V reports the runtime as mean $\pm$ std over 20 repeated runs for each sensor setup. Overall, EasyCalib finishes the

TABLE V  
RUNTIME BREAKDOWN OF EASYCALIB UNDER DIFFERENT SENSOR SETUPS. EACH ENTRY IS REPORTED AS MEAN $\pm$ STD OVER 20 REPEATED RUNS. TIMES ARE IN SECONDS UNLESS OTHERWISE NOTED

| Calibration stage              | Setup A                         | Setup B                         | Setup C                         |
|--------------------------------|---------------------------------|---------------------------------|---------------------------------|
| <b>3D corner extraction</b>    |                                 |                                 |                                 |
| Point-cloud clustering         | 0.04 $\pm$ 0.01                 | 0.50 $\pm$ 0.02                 | 0.20 $\pm$ 0.02                 |
| Structural regularization      | 0.07 $\pm$ 0.01                 | 0.15 $\pm$ 0.01                 | 0.12 $\pm$ 0.01                 |
| Feature extraction             | 0.03 $\pm$ 0.01                 | 0.02 $\pm$ 0.01                 | 0.17 $\pm$ 0.02                 |
| <i>Subtotal</i>                | <b>0.14<math>\pm</math>0.08</b> | <b>0.67<math>\pm</math>0.04</b> | <b>0.49<math>\pm</math>0.05</b> |
| <b>2D corner extraction</b>    |                                 |                                 |                                 |
| Trinarization                  | 0.05 $\pm$ 0.01                 | 0.01 $\pm$ 0.00                 | 0.01 $\pm$ 0.00                 |
| Edge detection                 | 0.18 $\pm$ 0.04                 | 0.04 $\pm$ 0.01                 | 0.03 $\pm$ 0.01                 |
| Feature extraction             | 0.30 $\pm$ 0.08                 | 0.07 $\pm$ 0.02                 | 0.05 $\pm$ 0.01                 |
| <i>Subtotal</i>                | <b>0.53<math>\pm</math>0.12</b> | <b>0.12<math>\pm</math>0.03</b> | <b>0.09<math>\pm</math>0.01</b> |
| <b>Coarse calibration (ms)</b> | <b>0.17<math>\pm</math>0.01</b> | <b>0.22<math>\pm</math>0.01</b> | <b>0.11<math>\pm</math>0.01</b> |
| <b>Fine calibration</b>        |                                 |                                 |                                 |
| Feature matching (ms)          | 0.05 $\pm$ 0.02                 | 0.09 $\pm$ 0.03                 | 0.04 $\pm$ 0.02                 |
| Iterations (#)                 | 3 $\pm$ 1                       | 4 $\pm$ 1                       | 4 $\pm$ 1                       |
| Per iteration (ms)             | 0.03 $\pm$ 0.01                 | 0.17 $\pm$ 0.12                 | 0.02 $\pm$ 0.01                 |
| <i>Subtotal (ms)</i>           | <b>0.10<math>\pm</math>0.03</b> | <b>0.68<math>\pm</math>0.08</b> | <b>0.10<math>\pm</math>0.04</b> |
| <b>Total (s)</b>               | <b>0.67<math>\pm</math>0.14</b> | <b>0.79<math>\pm</math>0.05</b> | <b>0.58<math>\pm</math>0.05</b> |

complete extrinsic calibration in  $< 1$  s for all configurations, indicating that the pipeline is efficient and stable in practice.

The runtime is dominated by the feature extraction stage, especially the 3-D corner extraction. In particular, point-cloud clustering is the main bottleneck in Setup B (0.50 s), which is consistent with its larger and denser point set. Setup C also spends more time on 3-D processing due to higher point density and a heavier plane/edge extraction step. In contrast, Setup A has a lightweight 3-D processing cost. For the 2-D corner extraction, Setup A is significantly slower than

Setups B/C, mainly because it uses a much higher-resolution camera (Fig. 11), which increases the cost of trinarization, edge detection, and corner scoring. Importantly, the reported standard deviations are small relative to the means, suggesting consistent runtime across repeated runs. Both coarse calibration and fine calibration are computationally negligible compared with data processing. Coarse calibration takes only 0.1–0.2 ms, and the fine optimization converges in 3–4 iterations with a submillisecond total time. This breakdown confirms that EasyCalib is primarily limited by the front-end feature extraction (clustering and image processing), while the back-end optimization introduces minimal overhead and thus scales well with the number of iterations.

## V. CONCLUSION

In this article, we present a novel artificial target, EasyCalib, specifically designed for the extrinsic calibration between cameras and LiDAR sensors. Building upon its structured geometry and distinctive color patterns, we further develop a fully automatic and robust calibration algorithm that achieves subpixel accuracy without requiring any manual segmentation or initial extrinsic estimates. The proposed method significantly simplifies the calibration process while ensuring high precision and strong robustness in challenging environments. In future work, we plan to redesign the EasyCalib target for thermal infrared cameras by integrating heating elements on the three planar panels to generate distinctive thermal-radiation patterns, enabling fully automatic calibration between thermal cameras and LiDAR. In addition, extending EasyCalib to multi-LiDAR systems will be a key direction. We will leverage the EasyCalib center-line features observed by different LiDARs and formulate optimization constraints between 3-D line correspondences to achieve automatic extrinsic calibration for multi-LiDAR setups.

## ACKNOWLEDGMENT

The numerical calculations in this article were supported by the supercomputer systems in the Supercomputing Center of Wuhan University, Wuhan, China.

## REFERENCES

- [1] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 4164–4169.
- [2] 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.*, vol. 26, no. 1, Sep. 2021, pp. 2053–2059.
- [3] 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. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 11301–11307.
- [4] C. Yuan, X. Liu, X. Hong, and F. Zhang, "Pixel-level extrinsic self calibration of high resolution LiDAR and camera in targetless environments," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 7517–7524, Oct. 2021.
- [5] 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.
- [6] Z. Lin et al., "SGCalib: A two-stage camera-LiDAR calibration method using semantic information and geometric features," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2024, pp. 14527–14533.
- [7] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, "LiDAR-camera calibration using 3D-3D point correspondences," 2017, *arXiv:1705.09785*.
- [8] 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. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 5562–5569.
- [9] Y. Park, S. Yun, C. Won, K. Cho, K. Um, and S. Sim, "Calibration between color camera and 3D LiDAR instruments with a polygonal planar board," *Sensors*, vol. 14, no. 3, pp. 5333–5353, Mar. 2014.
- [10] J. Beltrán, C. Guindel, A. de la Escalera, and F. García, "Automatic extrinsic calibration method for LiDAR and camera sensor setups," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17677–17689, Oct. 2022.
- [11] C. Zheng and F. Zhang, "FAST-Calib: LiDAR-camera extrinsic calibration in one second," 2025, *arXiv:2507.17210*.
- [12] T. Tóth, Z. Pusztai, and L. Hajder, "Automatic LiDAR-camera calibration of extrinsic parameters using a spherical target," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 8580–8586.
- [13] J. Kümmerle, T. Kühner, and M. Lauer, "Automatic calibration of multiple cameras and depth sensors with a spherical target," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1–8.
- [14] M. Pereira, D. Silva, V. Santos, and P. Dias, "Self calibration of multiple LiDARs and cameras on autonomous vehicles," *Robot. Auto. Syst.*, vol. 83, pp. 326–337, Sep. 2016.
- [15] X. Gong, Y. Lin, and J. Liu, "3D LiDAR-camera extrinsic calibration using an arbitrary trihedron," *Sensors*, vol. 13, no. 2, pp. 1902–1918, Feb. 2013. [Online]. Available: <https://www.mdpi.com/1424-8220/13/2/1902>
- [16] Y. Choi, J. H. Park, and H.-Y. Jung, "Sample pose augmentation and adaptive weight-based refinement for 3-D LiDAR-camera extrinsic calibration using an orthogonal trihedron," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–14, 2024.
- [17] X. Zhao et al., "How challenging is a challenge? CEMS: A challenge evaluation module for SLAM visual perception," *J. Intell. Robot. Syst.*, vol. 110, no. 1, p. 42, Mar. 2024.
- [18] Z. Lin, Z. Gao, B. M. Chen, J. Chen, and C. Li, "Accurate LiDAR-camera fused odometry and RGB-colored mapping," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2495–2502, Mar. 2024.
- [19] C. Wen et al., "Toward efficient 3-D colored mapping in GPS/GNSS-denied environments," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 1, pp. 147–151, Jan. 2020.
- [20] J.-Y. Han, N.-H. Perng, and H.-J. Chen, "LiDAR point cloud registration by image detection technique," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 4, pp. 746–750, Jul. 2013.
- [21] N. Liu, X. Zhang, L. Wang, L. Xu, and S. Zhu, "3-D point cloud object completion via RGB images with complex geometric topology in urban scenes," *IEEE Geosci. Remote Sens. Lett.*, vol. 22, pp. 1–5, 2025.
- [22] R. Ishikawa, T. Oishi, and K. Ikeuchi, "LiDAR and camera calibration using motions estimated by sensor fusion odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 7342–7349.
- [23] D. Zermas, I. Izzat, and N. Papanikolopoulos, "Fast segmentation of 3D point clouds: A paradigm on LiDAR data for autonomous vehicle applications," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 5067–5073.
- [24] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vols. PAMI-8, no. 6, pp. 679–698, Nov. 1986.
- [25] L. Ferraz, X. Binefa, and F. Moreno-Noguer, "Very fast solution to the PnP problem with algebraic outlier rejection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 501–508.
- [26] L. Kneip, H. Li, and Y. Seo, "Upnp: An optimal o(n) solution to the absolute pose problem with universal applicability," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 127–142.
- [27] C. Zheng et al., "Fast-livo2: Fast, direct LiDAR-inertial-visual odometry," *IEEE Trans. Robot.*, vol. 41, pp. 326–346, 2025.



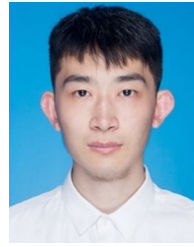
**Zhiyu Zhou** (Student Member, IEEE) received the B.E. degree from Wuhan University, Wuhan, China, in 2024, where he is currently pursuing the Ph.D. degree with the School of Remote Sensing and Information Engineering.

His research interests include robotics, multi-sensor fusion, SLAM, computer vision, and their applications.



**Zhi Gao** (Member, IEEE) received the B.Eng. and Ph.D. degrees from Wuhan University, Wuhan, China, in 2002 and 2007, respectively.

In 2008, he joined the Interactive and Digital Media Institute, National University of Singapore (NUS), Singapore, as a Research Fellow (A) and the Project Manager. In 2014, he joined the Temasek Laboratories, NUS (TL@NUS), as a Research Scientist (A) and a Principal Investigator. He is currently a Full Professor with the School of Remote Sensing and Information Engineering, Wuhan University. He has published more than 100 research papers in top journals and conferences, such as IJCV, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, CVPR, ECCV, ACCV, and BMVC. Since 2019, he has been supported by the Distinguished Professor Program of Hubei Province and the National Young Talent Program, China. His research interests include computer vision, machine learning, and remote sensing, and their applications. In particular, he has strong interests in vision for intelligent systems and vision-based intelligent systems.



**Zhipeng Lin** received the B.S. degree in measurement and control technology and instrumentation and the M.S. degree in instrument science and technology from Wuhan University, Wuhan, China, in 2017 and 2020, respectively. He is currently pursuing the Ph.D. degree in automation with the Department of Mechanical and Automation Engineering, Faculty of Engineering, The Chinese University of Hong Kong, Hong Kong.

His research interests include computer vision, deep learning, point cloud processing, and SLAM.



**Min Cao** received the bachelor's degree from Wuhan University, Wuhan, China, in 2000.

He is currently a Professor with Hubei University of Technology, Wuhan. He is the CEO of Wuhan Guanggu Zoyon Science and Technology Company Ltd., Wuhan. He is a Senior Engineer of photogrammetry and remote sensing. His research interests include electric power automation, precision engineering survey, highway inspection, and intelligent transportation systems.



**Wenbin Yang** received the B.E. degree from Wuhan University, Wuhan, China, in 2024, where he is currently pursuing the master's degree with the School of Remote Sensing and Information Engineering.

His research interests include mobile robots, path planning, and their applications.



**Jingshi Wang** is currently pursuing the Ph.D. degree with Shanghai Jiao Tong University, Shanghai, China.

He has authored or co-authored more than 15 papers in refereed journals and conferences and holds more than five granted patents. His research interests include sensor information processing and multisource information fusion.

Mr. Wang has received two provincial/ministerial-level awards for his research contributions.



**Xiaonan Wang** received the Ph.D. degree from the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China, in 2014.

His research interests are in close-range photogrammetry and computer vision, and his main research topics are data acquisition and data processing of handheld 3-D scanners.



**Yong Li** (Member, IEEE) received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 2019.

From 2019 to 2021, he was a Post-Doctoral Fellow with the Department of Computer Science, City University of Hong Kong, Hong Kong. From 2014 to 2016, he was a Visiting Scholar with the Department of Biomedical Informatics, University of California, San Diego, San Diego, CA, USA. He is currently an Engineer with Shanghai Key Laboratory of Aerospace Intelligent Control Technology, Shanghai, and Shanghai Aerospace Control Technology Institute, Shanghai. His research interests include compressive sensing, sparse representation, and image/signal processing.



**Qiyuan Zhang** received the Ph.D. degree from the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China, in 2024.

He is currently at ZG Technology Company Ltd., Wuhan, a subsidiary of Hexagon. His research interests include multisensor fusion, SLAM, mobile mapping, and 3-D reconstruction.