IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 May 2022, accepted 9 June 2022, date of publication 29 June 2022, date of current version 8 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3186974

PSNet: LiDAR and Camera Registration Using Parallel Subnetworks

YI WU[1,2], MING ZHU[1], AND JI LIANG[1,2]
[1]Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China
[2]University of Chinese Academy of Sciences, Beijing 100049, China

Corresponding author: Ming Zhu (zhu_mingca@163.com)

ABSTRACT The working environment of autonomous driving and robot navigation is so complex and dynamic that a single type of sensor is insufficient for performing object detection. Thus, in many perception schemes, the LiDAR-camera fusion strategy is preferred. However, the performance of a LiDAR-camera fusion heavily relies on a set of accurately calibrated extrinsic parameters. We propose PSNet, an end-to-end convolutional neural network (CNN) for calibration; this is the first calibration network to use parallel subnetworks to obtain multiresolution features and fuse them adaptively to encourage robustness against different initial error ranges. The method has three key characteristics: (i) Addition of a downsampling block to improve suitability for sparse projected depth maps; (ii) Connection of the high-to-low resolution convolution streams in parallel to obtain multiresolution features that are spatially more precise and contain richer semantic information; (iii) Fusion of multiresolution streams by the multiscale feature aggregation module. The network corrects errors from initial calibration to the ground truth online, rather than directly obtaining the accurate parameters. We evaluated our model on the KITTI datasets and it outperformed other CNN-based methods. In addition, extensive experiments evaluating the model with untrained and unfamiliar datasets demonstrated that our method exhibited good generalization ability.

INDEX TERMS Calibration, deep learning, multiscale features, parallel subnetworks.

I. INTRODUCTION

Robust robotic environmental awareness is a central capability for autonomous driving and robot navigation, and all sensors have their own inherent advantages and disadvantages. For example, cameras easily suffer from underexposure and overexposure due to sudden changes in light, but they can provide high-resolution 2D information. In contrast, light detection and ranging (LiDAR) sensors suffer in rainy and foggy weather; however, they can yield sparse but highly accurate 3D measurements. Thus, in many object detection and tracking schemes, a LiDAR-camera fusion strategy is preferred. The calibration of a LiDAR-camera is the basis for the effective fusion of these two sensors.

Most existing calibration methods [1]–[8] are target-based, which can achieve satisfactory accuracy with laborious manual adjustments and complex environmental settings. However, when a vehicle is in operation, sensor vibrations and environmental changes can add uncontrollable deviations to the extrinsic parameters. Therefore, to improve adaptability, online calibration has a high value. Although there are a series of approaches to tackle the online calibration challenge, they always require extensive additional knowledge, such as additional motion information [9], [10] or an educated initial guess. Some researchers apply deep learning to solve calibration problems. However, these methods directly learn the low-resolution representations of the projected depth map and corresponding image without considering the correlation between the resolution and the initial error range. Moreover, the feature extraction module of existing methods [11]–[14] is a ResNet-18, which is sufficient for handling many computer vision tasks, but we want to learn richer representations from sparse depth maps using a stronger backbone.

In this paper, we present PSNet, which monitors and corrects calibration errors online to save human resources and improve adaptability. The contributions of this study are summarized as follows:

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Liu.

(1) PSNet is a novel end-to-end approach for correcting the transformation between LiDAR and camera, and it is the first calibration network that connects the multiresolution convolution streams in parallel. It benefits from generated representations that are spatially more precise and have richer semantic information, which can help improve performance. According to the experimental results, our model not only achieves state-of-the-art performance compared with other convolutional neural network (CNN)-based methods but also has good generalization with untrained and unfamiliar datasets.

(2) PSNet verifies the correlation between the resolution and the initial error range. We used a novel architecture, namely, the multiscale feature aggregation module, which fused the multiresolution features adaptively to encourage network robustness against different initial error ranges.

(3) Compared with 2D RGB images, projected depth maps are sparse. In other words, some pixel values in the depth map are equal to zero. Therefore, we modified the downsampling block to make it more suitable for sparse projected depth maps.

II. RELATED WORKS

Many robotic methods can calibrate a LiDAR-camera system. Generally, existing methods can be divided into three categories: target-based, targetless, and deep learning methods.

A. TARGET-BASED METHODS

In target-based methods, to establish accurate 2D-3D point correspondences, checkerboards [1]–[4] or certain targets that have a specific appearance, e.g., polygonal planar boards [5], planar targets that have circular holes [6], and spherical targets [7], [8], are used for calibration. Regardless of the nature of the targets, target-based approaches first extract geometrical features between both data representations, then establish the 2D–3D correspondences. The calibration problem then becomes a pose estimation problem. The accuracy of these methods is restricted by the number of points provided by the targets and the resolution of the LiDAR. As extra preparation is required, these algorithms are time-consuming, laborious, and limited to offline use.

B. TARGETLESS METHODS

During live robot operation, the relative positioning of the LiDAR and camera will inevitably drift due to environmental changes or vibrations. The perception systems are the eyes of the robots. Miscalibration between sensors will influence the fusion of the LiDAR and camera, causing the perception systems to provide an inaccurate and unstable perception of the surrounding environment. Therefore, there is an urgent need for online calibration methods, which could significantly improve the adaptability of these robots.

Yuan *et al.* [15] did not use checkerboards but aligned natural edge features between both data representations and achieved high accuracy. The authors used the Canny algorithm to extract 2D edges; for point cloud edge extraction, they used RANSAC to extract planes and solved for the intersection of plane pairs. Zhu *et al.* [16] first projected LiDAR points onto the image plane, and the pixel values were depth and reflectivity values. Then, the authors extracted the 2D edges from the depth map and matched these edges with those obtained from the corresponding image. Pandey *et al.* [17] used the mutual information of the camera image and point clouds intensity map for calibration. Taylor *et al.* [18] measured the gradient orientation and built the gradient relation between images and point clouds.

Furthermore, some methods [9], [10] use sensor motion to estimate the extrinsic parameters. These motion-based methods attempt to find the transformation that best aligns the motion tracks of the two sensors.

C. DEEP LEARNING METHODS

Due to the automatic feature engineering of deep neural networks, extensive research has recently been conducted to solve the calibration problem using deep neural networks. Current deep learning methods take the projected depth maps and corresponding RGB camera images as the input, and by designing a CNN, the extrinsic parameters can be obtained. Regnet [19] was the first deep CNN for LiDAR-camera calibration; it packaged the conventional calibration steps into a single CNN. CalibNet [11] introduced geometric supervision, which included reducing the photometric error and point cloud distance error. CMRNet [20] is an approach for locating a camera in a LiDAR-Map, and it is the first method to use the correlation layer of PWC-Net [21] to match features acquired from two sensors to achieve 6-DoF extrinsic calibration. CalibRCNN [12] used the constraint relationship between successive frames for calibration, which improved the accuracy, and CalibDNN [13] added geometric and transformation supervisions to solve the calibration problem and applied the method on a challenging dataset.

III. PROPOSED METHOD

We designed an end-to-end calibration model for a LiDAR-camera system that is subject to the assumption: we know the ground truth LiDAR-camera extrinsic parameters T_{gt} and the camera calibration matrix K. Our method consists of a data preprocessing process, a network for regressing the predicted vectors, loss functions, and a calibration inference. The workflow of our method is shown in Fig. 1.

A. DATA PREPROCESSING

In the proposed method, depth maps projected by LiDAR point clouds with an initial and miscalibrated setup and corresponding RGB images are input to the network. As in practical applications, the initial extrinsic parameters can be approximately obtained via measurements or estimation. However, during the training phase, to obtain a large training dataset, we combined the random transformation parameters ΔT and the ground truth extrinsic parameters T_{gt} to obtain

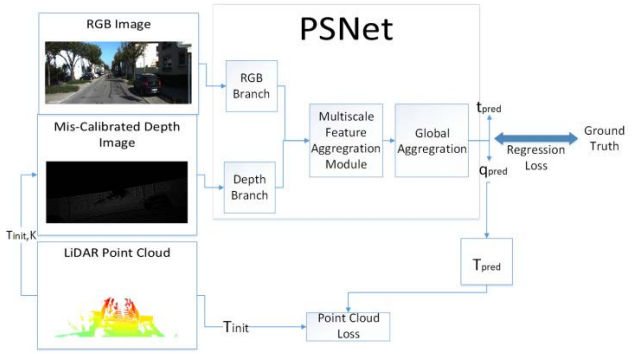

FIGURE 1. Method workflow. First, the proposed method projects a LiDAR point cloud onto the image plane with the initial extrinsic parameters T_{init} and intrinsic parameters K, and we obtain the miscalibrated depth map. Then, it takes the miscalibrated depth map and RGB image as input and outputs a 1 × 3 translation vector and a 1 × 4 rotation quaternion. Finally, we use the regression and point cloud losses to guide the learning process.

the initial parameters T_{init}.

$$T_{init} = \Delta T \times T_{gt} \tag{1}$$

To obtain the miscalibrated depth maps, first, we transformed each 3D LiDAR point $[x, y, z]$ in a point cloud from the LiDAR coordinate system to the camera coordinate system with the initial extrinsic calibration parameters T_{init}.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = T_{init} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{2}$$

where $[x', y', z', 1]$ and $[x, y, z, 1]$ represent the homogeneous coordinates of a point in the camera coordinate and LiDAR coordinate systems, respectively. It is worth noting that using the initial and miscalibrated parameters results in the homogeneous coordinate of a point being inconsistent with its actual location in the camera coordinate system. We refer to the points with incorrect coordinate values in the camera coordinate system as miscalibrated point clouds.

Each 3D point $[x', y', z']$ in a miscalibrated point cloud was projected onto the image plane $[u, v, z_c]$, named a miscalibrated depth image with intrinsic parameters K, where z_c is the pixel value and also the depth value of the camera coordinate. If there were no LiDAR points projected on the pixel, the pixel value was zero.

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \times \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} \tag{3}$$

B. NETWORK ARCHITECTURE

1) FEATURE EXTRACTION NETWORK

There were two separate branches of feature extraction from the RGB image and the depth map. Inspired by HRNet [22], our backbone network preserves high-resolution representations and connects multiple resolutions in parallel. As shown in Fig. 2, first, we encoded the input image as multiresolution representations. Then, we connected the multiresolution streams parallelly to exchange information between them.

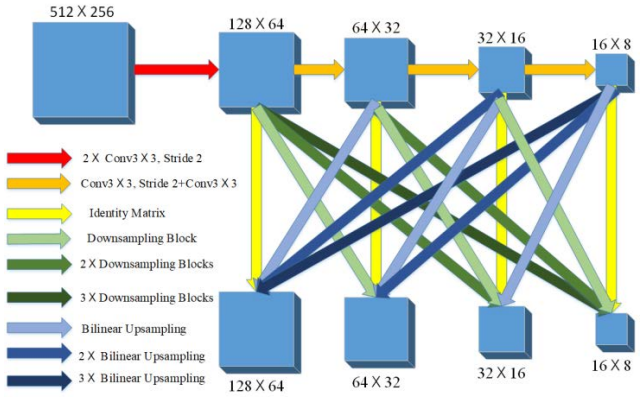

FIGURE 2. Feature extraction network.

We acquired the output representations, which are the sum of the transformed multiresolution representations, as follows:

$$f_r^o = F_{1r}(f_1^i) + F_{2r}(f_2^i) + F_{3r}(f_3^i) + F_{4r}(f_4^i) \quad r = 1, 2, 3, 4, \tag{4}$$

where F_{ir} is the transform function. If r < i, F_{ir} upsamples the input features through bilinear upsampling followed by a 1 × 1 convolution to align the number of channels. If r = i, F_{ir} is an identity matrix. Finally, if r > i, F_{ir} represents (r-i) downsampling blocks used to extract richer feature representations for each modality individually, especially for sparse depth images. The downsampling block not only reduces resolution but also increases the width, which is good for feature extraction. This design was modified from ParNet [23], and we added 2D max-pooling followed by a 1 × 1 convolution, which could retain more details for the depth images in which there were many pixels with a value of zero.

We found this block to increase performance (Table 5). The downsampling block structure is illustrated in Fig. 3. The proposed method has a key advantage, namely, the multiresolution representations extracted from the backbone are semantically stronger than in prior methods [11]–[13]. Furthermore, the high-resolution representations are spatially accurate.

2) MULTISCALE FEATURE AGGREGATION MODULE

During vehicle operation, due to various factors such as aging and loosening, or vibrations during aggressive driving, the miscalibration ranges of the extrinsic parameters are uncertain. There is a method [24] to study the correlation between resolution and object size. The authors propose a feature fusion module, which is essentially an attention module, to combine the advantages of different resolution features. Similarly, we study the correlation between resolution and miscalibration range. Intuitively, as for a small miscalibration

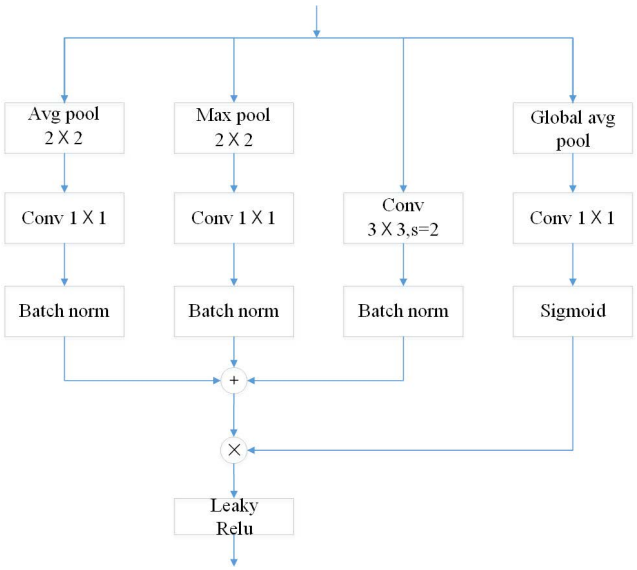

FIGURE 3. Downsampling block.

range, because details are preserved better in high-resolution features, a high-resolution model can build a more accurate correlation between images and depth maps. Therefore, a high-resolution model works better for a small miscalibration range, whereas a low-resolution model performs better for a large miscalibration range, as the low-resolution representations are semantically stronger. In Table 6, we experimentally verified this intuition. Thus, fusing the multiresolution features encouraged network robustness against miscalibration ranges.

In Fig. 4, this module consists of multiresolution representations acquired from the feature extraction network for each modality. Inspired by [24], to enable the network to adaptively control the flow of information from multiple branches with different feature scales, we proposed an automatic adjustment operation, namely, multiscale feature aggregation among multiple resolution branches.

First, we concatenated the feature maps along the channel dimension in each level to fuse the information from both modalities. Then we recalibrated the channel importance in each branch. Specifically, as Fig. 4 states, we downsampled the high-resolution concatenated features to the same resolution through stride-2 3 × 3 convolutions and fused features via an element-wise summation. Next, we performed global average pooling to obtain attention weights a. Further a is fed to a multi-layer perceptron (MLP) which consists of a fully connected (FC) layer with $\frac{2C}{r}$ output channels (r is the channel compression ratio), four parallel FC layers with $2C$ output channels, and Softmax function. Lastly, we multiplied the different scale features and their corresponding attention vectors to obtain the recalibrated features.

To incorporate multiscale information into the feature at a given single resolution, we borrowed the cross feature-level fusion block [24] to dynamically adjust the weights of different resolution feature branches.

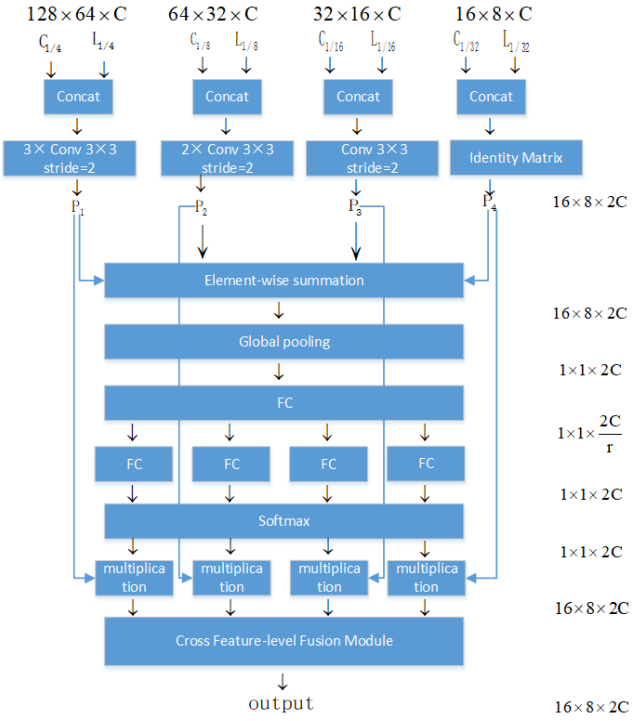

FIGURE 4. Multiscale feature aggregation module. $C_{1/4}$ and $L_{1/4}$ represent high-resolution features extracted from two separate branches of feature extraction network, respectively.

3) GLOBAL AGGREGATION

As the accurate calibration is built on a strong feature learning ability, that is, obtaining fused features from the multiscale feature aggregation module, we input them into DenseNet [25] for further feature learning. Thereafter, we used an FC layer and two branches with stacked FC layers to obtain a 1 × 3 translation vector and a 1 × 4 rotation quaternion.

C. LOSS FUNCTION

Regarding the loss function, we used LCCNet [14] as a reference. During training, we used two types of loss functions: regression loss L_T and point cloud distance loss L_p.

1) REGRESSION LOSS

After obtaining the predicted translation vector t_{pred}, we inspected the predicted translation vector t_{pred} and the translation part of random transformation parameters ΔT, that is, Δt. We used the smooth L1 loss to represent the translation loss. The translation loss function is as follows:

$$L_t = \frac{1}{3}\sum_{r\in(x,y,z)} Smooth\ L1(\Delta t_r - t_{pred\_r}), \tag{5}$$

where the smooth L1 loss is as follows:

$$SmoothL1 = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & x < -1\ or\ x > 1. \end{cases} \tag{6}$$

Regarding the rotation loss, as in [26], we set the angular distance as the difference between quaternions.

$$L_R = D_a(q_{gt} - q_{pred}) \tag{7}$$

The total regression loss combined the above two loss functions as defined below:

$$L_T = k_t L_t + k_q L_R. \tag{8}$$

2) POINT CLOUD DISTANCE LOSS

We used the same point cloud distance loss [14], as defined below:

$$L_p = \frac{1}{N}\sum_{i=1}^{N} \left\| T_{pred}^{-1} \times \Delta T \times P_i - P_i \right\|_2. \tag{9}$$

D. CALIBRATION INFERENCE

The network corrects errors from initial calibration T_{init} to the ground truth T_{gt} online, rather than directly obtaining the accurate parameters. Therefore, we combined the predicted parameters and the initial calibration parameters to obtain the calibrated parameters:

$$\hat{T}_{LC} = T_{pred}^{-1} \times T_{init}. \tag{10}$$

IV. EXPERIMENTS AND DISCUSSION

A. DATASET PREPARATION

We used different sequences in the KITTI dataset [27] that were acquired in various scenes with different calibration parameters. Some existing learning-based methods use different training and testing datasets. For a fair comparison, we used three different experimental setups as follows.

1) KITTI-ODOMETRY DATASET

The KITTI-Odometry dataset [28] consists primarily of the sequences "2011_09_30" and "2011_10_03", which are further divided into 21 smaller sequences (00-02,04-21). In addition, it includes a small sequence (03) which consists of 801 frames of the sequence "2011_09_26". Compared with the total size (43,552 frames) of the Odometry dataset, we can assume that the sequence "2011_09_26" is not in the KITTI-Odometry dataset. The images and point clouds in each small sequence were paired based on the timestamp synchronization of the camera and the LiDAR. We only used the left images and point clouds from the small sequence 00 as the testing dataset, and the small sequences from 01 to 21 as the training datasets. The KITTI dataset provided the extrinsic parameters in each sequence using the method of [1] and we applied them as the ground truth T_{gt}.

2) KITTI-RAW SEQUENCE "2011_09_26"

We used all drives (except 0005 and 0070) for training and the 0005 and 0070 drives as the testing set.

We added a random transformation ΔT to miscalibrate the point clouds. For the above two datasets, the deviation range of miscalibration was set to $\pm 10°$ rotation and ± 0.25 m translation of any axis.

3) PARTS OF THE KITTI-ODOMETRY DATASET

To compare our method with CMRNet [20], which is an end-to-end method with an iteration scheme, we used the same training and testing dataset setup. We set sequences 03-09 as the training dataset (total of 11,697 frames) and sequence 00 as the testing dataset (4,541 frames).

The maximum deviation range for the first iteration was [± 2 m, $\pm 10°$] and for the second and third iterations were [± 1 m, $\pm 2°$] and [± 0.6 m, $\pm 2°$], respectively.

As the image dimension in the KITTI dataset was different, we downsampled the original and depth images to 512×256 through bilinear interpolation.

B. EVALUATION METRICS

To analyze the experimental results and fairly compare them with other learning-based methods, we used [14] as a reference. For the translation, the absolute translation error was expressed as follows:

$$E_t = \left| t_{pred} - t_{gt} \right|. \tag{11}$$

We calculated the absolute translation error in the X-, Y-, and Z-directions as E_X, E_Y, and E_Z, respectively, and the mean value as $\bar{t}$.

For the rotation, we transformed the rotation matrix to Euler angles and computed the angle errors of roll, pitch, yaw, and the mean value $\bar{R}$.

In addition, we applied the localization error mentioned in [20]: the median and mean translation errors and standard deviation for the translation component and the corresponding rotation errors for the rotation component.

C. TRAINING DETAILS

We built the network with the Pytorch library. During training, we used the Adam optimizer and set the initial learning rate to $1e^{-4}$. In addition, we halved the learning rate after 20, 50, and 70 epochs. We trained the PSNet on an Nvidia RTX2080Ti GPU with batch size 16. For the KITTI-Odometry dataset and KITTI-raw sequence "2011_09_26," the training epoch of the model was set to 50. The training epoch of the model on parts of the KITTI-Odometry dataset was set to 100.

D. RESULTS AND DISCUSSION

We evaluated the calibration performance of PSNet on the testing dataset described in Section IV.A.

1) TESTING ON THE KITTI-RAW SEQUENCE "2011_09_26" AND KITTI-ODOMETRY DATASET

The calibration results are shown in Table 1. CalibNet, CalibDNN, and CalibRCNN were all set with an initial off-range of (± 0.25 m, $\pm 10°$). Dealing with the miscalibration range, CalibDNN and CalibRCNN were simple with a single model and single iteration, whereas CalibNet used two separate models for the rotation and translation calibration. Their experimental datasets were different; to compare the

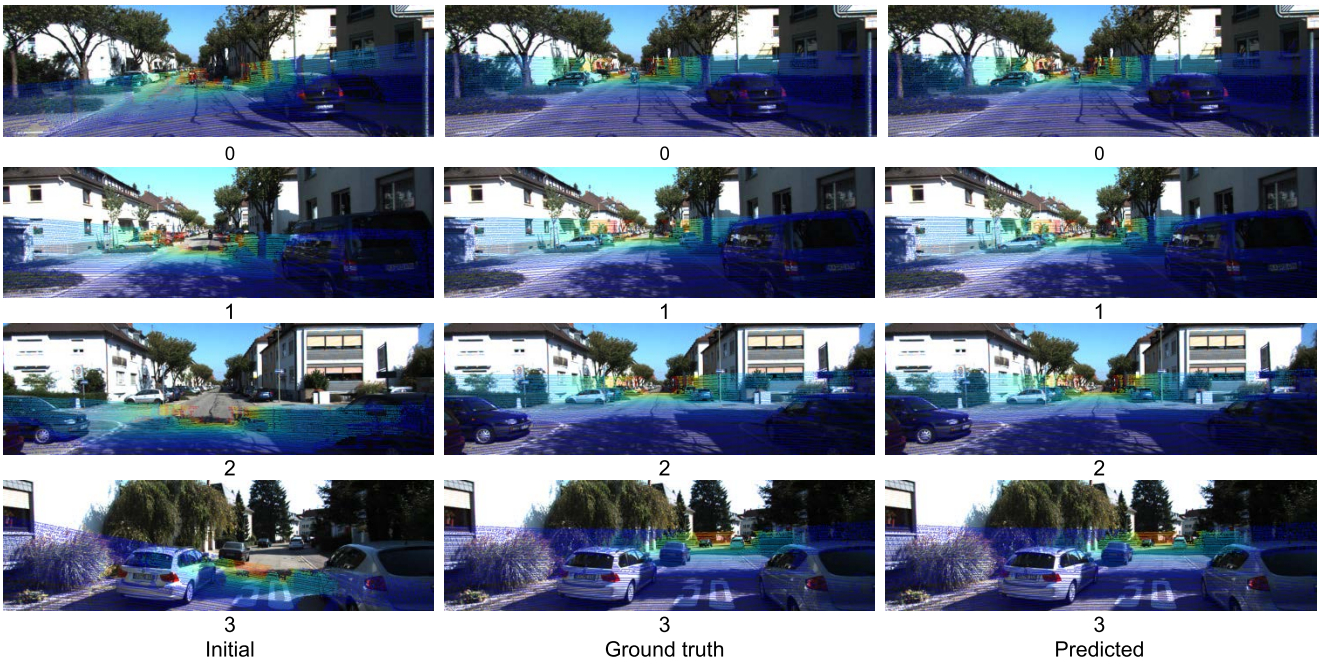

FIGURE 5. Examples of PSNet predictions.

performance of PSNet with these methods, we designed two sets of experiments with two datasets according to the experimental setting of each method. For CalibNet and CalibDNN, their approach was evaluated on the KITTI-raw sequence "2011_09_26," and CalibDNN outperformed CalibNet.

Compared with CalibDNN, PSNet achieved a mean absolute error for translation prediction of 0.030 m (x, y, z: 0.044, 0.015, 0.032 m), which was superior to CalibDNN and CalibNet, and a rotation prediction of 0.22° (roll, pitch, yaw: 0.17°, 0.29°, 0.20°), which was also acceptable. It is worth noting that we comprehensively considered several errors after calibration. By analyzing the test results on KITTI-raw sequence "2011_09_26," we found that the roll and yaw angle errors achieved by PSNet were worse than those of other methods listed in Table 1; however, the mean $\overline{R}$ value was similar to that of CalibDNN and the absolute translation errors in Y, Z were noticeably smaller than those of the other methods. Therefore, we concluded that PSNet outperforms CalibDNN and CalibNet.

For CalibRCNN, the approach was evaluated with the KITTI-Odometry dataset. It obtained a mean absolute error for translation prediction of 0.053 m (x, y, z: 0.062, 0.043, 0.054 m), and its rotation error was 0.42° (roll, pitch, yaw: 0.19°, 0.64°, 0.44°), whereas our method achieved a mean error of 0.031 m (x, y, z: 0.038, 0.028, 0.026 m) in translation and 0.15° (roll, pitch, yaw: 0.06°, 0.26°, 0.12°) in rotation. This shows that our results were better than those of CalibRCNN.

To the best of our knowledge, PSNet performs better than existing CNN-based calibration methods with a single model and single iteration.

Fig. 5 shows some examples of PSNet predictions. The first column represents LiDAR point cloud projections onto the image plane with the initial extrinsic parameters. The second column represents LiDAR point cloud projections onto the image plane with the ground truth extrinsic parameters, and the last column shows the corresponding calibrated results. As is apparent, the calibrated depth image is consistent with the ground truth. Although the initial depth maps shown in

TABLE 1. Calibration results and comparison with other methods.

Method	CalibNet	CalibDNN	Our Method	CalibRCNN	Our Method
Data set	KITTI-raw sequence "2011_09_26"	KITTI-raw sequence "2011_09_26"	KITTI-raw sequence "2011_09_26"	KITTI-Odometry dataset	KITTI-Odometry dataset
E_{Roll} (°)	0.15	**0.11**	0.17	0.19	**0.06**
E_{Pitch} (°)	0.90	0.35	**0.29**	0.64	**0.26**
E_{Yaw} (°)	**0.18**	**0.18**	0.20	0.44	**0.12**
$\overline{R}$ (°)	0.41	**0.21**	0.22	0.42	**0.15**
E_X (m)	0.042	**0.038**	0.044	0.062	**0.038**
E_Y (m)	0.016	0.018	**0.015**	0.043	**0.028**
E_Z (m)	0.072	0.096	**0.032**	0.054	**0.026**
$\overline{t}$ (m)	0.043	0.051	**0.030**	0.053	**0.031**

The second row shows the dataset used by each method. We compared the results with the same dataset settings and highlighted the smallest data in each row under the same experimental setting.

Fig. 5 are low quality, our method still achieved accurate calibration results.

2) TESTING ON PARTS OF THE KITTI-ODOMETRY DATASET

It is worth noting that CMRNet used the iterative approach to improve the accuracy of the final localization; different CNNs were trained by considering descending error ranges for both the translation and rotation components of the initial pose. In CMRNet, the values of different calibration deviation ranges were clearly stated.

TABLE 2. Iterative pose refinement.

	Initial Error Range		Localization Error	
	Translation (m)	Rotation(°)	Translation (m)	Rotation(°)
Initial pose	-	-	≈1.86	≈9.88
Iteration 1	[-2,+2]	[-10,+10]	0.34	1.01
Iteration 2	[-1,+1]	[-2,+2]	0.32	0.57
Iteration 3	[-0.6,+0.6]	[-2,+2]	0.25	0.63

TABLE 3. Comparison results using parts of the KITTI-odometry dataset.

	Translation (m)			Rotation(°)		
	Median	Mean	Std. Dev.	Median	Mean	Std. Dev.
CMRNet	0.27	0.33	0.22	1.07	1.07	**0.77**
Our method	**0.25**	**0.26**	**0.07**	**0.63**	**0.93**	1.21

The maximum deviation range for the first iteration was [±2 m, ± 10°] and for the second and third iterations were [±1 m, ± 2°] and [±0.6 m, ± 2°], respectively. The RGB images and corresponding point clouds input the maximum deviation range network, and then we regarded the prediction as T_0, and calibrated the miscalibrated point clouds with T_0. The RGB images and corresponding new calibrated point clouds were input to the next network to predict a new transformation T_1. In this case, the process to obtain the calibrated extrinsic calibration matrix is expressed as:

$$T_{LC} = (T_0 \times T_1 \times T_2)^{-1} \times T_{init}. \tag{12}$$

Table 2 shows the median localization errors for the three iterations. In Table 3, we provide a comparison between PSNet and CMRNet. The results indicate that PSNet outperformed CMRNet.

3) CALIBRATION ON UNTRAINED DATASETS

As stated in Section IV, different sequences were acquired in various scenes with different calibration parameters. The small sequences, which all belong to one sequence, have the same extrinsic and camera calibration parameters. The KITTI-Odometry dataset mainly consisted of sequences "2011_09_30" and "2011_10_03" and a small (801 frames) part of sequence "2011_09_26." Inspired by CalibRCNN, to test whether our method had good generalization ability, we trained a model on the KITTI-Odometry dataset and evaluated its performance on the KITTI-raw sequence "2011_09_26," which had unfamiliar scenes.

We observed that the mean errors for translation and rotation tested on KITTI-raw sequence "2011_09_26" were 0.042 m and 0.19°, respectively, which were slightly larger due to different scenarios and initial calibration parameters. The experimental results show that our method exhibited a much better generalization ability than CalibRCNN.

TABLE 4. Calibration results on untrained datasets and comparison with other methods.

Method	CalibRCNN	PSNet	CalibRCNN	PSNet
Training Dataset	KITTI-Odometry dataset	KITTI-Odometry dataset	KITTI-Odometry dataset	KITTI-Odometry dataset
Testing Dataset	KITTI-Odometry dataset	KITTI-Odometry dataset	KITTI-raw sequence "2011_09_26"	KITTI-raw sequence "2011_09_26"
E_{Roll} (°)	0.19	**0.06**	0.21	**0.16**
E_{Pitch} (°)	0.64	**0.26**	2.21	**0.16**
E_{Yaw} (°)	0.44	**0.12**	0.50	**0.24**
$\overline{R}$ (°)	0.42	**0.15**	0.97	**0.19**
E_X (m)	0.062	**0.038**	0.078	**0.027**
E_Y (m)	0.043	**0.028**	**0.032**	0.077
E_Z (m)	0.054	**0.026**	0.062	**0.022**
$\overline{t}$ (m)	0.053	**0.031**	0.057	**0.042**

The second and third rows show the training and testing datasets, respectively, used by each method. We compared the results with the same dataset settings and highlighted the smallest data in each row under the same experimental setting.

E. ABLATION STUDY

Our proposed calibration network consisted of two main stages: a backbone with a downsampling block and a multiscale feature aggregation module. To test the effectiveness of these modules, we designed an ablation study to prove our design choice on the KITTI-Odometry dataset.

1) DOWNSAMPLING BLOCK

We designed an ablation study to compare the effects of the backbone with or without the downsampling blocks. The backbone with the downsampling blocks used them to obtain the low-resolution representations, whereas the

TABLE 5. Downsampling block ablation experiment on KITTI-odometry dataset and comparison with ResNet-18.

Feature Extraction Network	Backbone with downsampling blocks	Backbone without downsampling blocks	ResNet-18
Params(M)	13.5	12.1	11.7
E_X (m)	0.038	0.070	**0.023**
E_Y (m)	0.028	**0.017**	0.022
E_Z (m)	0.026	**0.019**	0.053
$\overline{t}$ (m)	**0.031**	0.035	0.033
E_{Roll} (°)	**0.06**	0.40	0.47
E_{Pitch} (°)	**0.26**	0.28	0.61
E_{Yaw} (°)	**0.12**	0.37	0.71
$\overline{R}$ (°)	**0.15**	0.35	0.60

The second row shows the number of parameters of a single feature extraction network.

backbone without the downsampling blocks directly obtained the low-resolution representations through stride-2 3 × 3 convolutions. The results are provided in Table 5. We found that the backbone with the downsampling blocks performed better.

At the same time, most existing methods used ResNet-18 as their feature extraction network, whose number of parameters is similar to our method. Therefore, we designed a comparison experiment with ResNet-18. The results show that our feature extraction network outperforms ResNet-18.

2) MULTISCALE FEATURE AGGREGATION MODULE

We prepared an ablation study to test the effectiveness of our multiscale feature aggregation module. First, we studied how the representation resolution affects the calibration performance on different miscalibration ranges. To elucidate the experimental effect, we chose the translation and rotation ranges of [±1.5 m, ±20°], [±0.3 m, ±15°], and [±0.1 m, ±1°]. On each miscalibration range, we trained two different resolution networks (1/4, 1/32 size); and the results are shown in Table 6. We observed that a high-resolution model works better for a small miscalibration range, whereas a low-resolution model performs better for a large miscalibration range. Specifically, for the off-range of [±1.5 m, ±20°], we found that the absolute translation errors of the three models were similar; however, the rotation errors of a low-resolution model were noticeably smaller than those of a high-resolution model. Therefore, we believe a low-resolution model performs better for a large miscalibration range. Similarly, for the off-range of [±0.1 m, ±1°], we found that the absolute rotation errors of a high-resolution model were noticeably smaller than those a low-resolution model. Furthermore, we also observed that a model employing the multiscale feature aggregation module was robust against different initial error ranges due to the incorporation of multiresolution information in the fused features. At the same time, the runtime of the calibration process was 18 ms for an iteration on a single Nvidia RTX2080Ti GPU, which indicates that, although we used our multiscale feature aggregation module, the model still fulfilled real-time requirements.

TABLE 6. Multiscale feature aggregation module ablation experiment On KITTI-odometry dataset.

Initial Error Range	[±1.5 m, ±20°]			[±0.3 m, ±15°]			[±0.1 m, ±1°]		
Resolution	H	L	P	H	L	P	H	L	P
E_X (m)	**0.1**	0.1	0.1	**0.0**	0.1	0.0	0.0	0.0	**0.0**
	64	69	77	**61**	16	81	28	25	**21**
E_Y (m)	0.1	0.1	**0.1**	**0.0**	0.0	0.0	0.0	0.0	**0.0**
	72	89	**67**	**46**	88	49	24	32	**21**
E_Z (m)	0.2	**0.1**	0.1	0.0	0.0	**0.0**	**0.0**	0.0	0.0
	02	**82**	98	85	75	**66**	**22**	23	28
$\overline{t}$ (m)	**0.1**	0.1	0.1	**0.0**	0.0	0.0	0.0	0.0	**0.0**
	79	80	81	**64**	93	65	25	27	**23**
E_{Roll} (°)	0.6	**0.4**	0.5	0.2	0.2	**0.2**	**0.0**	0.1	0.1
	15	**68**	21	43	83	**37**	**71**	84	12
E_{Pitch} (°)	0.4	0.3	**0.2**	0.2	**0.2**	0.2	**0.0**	0.1	0.0
	55	57	**28**	97	**66**	81	**39**	56	97
E_{Yaw} (°)	0.4	**0.3**	0.4	0.2	0.2	**0.2**	**0.0**	0.2	0.1
	74	**70**	10	45	85	**41**	**75**	42	65
$\overline{R}$ (°)	0.5	0.3	**0.3**	0.2	0.2	**0.2**	**0.0**	0.1	0.1
	15	98	**86**	62	78	**53**	**62**	94	25

H , L, and P in column "Resolution" indicate whether the inference is carried out with high-resolution (1/4 size), low-resolution (1/32 size), or multiscale feature aggregation (our method). We highlighted the smallest data in each row under the same experimental setting.

V. CONCLUSION

In this study, we proposed a novel end-to-end calibration network for a 3D LiDAR and 2D camera system. The proposed network mainly consists of three key characteristics: (i) The addition of a downsampling block to increase suitability for sparse projected depth maps; (ii) The connection of the high-to-low resolution convolution streams in parallel to obtain multiresolution features that are spatially more precise and have richer semantic information; (iii) The fusion of the multiresolution streams by the multiscale feature aggregation module. We experimentally verified this intuition, namely, that a high-resolution model performs better for small miscalibration ranges, whereas a low-resolution model performs better for large miscalibration ranges. These methods could be aided by employing the iterative refinement approach considering descending error ranges to improve the calibration accuracy. The runtime of the calibration process was 18 ms for an iteration on a single Nvidia RTX2080Ti GPU, which meets real-time requirements.

We designed extensive experiments on different datasets to demonstrate the state-of-the-art performance of the proposed methods. Compared with existing single-model

single-iteration methods, the performance of our model was state-of-the-art. Compared with existing state-of-the-art methods with iteration refinement approaches and clear different calibration deviation range values, we achieved a competitive result. The results on untrained datasets showed that PSNet has good generalization ability. We plan to further improve its performance and apply CNNs to find the transformation between LiDAR and stereo cameras.

ACKNOWLEDGMENT

The authors would like to thank Editage (www.editage.cn) for English language editing.

REFERENCES

[1] A. Geiger, F. Moosmann, O. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 3936–3943.

[2] P. An, T. Ma, K. Yu, B. Fang, J. Zhang, W. Fu, and J. Ma, "Geometric calibration for LiDAR-camera system fusing 3D-2D and 3D-3D point correspondences," *Opt. Exp.*, vol. 28, no. 2, pp. 2122–2141, Jan. 2020, doi: 10.1364/OE.381176.

[3] G. Koo, J. Kang, B. Jang, and N. Doh, "Analytic plane covariances construction for precise planarity-based extrinsic calibration of camera and LiDAR," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6042–6048.

[4] 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.

[5] Y. Park, S. Yun, C. S. 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, 2014, doi: 10.3390/s140305333.

[6] C. Guindel, J. Beltran, D. Martin, and F. Garcia, "Automatic extrinsic calibration for LiDAR-stereo vehicle sensor setups," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–6.

[7] J. Kummerle, T. Kuhner, 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.

[8] T. Toth, 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.

[9] B. Nagy, L. Kovacs, and C. Benedek, "Online targetless end-to-end camera-LIDAR self-calibration," in *Proc. 16th Int. Conf. Mach. Vis. Appl. (MVA)*, May 2019, pp. 1–6.

[10] X. Liu and F. Zhang, "Extrinsic calibration of multiple LiDARs of small FoV in targetless environments," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2036–2043, Feb. 2021, doi: 10.1109/LRA.2021.3061387.

[11] G. Iyer, R. K. Ram, J. K. Murthy, and K. M. Krishna, "CalibNet: Geometrically supervised extrinsic calibration using 3D spatial transformer networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1110–1117.

[12] J. Shi, Z. Zhu, J. Zhang, R. Liu, Z. Wang, S. Chen, and H. Liu, "CalibRCNN: Calibrating camera and LiDAR by recurrent convolutional neural network and geometric constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 10197–10202.

[13] G. Zhao, J. Hu, S. You, and C.-C. J. Kuo, "CalibDNN: Multimodal sensor calibration for perception using deep neural networks," *Proc. SPIE*, vol. 11756, pp. 324–335, Apr. 2021.

[14] X. Lv, B. Wang, Z. Dou, D. Ye, and S. Wang, "LCCNet: LiDAR and camera self-calibration using cost volume network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 2894–2901.

[15] 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, Jul. 2021, doi: 10.1109/LRA.2021.3098923.

[16] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "CamVox: A low-cost and accurate LiDAR-assisted visual SLAM system," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 5049–5055.

[17] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic targetless extrinsic calibration of a 3D LiDAR and camera by maximizing mutual information," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, Toronto, ON, Canada, 2012, pp. 2053–2059.

[18] Z. Taylor and J. Nieto, "Automatic calibration of LiDAR and camera images using normalized mutual information," in *Proc. Robot. Automat. (ICRA)*. Karlsruhe, Germany, 2013, pp. 1–8.

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

[20] 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. (ITSC)*, Oct. 2019, pp. 1283–1289.

[21] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8934–8943.

[22] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Oct. 2021, doi: 10.1109/TPAMI.2020.2983686.

[23] A. Goyal, A. Bochkovskiy, J. Deng, and V. Koltun, "Non-deep networks," 2021, *arXiv:2110.07641*.

[24] L. Qi, J. Kuen, J. Gu, Z. Lin, Y. Wang, Y. Chen, Y. Li, and J. Jia, "Multi-scale aligned distillation for low-resolution detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 14443–14453.

[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE CVPR*, Jul. 2017, pp. 4700–4708.

[26] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A convolutional network for real-time 6-DOF camera relocalization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2938–2946.

[27] 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, doi: 10.1177/0278364913491297.

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