IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

I2D-LocX: An Efficient, Precise and Robust Method for Camera Localization in LiDAR Maps

Huai Yu, *Member, IEEE*, Xubo Zhu, Shu Han, Wen Yang, *Senior Member, IEEE*, and Gui-Song Xia, *Senior Member, IEEE*

***Abstract*—Camera localization within LiDAR maps has gained significant attention due to its potential for accurate positioning with low-cost and lightweight sensors compared to LiDAR-based systems. However, existing methods often prioritize localization accuracy, sometimes compromising efficiency, which can limit their suitability for real-time applications. To address these issues, we propose I2D-LocX, a lightweight monocular camera localization framework with three branches, establishing pixel-level and feature-level constraints to enhance localization performance without increasing model complexity. Specifically, the main branch generates a flow map to represent pixel-point displacements. One auxiliary branch shares the same input as the main branch and employs an additional decoder to evaluate the confidence of the flow map. The other auxiliary branch leverages a zero-flow generated from the displacement-free input to guide feature matching, thereby enhancing localization robustness. Notably, both auxiliary branches share parameters with the main branch and are omitted during inference, ensuring computational efficiency. Extensive experiments on benchmark datasets, including KITTI-Odometry, Argoverse, Waymo, and nuScenes, show that I2D-LocX can achieve centimeter-level localization accuracy with about 37 ms inference time, greatly improving the localization performance for real-world applications.**

***Index Terms*—Camera localization, LiDAR maps, image-to-point cloud registration.**

I. Introduction

RECENTLY, localizing monocular cameras within Light Detection And Ranging (LiDAR) point cloud maps has garnered significant attention in fields such as autonomous driving and robotic navigation [1], [2]. This surge of interest is largely attributed to advancements in hardware and the maturation of mapping algorithms. LiDAR-based mapping and localization [3], [4] enable high-precision positioning through point clouds-to-point clouds registration. However, these methods rely on expensive and heavy LiDAR sensors, limiting their applications to cost-efficient and payload-limited platforms. Conversely, camera-based Simultaneous Localization And Mapping (SLAM) [5] are more cost-effective but are susceptible to variations in lighting, weather, and environmental conditions. By leveraging LiDAR for mapping and subsequently using cameras for localization, it becomes possible to achieve robust and high-precision 3D maps that are insensitive to illumination changes, while offering a low-cost solution for mobile robot localization. Nevertheless, the significant challenges of cross-modal matching between LiDAR and camera data severely constrain the effectiveness of this approach, often resulting in poor localization accuracy and robustness.

Traditional approaches to cross-modal registration-based localization primarily focus on extracting shared feature descriptors from 2D and 3D spaces [6], [7]. Alternatively, some methods rely on matching through same-modal transformations [8], [9]. However, these methods often exhibit limited generalizability and robustness. In contrast, the advent of deep learning has introduced more sophisticated solutions, including networks that directly regress camera poses [2], [10], models that compute 2D–3D correspondences via neural networks followed by pose estimation using Perspective-n-Point (PnP) [11], [12], [13], and frameworks that project depth maps and leverage cross-modal flow estimation networks to infer matching relationships [1], [14], [15]. Despite their impressive performance, these methods frequently neglect the critical need for computational efficiency in mobile robot applications, hindering their practical deployment. The challenge is to design lightweight models that balance high precision with strong generalization capabilities.

We observe that the design of lightweight networks often suffers from performance degradation due to insufficient constraints, particularly during the cross-modal feature learning phase, which results in decreased registration performance. Moreover, considering the sparsity of LiDAR point clouds, previous approaches [1] have explored densification strategies. However, these methods introduce noise, adversely impacting the regression performance of lightweight models. To address these challenges, our insight is to enhance the cross-modal feature learning during the training while improving efficiency during inference. Since we have GT correspondence between the LiDAR point cloud and the RGB image, features can be directly learned through zero-displacement matching. Furthermore, we assess the confidence of each output pixel to mitigate the constraint insufficiency caused by sparsity, in contrast to previous methods that treat all outputs uniformly without differentiation.

In this letter, we propose I2D-LocX, a novel lightweight monocular camera localization framework in LiDAR maps.

Received 5 March 2025; accepted 8 June 2025. Date of publication 19 June 2025; date of current version 25 June 2025. This article was recommended for publication by Associate Editor G. Caron and Editor P. Vasseur upon evaluation of the reviewers' comments. The work was supported in part by the National Natural Science Foundation of China under Grant 62301370, in part by the Natural Science Foundation of Hubei Province, China under Grant 2025AFB640 and in part by the Natural Science Foundation of Wuhan under Grant 2024040801020233. *(Corresponding author: Huai Yu.)*

Huai Yu, Xubo Zhu, and Wen Yang are with the School of Electronic Information, Wuhan University, Wuhan 430072, China (e-mail: yuhuai@whu.edu.cn; zhuxubo@whu.edu.cn; yangwen@whu.edu.cn).

Shu Han is with the School of Computer Science, Wuhan University, Wuhan 430072, China (e-mail: hanshu@whu.edu.cn).

Gui-Song Xia is with the School of Artificial Intelligence, Wuhan University, Wuhan 430072, China (e-mail: guisong.xia@whu.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2025.3581122, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3581122

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

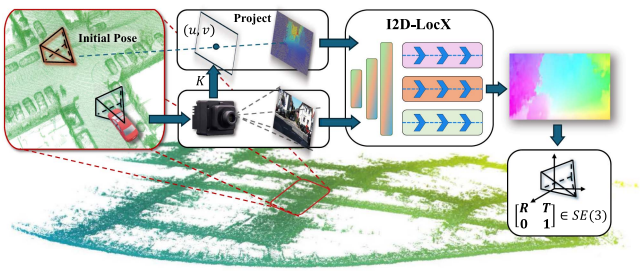

Fig. 1. The pipeline of I2D-LocX. Given the global LiDAR maps and an initial pose guess of a camera image, I2D-LocX can efficiently obtain 2D pixel - 3D point matches in flow representation. We then conduct a PnP solver on the 2D–3D correspondences to obtain the precise camera pose.

By leveraging a three-branch constraint design, our framework achieves high-precision pose estimation. As shown in Fig. 1, the proposed network transforms the camera localization problem into a cross-modal registration task, with pose estimation achieved via RANdom SAmple Consensus (RANSAC) based PnP. Specifically, we project LiDAR point clouds onto the camera's pixel coordinate system using a rough initial pose to generate depth maps. A lightweight flow estimation network is then employed to solve the pixel-to-point registration. To enhance accuracy and generalization, we introduce two auxiliary branches and design a cross-modal three-branch loss to provide additional constraints for solving the 2D–3D registration problem. Our main contributions are summarized as follows:

- We propose a novel lightweight camera localization method within LiDAR maps, which predicts cross-modal correspondences through a three-branch framework. Through reasonable framework design, the model achieves centimeter-level accuracy with a runtime of only 37 ms during inference.
- We introduce the Zero-Flow Feature Cost Branch to address the challenge of LiDAR sparsity. This branch uses zero-flow derived from depth maps and visual images under the true pose as guidance to optimize the feature extractors. Additionally, we design a specialized loss function that guides cross-modal feature extraction and matching through zero-flow.
- We propose the 2D–3D Pixel Cost Branch to address the performance trade-offs of lightweight models. This branch evaluates the confidence of outputs and incorporates a pixel-point loss, effectively constraining the matching outputs for improved accuracy.
- We validate the proposed I2D-LocX through extensive experiments on four public datasets (i.e., KITTI-Odometry, Argoverse, Waymo, and nuScenes). The results demonstrate that our method achieves accurate and robust monocular camera localization in LiDAR maps, showcasing strong generalization capabilities.

II. Related Work

With the rapid advancements in hardware capabilities and the continuous evolution of algorithmic techniques, deep learning-based approaches have emerged as the mainstream solution for monocular camera localization within prior LiDAR point cloud maps. These methods have demonstrated significant advantages over traditional techniques, achieving both higher accuracy and robustness. This section reviews three paradigms grounded in deep learning: localization via direct pose estimation, localization via cross-modal registration and localization via same-modal registration.

A. Localization Via Direct Pose Estimation

Cattaneo et al. [10], inspired by RegNet [16], present CMRNet, which is the first to address cross-modal localization using an end-to-end deep network. However, the embedding of camera parameters within the model reduces its generalization capability. Li et al. [17] reformulate the localization problem as a binary classification and back-projection optimization task. They determine whether 3D points lie within the camera's view frustum and then solve for the pose that satisfies the frustum constraints using an optimization solver. Jeon et al. [18] propose EFGHNet, which adopts a two-phase divide-and-conquer strategy to reduce relative pose discrepancies by decomposing the localization problem into separate tasks of rotation and translation estimation. However, the lack of further pose refinement limits its estimation accuracy. To address this, Wang et al. [2] introduce a coarse-to-fine 2D–3D registration architecture featuring a 2D–3D cost volume module, which implicitly constructs soft point-to-pixel correspondences to improve localization accuracy. While these methods enable efficient pose estimation via end-to-end learning, they still rely on the inherent hardware parameters, which constrain their adaptability.

B. Localization Via Cross-Modal Registration

Feng et al. [11] propose 2D3D-Match, one of the earliest localization methods addressing cross-modal registration, to avoid embedding camera intrinsic parameters. This approach estimates pose by solving the 2D–3D correspondences using EPnP. Ren et al. [12] introduce CorrI2P, which detects overlapping region features between images and point clouds, and leverages dense symmetric correspondences, followed by pose optimization using EPnP with RANSAC. Yao et al. [13] propose a quantity-aware strategy to address potential resolution gaps between 2D and 3D data, designing a coarse-to-fine matching network, CFI2P, based on this concept. However, despite decoupling the camera from its intrinsic parameters, these designs still rely on the specific hardware parameters of LiDAR sensors.

C. Localization Via Same-Modal Registration

Considering that registration-based methods inevitably rely on LiDAR hardware parameters, Cattaneo et al. [14] improve CMRNet by adopting a same-modal registration approach, followed by pose estimation using PnP. This method projects LiDAR point clouds into depth images, which not only reduces the modality gap but also mitigates the reliance on specific LiDAR hardware parameters. Subsequently, Chang et al. [19] propose Hypermap, which leverages feature maps to enhance the efficiency of map projection and feature extraction. This method effectively reduces the storage requirements of the local map while maintaining high accuracy. In contrast, I2D-Loc [1] and CMRNext [15] choose variants based on RAFT for same-modal registration, achieving more accurate localization results. However, due to the high computational complexity, they cannot meet the real-time localization requirements for in-vehicle applications. In addition, MDPCalib [20] utilize sensor motion estimates from visual and LiDAR odometry, in conjunction with 2D-to-3D correspondences derived from CMRNext [15], to

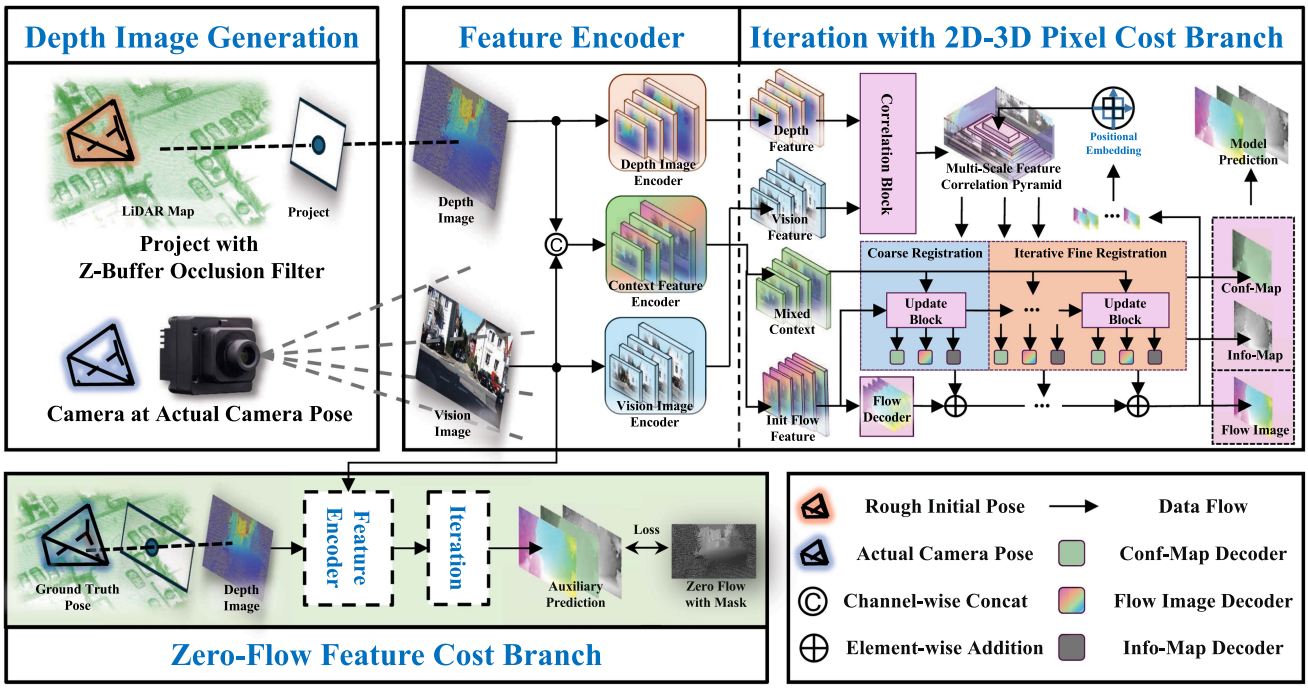

Fig. 2. The three branches of I2D-LocX: The main branch outputs the pixel displacement between the vision and depth images. The 2D–3D Pixel Cost Branch shares the same input as the main branch and generates the Confidence Map of the Flow Image. The Zero-Flow Feature Cost Branch takes the depth map of the ground-truth pose as input and applies the zero-flow constraint.

enable target-less and fully automatic camera–LiDAR extrinsic calibration [21].

Inspired by this need, we propose I2D-LocX, a lightweight three-branch camera localization method that establishes point-to-pixel correspondences between LiDAR depth and image data. Unlike previous methods, we do not increase the complexity of the network. Instead, we introduce auxiliary branches that share parameters with the main branch, creating additional constraints to achieve high-precision localization with low computational complexity.

III. Methodology

This section describes the workflow of our method, I2D-LocX, as shown in Fig. 2. We start by formally defining the problem and then proceed to describe the process of input data generation. Next, we present the network architecture, which consists of a three-branch structure based on an iterative flow prediction framework. Finally, we define the localization loss for the three-branch structure.

A. Problem Formulation

Common camera localization methods in LiDAR maps formulate the localization problem as taking a monocular camera image $\boldsymbol{I}_{vis}$ with a init pose $\boldsymbol{T}^{init} \in SE(3)$, a LiDAR map $\boldsymbol{P}_w$ in the world coordinate system and the camera intrinsic parameters $\boldsymbol{K}$ as inputs, followed by a camera pose estimation network $\theta_{\boldsymbol{K}}$ to refine the camera pose $\boldsymbol{T}_{cam}$. This network is trained to map a fixed pose estimation function $\mathcal{F}_{\boldsymbol{I},\boldsymbol{P}\to\boldsymbol{T}}$ corresponding to the localization task. In simple terms, it can be stated as follows:

$$\boldsymbol{T}_{cam}^{pred} = \mathcal{F}_{\boldsymbol{I},\boldsymbol{P}\to\boldsymbol{T}}\left(\boldsymbol{I}_{vis};\boldsymbol{P}_w;\boldsymbol{T}^{init};\boldsymbol{K};\theta_{\boldsymbol{K}}\right). \tag{1}$$

However, the embedding of the camera intrinsic parameters $\boldsymbol{K}$ results in tight coupling between the network $\theta_{\boldsymbol{K}}$ and the camera tensor. To decouple the camera intrinsic parameters $\boldsymbol{K}$ from the network $\theta_{\boldsymbol{K}}$, a novel network $\theta_{\mathcal{C}}$ is proposed to estimate the registration $\mathcal{C}_{\boldsymbol{I}-\boldsymbol{P}}$ between camera and LiDAR data. In contrast to previous approaches, this network requires only visual images $\boldsymbol{I}_{vis}$ and LiDAR data $\boldsymbol{P}^{init}$ at the camera's initial pose $\boldsymbol{T}^{init}$ as input. Therefore, this task is rewritten as:

$$\mathcal{C}_{\boldsymbol{I}-\boldsymbol{P}} = \mathcal{F}_{\boldsymbol{I},\boldsymbol{P}\to\mathcal{C}}\left(\boldsymbol{I};\boldsymbol{P}^{init};\theta_{\mathcal{C}}\right), \tag{2}$$

$$\boldsymbol{T}_{cam}^{pred} = \mathcal{F}_{\mathcal{C}\to\boldsymbol{T}}\left(\mathcal{C}_{\boldsymbol{I}-\boldsymbol{P}};\boldsymbol{P}^{init};\boldsymbol{K}\right), \tag{3}$$

where $\mathcal{F}_{\boldsymbol{I},\boldsymbol{P}\to\mathcal{C}}$ represents 2D–3D registration function for camera and LiDAR data, while $\mathcal{F}_{\mathcal{C}\to\boldsymbol{T}}$ refers to pose estimation performed using PnP with RANSAC. However, this approach also embeds the hardware parameters of the LiDAR into the network $\theta_{\mathcal{C}}$, making it difficult to generalize to other types of LiDAR. Therefore, a flow-based network $\theta_{\boldsymbol{f}}$ is proposed to transform the 2D–3D registration into a 2D–2D registration problem, thereby decoupling the hardware parameters of the LiDAR. Specifically, it first projects the LiDAR data into the pixel coordinate system to generate a depth map $\boldsymbol{D}^{init}$. Then, a flow network is used to fit pixel displacements $\boldsymbol{f}$, thereby establishing correspondences between the camera pixels and the LiDAR points, which are then used for pose estimation. Hence, the task is re-formulated as:

$$\boldsymbol{I}_{depth}^{init} = \mathcal{F}_{\boldsymbol{P}\to\boldsymbol{D}}\left(\boldsymbol{P}_w;\boldsymbol{T}^{init};\boldsymbol{K}\right), \tag{4}$$

$$\boldsymbol{f}^{pred} = \mathcal{F}_{\boldsymbol{I},\boldsymbol{D}\to\boldsymbol{f}}\left(\boldsymbol{I}_{vis};\boldsymbol{D}^{init};\theta_f\right), \tag{5}$$

$$\boldsymbol{T}_{cam}^{pred} = \mathcal{F}_{\boldsymbol{f}\to\boldsymbol{T}}\left(\boldsymbol{f}^{pred};\boldsymbol{D}^{init};\boldsymbol{K}\right). \tag{6}$$

Here, $\mathcal{F}_{\boldsymbol{P}\to\boldsymbol{D}}$ denotes depth image generation from 3D points. $\mathcal{F}_{\boldsymbol{I},\boldsymbol{D}\to\boldsymbol{f}}$ computes the flow field from visual image and depth

image inputs. $\mathcal{F}_{\boldsymbol{f}\to\boldsymbol{T}}$ estimates the camera pose from the flow field using PnP with RANSAC, similarly to $\mathcal{F}_{\mathcal{C}\to\boldsymbol{T}}$. The cross-modal registration function $\mathcal{F}_{\boldsymbol{I},\boldsymbol{D}\to\boldsymbol{f}}$, despite being extensively studied in prior research [1], [14], [15], achieving an optimal balance between efficiency and effectiveness remains a persistent challenge.

B. Depth Map Generation

To enable flow-based cross-modal registration method $\theta_{\boldsymbol{f}}$, we need to efficiently generate LiDAR depth maps and corresponding ground truth flow images. First, the LiDAR points $\boldsymbol{P}_w$ are transformed from the world coordinate system to the pixel coordinate system based on the initial pose $\boldsymbol{T}^{init}$ and camera intrinsic parameters $\boldsymbol{K}$. The transformed point set is then converted into a depth image based on the pixel coordinates. Following the approach in [10], we apply a projection method to generate an unoccluded depth map using a z-buffer occlusion filter, which first retains the nearest points along each line of sight and then removes remaining occluded points based on their angular relationships with neighboring points, resulting in more consistent and accurate depth maps. The process can be summarized as follows:

$$\boldsymbol{P}^{init} = \boldsymbol{T}^{init}\boldsymbol{P}_w, \tag{7}$$

$$\begin{pmatrix} u^{init} \\ v^{init} \\ 1 \end{pmatrix} = \frac{1}{Z^{init}} \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X^{init} \\ Y^{init} \\ Z^{init} \end{pmatrix} \triangleq \frac{\boldsymbol{K}\boldsymbol{P}^{init}}{Z^{init}}, \tag{8}$$

$$\boldsymbol{D}^{init}\left(u^{init}, v^{init}\right) = Z^{init}. \tag{9}$$

Then, we utilize the point set indices derived from both the accurate pose and the initial pose. This allows us to establish precise coordinate correspondences in the pixel coordinate system, which serve as the ground truth flow data for training. This process can be formulated as:

$$\begin{pmatrix} u^{GT} \\ v^{GT} \\ 1 \end{pmatrix} = \frac{1}{Z^{GT}}\boldsymbol{K}\boldsymbol{P}^{GT}, \tag{10}$$

$$\boldsymbol{f}^{GT}\left(u^{init}, v^{init}\right) = \left[u^{GT} - u^{init}, v^{GT} - v^{init}\right]. \tag{11}$$

Therefore, due to the sparsity of LiDAR points, the generation of valid matching point pairs becomes even sparser.

C. Network Architecture

The three-branch 2D–3D registration network I2D-LocX is modified from the SEA-RAFT [22]. The main branch adheres to the standard flow-based cross-modal localization function $\mathcal{F}_{\boldsymbol{I},\boldsymbol{D}\to\boldsymbol{f}}$. Therefore, the following sections will focus on the hybrid flow feature extractor, the iterative framework from coarse to fine, and the newly proposed two branches: the Zero-Flow Feature Cost Branch and the 2D–3D Pixel Cost Branch.

1) Hybrid Context Feature Encoder: To effectively extract light-weight cross-modal features, we adopt a pseudo-siamese network as the feature encoder to obtain lower-resolution features. The feature encoder consists of three variants of ResNet34, each designed to extract depth features $\boldsymbol{F}_d$, vision features $\boldsymbol{F}_i$, and hybrid context features $\boldsymbol{F}_h$. The $\boldsymbol{F}_h$ can be further divided into context features $\boldsymbol{F}_c$ and initial flow features $\boldsymbol{F}_f^{init}$.

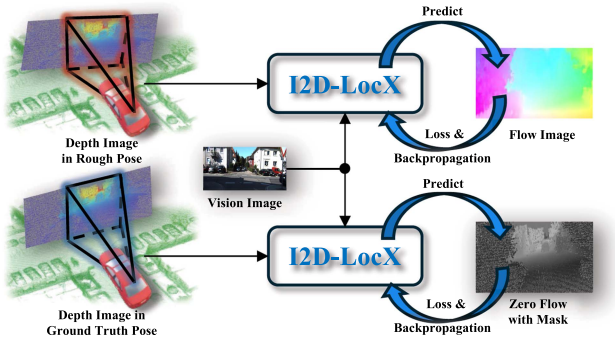

Fig. 3. Zero-Flow Feature Cost Branch.

Additionally, the input dimensions of the hybrid context encoder and depth image encoder are adjusted to 4 and 1, respectively, to accommodate vision-depth mixed information and the depth map inputs.

2) Coarse to Fine Iteration Registration: Inspired by RAFT [23], we first construct a multi-scale correlation pyramid from the the depth features $\boldsymbol{F}_d$ and vision features $\boldsymbol{F}_i$ by computing dot products between all pixel pairs at each scale. Multi-scale features are then sampled using position embedding-based indexing and fed into the update block, where they are combined with the hybrid context feature to obtain the depth-image flow. The update block is applied iteratively, with each step using the current flow to guide feature sampling, progressively refining the flow estimate through this feedback process.

3) Zero-Flow Feature Cost Branch: Due to the sparsity of LiDAR data, valid indices yield sparse coordinate displacements, resulting in weaker feature constraints from sparse masks. As show in Fig. 3, to enhance feature-level constraints, the ground truth pose is used to project and generate a depth map, where each valid pixel directly corresponds to a vision image. As a result, the coordinate displacement for each valid pixel is zero, representing zero-flow.

This branch adopts the same architecture as the main branch and shares all parameters with it. By leveraging dense correspondences generated from ground truth projections without adding any extra model parameters, it increases the data density and provides more effective feature-level supervision, thereby facilitating the learning of correct feature correspondences.

4) 2D–3D Pixel Cost Branch: The design of lightweight networks often compromises fitting performance due to the reduced model complexity, which in turn decreases the constraints on fitting. To address this, we introduce an additional branch that is used only during training to predict the confidence of the model's output and to incorporate this confidence into a weighted loss. Specifically, the branch produces a confidence map (conf-map) with the same resolution as the optical flow, which re-weights the contribution of each pixel pair to the loss. In parallel, it predicts an information weight map (info-map) that balances the influence between confidence-weighted and unweighted terms. The 2D–3D Pixel Cost Branch shares the same model architecture as the main branch, except for its decoder, which is modified to have an output channel dimension of 2+2, corresponding to the conf-map and info-map, respectively. During inference, both auxiliary branches are discarded seamlessly, as they do not participate in generating the correspondences.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

D. Loss Functions

1) Main Branch Loss: Following [1], We employ a Mask-EPE Loss to evaluate the optical flow estimation error. First, we compute a mask based on the valid point indices in the ground truth vision-depth flow $\boldsymbol{f}^{GT}$. Subsequently, the loss is calculated pixel-wise as the L1 norm within the masked region. This process can be succinctly described as follows:

$$mask(u,v,\boldsymbol{I},w)=\begin{cases}w(u,v), & \boldsymbol{I}(u,v)\neq 0,\\ 0, & otherwise.\end{cases} \tag{12}$$

$$\mathcal{L}_{epe}=\frac{\sum_{u,v} mask(u,v,\boldsymbol{f}^{GT},w)\left\|\boldsymbol{f}^{GT}(u,v)-\boldsymbol{f}^{pre}(u,v)\right\|_1}{\sum_{u,v} mask(u,v,\boldsymbol{f}^{GT},w)}. \tag{13}$$

Here,w represents the confidence weight for pixel displacements, with $w(u,v)=1$ in this case.

2) Auxiliary Branch Loss: In addition to the main branch loss, the two auxiliary branches introduce supplementary constraints at both the pixel-point and feature levels. The 2D–3D Pixel Cost Branch specifically evaluates the output's reliability using the conf-map and the info-map. Accordingly, the loss for this branch is defined as:

$$\mathcal{L}_{epe-\alpha\beta}=\alpha\mathcal{L}_{epe}+(1-\alpha)\,\mathcal{L}_{epe-\beta}. \tag{14}$$

In $L_{epe-\beta}$, $w(u,v)$ is set to $\beta(u,v)$, where the values of $\beta(u,v)$ are derived from the conf-map. Meanwhile, $\alpha(u,v)$ is a pixel-wise weighting factor derived from the info-map and serves to balance the contributions of $\mathcal{L}_{epe}$ and $\mathcal{L}_{epe-\beta}$.

Due to the sparsity of LiDAR data, the corresponding valid indices generate sparse coordinate displacements, leading to weaker constraints imposed by $\mathcal{L}_{epe-\alpha\beta}$ on the feature learning process because of sparse masks. To address this limitation, we propose the 2D–3D Feature Cost Branch to enhance the constraints provided by each iteration of data. This branch takes the LiDAR depth map projection $\boldsymbol{D}^{GT}$, generated from the ground truth pose as input, and computes the loss based on both the corresponding zero-flow $\boldsymbol{f}^{GT}_{zero}(u,v)=0$ and the predicted flow $\boldsymbol{f}^{pre}_{zero}(u,v)$. Specifically, it can be expressed as follows:

$$\mathcal{L}^{zero}_{epe-\alpha\beta}=\alpha^{zero}\mathcal{L}^{zero}_{epe}+(1-\alpha^{zero})\,\mathcal{L}^{zero}_{epe-\beta}. \tag{15}$$

3) Total Loss: Given the iterative structure of the network, we use $\gamma=0.8$ to represent the exponentially increasing weights as the number of iterations increases. Additionally, $\lambda=0.7$ is introduced to balance the contributions of the main branch and the Zero-Flow Feature Cost Branch, preventing the zero-flow constraint from causing the network parameters to approach zero. Consequently, the total loss is defined as:

$$\mathcal{L}_{total}=\sum_{i=1}^{N}\gamma^{N-i}\left(\lambda\mathcal{L}^{i}_{epe-\alpha\beta}+(1-\lambda)\,\mathcal{L}^{i-zero}_{epe-\alpha\beta}\right). \tag{16}$$

IV. Experiments

This section presents the dataset, evaluation metrics, and experimental setup. We also conduct ablation studies and compare with state-of-the-art approaches to demonstrate the effectiveness of the proposed I2D-LocX.

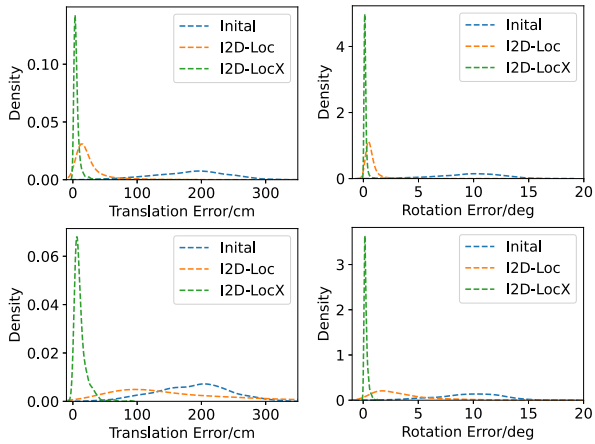

Fig. 4. Error distributions on KITTI and Argoverse. The top two figures are translation and rotation error on KITTI, and the bottom two are on Argoverse.

A. Experimental Settings

1) Datasets: To validate the generalization and robustness of the proposed method, we conducted extensive experiments on four real-world autonomous driving datasets: KITTI-Odometry [25], Argoverse [26], nuScenes [27], and Waymo [28]. These datasets span diverse cities, countries, sensor configurations, and imaging conditions. Fig. 5 illustrates the imaging conditions under different scenarios and portions of the LiDAR maps.

For KITTI-Odometry, we use the sequences 03, 05–09 as the training set and the sequence 00 (which contains 4,541 frames) as the validation set. Argoverse consists of two subsets: Argoverse 3D Tracking and Argoverse Perception. We use the sequences in train4 (which contains 5 sequences) from Argoverse Perception as the validation set, while the remaining sequences in train01–03 serve as the training set. To evaluate generalization, we use the entire nuScenes test set, which consists of 150 sequences, and the complete Waymo validation set, which contains 150 sequences, for evaluation without any training on these datasets.

The image sizes and LiDAR hardware configurations vary across these datasets. To standardize the experimental setup, all images are first resampled to an appropriate size, with corresponding adjustments to the camera intrinsic parameters. The images are then cropped to a resolution of 960 × 320. LiDAR data is projected using the extrinsic parameters and adjusted intrinsic parameters based on the projection formula.

2) Evaluation Metric: In line with I2D-loc [1], we assess the performance of registration using the following three metrics: Relative Translation Error (RTE), measured in centimeters; Relative Rotation Error (RRE), quantified in degrees; and Registration Recall (RR), defined as the percentage of successful registrations in the test set. A registration is considered successful if the RTE is less than $\tau_{RTE}=4$ m and the RRE is less than $\tau_{RRE}=20^{\circ}$.

3) Implementation Details: The network architecture is adapted from SEA-RAFT [22]: we adopt the ResNet-34 backbone for feature extraction, retain the coarse-to-fine correlation and update modules, and integrate the two training-only auxiliary branches described in Section III. To increase data diversity, we apply random cropping as well as contrast, saturation, and brightness adjustments, together with horizontal flipping. For

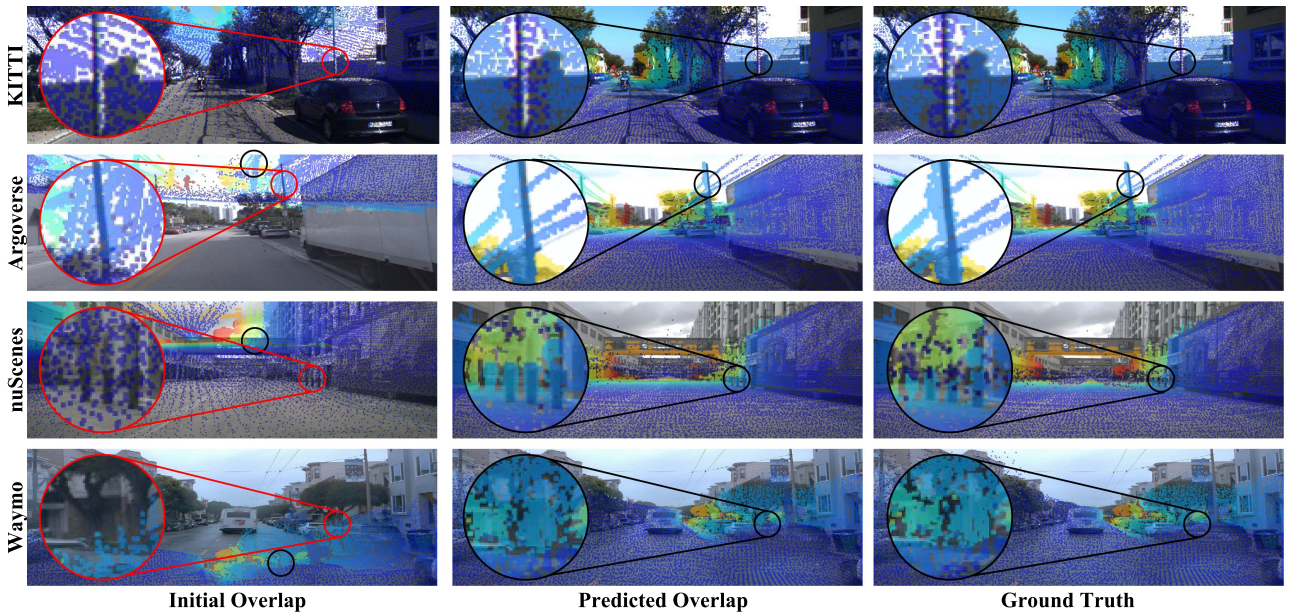

Fig. 5. Examples of the localization visualizations from KITTI, Argoverse, nuScenes, and Waymo datasets, respectively. The black circle marks the LiDAR projection overlap region in the initial, predicted, and ground truth views, while the red circle highlights the overlap in the initial pose.

TABLE I
COMPARISON OF METHODS ON KITTI AND ARGOVERSE DATASETS

Method	Time [ms] ↓	KITTI-Odometry			Argoverse		
		RTE [cm] ↓	RRE [°] ↓	RR [%] ↑	RTE [cm] ↓	RRE [°] ↓	RR [%] ↑
CMRNet [10]	**14.7**	46	1.60	—	145	2.32	—
CMRNet++ [14]	1300	55	1.46	97.82	80	1.55	93.76
HyperMap [19]	—	48	1.42	—	58	0.93	—
I2D-Loc [1]	326	18	0.70	98.39	47	0.71	92.48
CMRNext [15]	200	10	0.29	—	13	0.20	—
I2P-Net [2]	48	7	0.67	—	—	—	—
I2D-LocX	72	**4.145**	**0.133**	**99.80**	**7.861**	**0.174**	**100.00**

training, we set the learning rate to 1×10^{-4}, weight decay to 1×10^{-5}, and batch size to 4, with OneCycleLR as the learning rate scheduler. The AdamW optimizer is used to train the network for 100 epochs, with each sample undergoing 4 coarse-to-fine update iterations. The initial pose $\boldsymbol{T}^{init}$ is perturbed by adding uniformly distributed noise in the range of $[-2\,\text{m}, 2\,\text{m}]$ to the GT translation and $[-10°, 10°]$ to the GT rotation. All experiments are conducted on a single NVIDIA GTX 3090 GPU.

TABLE II
COMPARISON OF METHODS ON WAYMO AND NUSCENES DATASETS

Method	Waymo Open Dataset			nuScenes		
	RTE [cm] ↓	RRE [°] ↓	RR [%] ↑	RTE [cm] ↓	RRE [°] ↓	RR [%] ↑
CMRNet [10]	153.413	75.572	41.26	146.017	75.821	37.63
I2D-Loc [1]	59.642	4.323	61.11	40.056	2.546	73.47
I2D-LocX	**45.047**	**3.860**	**80.91**	**27.202**	**1.973**	**86.79**

B. Comparison Experiment

This section presents the experiments and analysis of our method on four datasets, along with comparisons to several state-of-the-art approaches.

1) Comparison of Camera Localization Performance: For comparison fairness, considering that several methods are not open-sourced, we first train our approach under the same dataset configurations before conducting comparisons. The quantitative results are reported in Table I, which presents evaluations on the validation sets of the trained datasets using three metrics. On the KITTI-Odometry and Argoverse datasets, our method achieves the best performance across all three metrics, with particularly significant improvements in RTE and RRE. As shown in Fig. 4, to further illustrate the effectiveness of our approach, we provide visualizations of the error distributions and output predictions, which show a clearer and more accurate alignment of our results compared to other methods. While I2P-Net [2] employs a cross-modal feature matching network for pose estimation and achieves competitive results, our method leverages same-modal inputs and incorporates pixel-level and feature-level distance constraints, leading to superior matching outcomes compared to other approaches.

To further validate the robustness of our approach, we also evaluate it on public datasets without prior training, including the Waymo Open Dataset and nuScenes, as presented in Table II. On the nuScenes dataset, our method demonstrates superior localization accuracy and a significantly higher success rate

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE III
COMPARISON OF TRACKING ON KITTI 00 SEQUENCE

Method	RTE [cm] ↓	RRE [°] ↓	RR [%] ↑
CMRNet [10]	163.58	126.02	—
I2D-Loc++ [24]	13	0.49	—
I2D-LocX	**3.035**	**0.092**	**100.00**

TABLE IV
ABLATION EXPERIMENT OF INDIVIDUAL COMPONENTS

Method	Time [ms] ↓	RTE [cm] ↓	RRE [°] ↓	RR [%] ↑
I2D-LocX w/o FO	241	**4.145**	**0.133**	**99.80**
I2D-LocX w/o HCFE	37	9.39	0.229	99.64
I2D-LocX w/o PCB	37	22.169	0.652	99.14
I2D-LocX w/o FCB	37	14.437	0.455	97.40
I2D-LocX w/o FR	**26**	11.619	0.268	99.80
I2D-LocX	37	**4.145**	**0.133**	**99.80**

FO, HCFE, PCB, FCB, FR represent Framework Optimization, Hybrid Context Feature Encoder, 2D-3D Pixel Cost Branch, Zero-Flow Feature Cost Branch, and Fine Registration, respectively.

compared to other approaches, confirming its generalization capability in challenging, previously unseen urban environments. For the Waymo dataset, which includes scenes collected from non-standard urban scenarios, all methods exhibit a performance drop. Nevertheless, our method maintains an edge over I2D-Loc and CMRNet in localization accuracy, delivering consistently lower errors even in cases where full success is not achieved.

In addition to these quantitative results, we provide qualitative visualizations of the input and output for four datasets in Fig. 5. These datasets vary significantly in terms of lighting conditions, weather, and sensor parameters. Despite these differences, our method consistently achieves good performance across all scenarios. These results underscore the robustness and reliability of our method under diverse and challenging conditions.

2) Comparison of Tracking Performance: We further evaluate the tracking performance of our method on the KITTI dataset, as detailed in Table III. The experimental setup for tracking is consistent with that of I2D-Loc++ [24], where only the initial pose of the first frame is provided, and the initial pose of each subsequent frame is derived from the pose estimation results of the preceding frame. Our method achieves superior performance in the tracking task on the KITTI dataset, primarily due to its excellent localization accuracy. The reduced pose drift in the tracking scenario is attributed to the lower localization error in the preceding frame, resulting in a smaller initial pose error for the current frame compared to other methods. Consequently, our method demonstrates enhanced tracking performance on the KITTI dataset, effectively leveraging its accurate localization capabilities.

C. Ablation Experiment

To validate the effectiveness of the proposed modules in I2D-LocX, we conducted a series of ablation experiments on the KITTI dataset, with the results summarized in Table IV. To improve computational efficiency without compromising accuracy, we optimized the codebase by redesigning the data flow and computation logic of key modules and refactored the implementation using a higher-version framework. Experimental results demonstrate a significant performance boost, achieving approximately a fivefold acceleration compared to the previous version. This improvement not only reduces the time required for model training and inference but also enables real-time processing of real-world scenarios. To enhance the performance of the lightweight model, we focused on the correlation between cross-modal features. By skillfully converting 2D–3D matching into 2D–2D matching, we leveraged the Hybrid Context Feature Encoder to extract cross-modal hybrid flow features at the pixel scale, avoiding discrepancies in feature descriptions caused by modality differences and resulting in more precise feature matching. Additionally, to improve network fitting performance without increasing model parameters, we introduced two auxiliary branches that share parameters with the main branch and supervise both feature-level fitting distances and pixel-point level fitting accuracy. While this design marginally increases training time, it ensures precise feature encoding and decoding, minimizing pose estimation errors. Importantly, these auxiliary branches can be omitted during inference, ensuring no additional computational overhead. Lastly, although iterative coarse-to-fine fitting increases computation time, it significantly improves performance. As shown in Table V, experiments analyzing the impact of iteration count on localization accuracy reveal that localization performance peaks at iter=4, as the model was trained with this iteration setting. Furthermore, increasing the iteration count improves the localization success rate by enabling more effective feature querying from the multi-scale pyramid, progressively refining pose estimates, and correcting those with large initial offsets. To further validate the efficiency of the proposed method, we conducted runtime speed tests for all modules, with the results reported in Table VI. This table presents a runtime analysis of the system's various modules. With an input resolution of 960 × 320, the system performs at 14 fps. The experiments showed that our method achieves real-time performance on a standard GPU by optimizing the code and refactoring the framework. This reduction in runtime is particularly beneficial for deployment in resource-constrained environments where real-time processing is critical.

TABLE V
ABLATION EXPERIMENT OF ITERATION

Iters	Time [ms] ↓	RTE [cm] ↓	RRE [°] ↓	RR [%] ↑
1	**26**	11.619	0.268	99.80
4	37	**4.145**	**0.133**	99.80
8	47	4.294	0.135	99.91
12	58	4.366	0.139	**100.00**

TABLE VI
RUNTIME PERFORMANCES

	LiDAR Project	Model Infer	PnP+RANSAC	Total
Time [ms]	10	37	25	72(∼14Hz)

V. CONCLUSIONS AND FUTURE WORK

We present I2D-LocX, a lightweight monocular camera localization method for LiDAR maps. The framework uses a three-branch design for high-precision pose estimation, a flow estimation network for pixel-to-point registration, and auxiliary branches to enhance accuracy. The 2D–3D Pixel Cost Branch evaluates output confidence and constrains pixel-point correspondences, while the Zero-Flow Feature Cost Branch solves the

feature sparsity issue caused by LiDAR sparsity. Experiments on four datasets demonstrate that our proposed method improves monocular camera localization accuracy and robustness, achieving centimeter-level precision with a 37 ms inference time. In future work, we will apply I2D-LocX to large FoV fisheye or spherical image to address the limited overlap between the camera field of view and the LiDAR map.

REFERENCES

[1] K. Chen et al., "I2D-Loc: Camera localization via image to LiDAR depth flow," *ISPRS J. Photogramm. Remote Sens.*, vol. 194, pp. 209–221, 2022.

[2] G. Wang et al., "End-to-end 2D–3D registration between image and LiDAR point cloud for vehicle localization," 2023, *arXiv:2306.11346*.

[3] W. Xu and F. Zhang, "FAST-LIO: A fast robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

[4] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, "Faster-LIO: Lightweight tightly coupled LiDAR-inertial odometry using parallel sparse incremental voxels," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022.

[5] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[6] H. Yu, W. Zhen, W. Yang, and S. Scherer, "Line-based 2-D–3-D registration and camera localization in structured environments," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 11, pp. 8962–8972, Nov. 2020.

[7] E. Yudin et al., "Cloudvision: DNN-based visual localization of autonomous robots using prebuilt LiDAR point cloud," in *Proc. IEEE 97th Veh. Technol. Conf.*, 2023, pp. 1–6.

[8] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular camera localization in 3D LiDAR maps," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*., 2016, pp. 1926–1931.

[9] X. Zuo, P. Geneva, Y. Yang, W. Ye, Y. Liu, and G. Huang, "Visual-inertial localization with prior LiDAR map constraints," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 3394–3401, Oct. 2019.

[10] D. Cattaneo, M. Vaghi, A. L. Ballardini, S. Fontana, D. G. Sorrenti, and W. Burgard, "CMRNet: Camera to LiDAR-map registration," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 1283–1289.

[11] M. Feng, S. Hu, M. H. Ang, and G. H Lee, "2D3D-Matchnet: Learning to match keypoints across 2D image and 3D point cloud," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 4790–4796.

[12] S. Ren, Y. Zeng, J. Hou, and X. Chen, "CorrI2P: Deep image-to-point cloud registration via dense correspondence," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 3, pp. 1198–1208, Mar. 2023.

[13] G. Yao, Y. Xuan, Y. Chen, and Y. Pan, "Quantity-aware coarse-to-fine correspondence for image-to-point cloud registration," *IEEE Sens. J.*, vol. 24, no. 20, pp. 33826–33837, Oct. 2024.

[14] D. Cattaneo, D. G. Sorrenti, and A. Valada, "CMRNet: Map and camera agnostic monocular visual localization in LiDAR maps," in *Proc. IEEE Int. Conf. Robot. Automat. Workshop*, 2020, pp. 1–5.

[15] D. Cattaneo and A. Valada, "CMRNext: Camera to LiDAR matching in the wild for localization and extrinsic calibration," *IEEE Trans. Robot.*, vol. 41, pp. 1995–2013, 2025.

[16] N. Schneider, F. Piewak, C. Stiller, and U. Franke, "RegNet: Multimodal sensor registration using deep neural networks," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 1803–1810.

[17] J. Li and G. H. Lee, "DeepI2P: Image-to-point cloud registration via deep classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15960–15969.

[18] Y. Jeon and S.-W. Seo, "EFGHNet: A versatile image-to-point cloud registration network for extreme outdoor environment," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7511–7517, Jul. 2022.

[19] M.-F. Chang, J. Mangelson, M. Kaess, and S. Lucey, "HyperMap: Compressed 3D map for monocular camera registration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11739–11745.

[20] K. Petek, N. Vödisch, J. Meyer, D. Cattaneo, A. Valada, and W. Burgard, "Automatic target-less camera-LiDAR calibration from motion and deep point correspondences," *IEEE Robot. Automat. Lett.*, vol. 9, no. 11, pp. 9978–9985, Nov. 2024.

[21] S. Huang, C. Lin, and Y. Zhao, "What really matters for learning-based LiDAR-camera calibration," 2025, *arXiv:2501.16969*.

[22] Y. Wang, L. Lipson, and J. Deng, "SEA-RAFT: Simple, efficient, accurate RAFT for optical flow," in *Proc. Eur. Conf. Comput. Vis.*, 2025, pp. 36–54.

[23] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *Proc. Eur. Conf. Comput. Vis*., 2020, pp. 402–419.

[24] H. Yu, K. Chen, W. Yang, S. Scherer, and G.-S. Xia, "I2D-Loc++: Camera pose tracking in LiDAR maps with multi-view motion flows," *IEEE Robot. Automat. Lett.*, vol. 9, no. 9, pp. 8162–8169, Sep. 2024.

[25] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[26] M.-F. Chang et al., "Argoverse: 3D tracking and forecasting with rich maps," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8748–8757.

[27] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[28] P. Sun et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2446–2454.