Hindawi
Mobile Information Systems
Volume 2022, Article ID 5058611, 10 pages
https://doi.org/10.1155/2022/5058611

Research Article

A Data-Driven Miscalibration Detection Algorithm for a Vehicle-Mounted Camera

Haiyang Jiang,[1] **Yuanyao Lu**,[2] **and Jingxuan Wang**[2]

[1]*School of Electrical and Control Engineering, North China University of Technology, Beijing, China*
[2]*School of Information Science and Technology, North China University of Technology, Beijing, China*

Correspondence should be addressed to Yuanyao Lu; luyy@ncut.edu.cn

Received 6 April 2022; Revised 15 June 2022; Accepted 11 October 2022; Published 25 October 2022

Academic Editor: Ashish Bagwari

LiDAR and camera are two commonly used sensors in autonomous vehicles. In order to fuse the data collected by these two sensors to accurately perceive the 3D world, it is necessary to perform accurate internal parameters' and external parameters' calibration on the two sensors. However, during the long-term deployment and use of autonomous vehicles, factors such as aging of the equipment, transient changes in the external environment, and interference can cause the initially correctly calibrated camera internal parameters to no longer be applicable to the current environment, requiring a recalibration of the camera internal reference. Since most of the current work is focused on the research of perception algorithms and the calibration of various sensors, there has not been much research in identifying when a sensor needs to be recalibrated. Consequently, this paper proposed a data-driven detection method for the miscalibration of RGB cameras to detect the miscalibrated camera internal parameters. The specific operation process is to first add a random perturbation factor to the correctly calibrated camera internal parameters to generate an incorrect camera internal parameter and then calibrate the raw image with the incorrect internal parameter to generate a miscalibrated image data. The miscalibrated image data are used as the input data of the neural network to train the network model and generate a network model for detecting the miscalibration parameters. On the KITTI dataset, we conducted training as well as model deployment with the data collected from Cam2 and Cam3, respectively, and evaluated the abovementioned two models. The experimental results show that our proposed method has some application value in detecting errors in the calibration of the camera's internal parameters.

1. Introduction

In autonomous driving, a single sensor cannot accurately perceive the complex and ever-changing road traffic environment, and therefore, a fusion of data collected by several different kinds of sensors is the mainstream solution for current autonomous driving perception systems. To enable a more accurate fusion of sensor data, accurate calibration of the sensors becomes the basis for all applications. As two commonly used sensors in autonomous vehicles, LiDAR and cameras, once their calibrated internal and external parameters are determined, they will be in a constant state throughout the operating period of the autonomous vehicle. However, during long-term deployment and use, due to the aging of the camera equipment, transient changes in the external environment, and distraction, the initial calibration of the camera's internal parameters are no longer applicable to the current environment, and the camera's internal parameters need to be recalibrated. The periodic calibration of the internal parameters of the camera will waste a lot of manpower and resources. A better state would be to calibrate the camera when the system detects a miscalibration, so identifying when the camera needs to be recalibrated is the main purpose of this paper. Miscalibrated camera internal parameters can adversely affect the performance of the perception and control modules for autonomous driving, which makes detecting camera data faults crucial to the safety and stability of autonomous vehicles. In order to avoid system errors caused by the miscalibration of camera internal parameters, this paper proposes a data-driven RGB

camera miscalibration detection method to detect the miscalibration of camera internal parameters. The main contributions of this paper are as follows:

(1) We propose a method to generate miscalibration datasets. The method is based on the idea that incorrect calibration parameters will cause the pixel projection position of the raw image to change relative to the pixel projection position with the correctly calibrated parameters. Adding a random perturbation factor to existing correctly calibrated camera internal parameters to generate incorrect camera internal parameters, thus generating miscalibration image data.

(2) We designed a convolutional neural network that uses incorrectly calibrated images as the input data and trained a network model that can detect whether the calibration of the camera is incorrect and thus identify when the camera needs to be recalibrated for internal parameters.

2. Related Work

Projecting the point cloud onto the image requires a joint calibration of the LiDAR and the camera. The calibration process of different sensors is the process of estimating the rigid body transformation between two sensor reference coordinate systems. The correctly calibrated sensor can reproject the 3D points from the world coordinate system to the 2D pixel coordinate system, and the conversion process of the coordinate system is shown in Figure 1.

During the projection of the point cloud onto the image, accurate camera calibration is essential to ensure that the point cloud data are accurately projected onto the image. The current calibration techniques for cameras are mainly divided into two methods: offline calibration [1–3] and online calibration [4, 5]. Offline calibration of the camera is performed by observing the geometry of a known target in 3D space, so various types of calibration boards such as checkerboard grids [6, 7] and targets [8] have been proposed in recent years. These methods usually treat the calibration problem as a nonlinear optimization problem, estimating the camera parameters by minimizing the reprojection error. However, it is difficult to scale up in practical applications due to the limitations of equipment and professional staff and other requirements. Online calibration refers to the calibration performed during the normal operation of the system. Although the method is novel, it requires substantial computational resources and is also subject to various restrictive factors such as the smoothness of the surface on which the vehicle is driven. In addition, a self-calibration method is introduced in reference [9], which uses only images to estimate the camera's internal parameters. References [10–13] use motion constraints of the camera to calibrate the internal parameters of the camera, but these methods consume a lot of computational resources.

With the rapid development of techniques such as deep learning [14–16] and computer vision [17–21], in recent years, many researchers have proposed the use of data-driven methods to estimate the calibration of sensors. Workman et al. [22] used a convolutional neural network to estimate the focal length in the camera's internal parameter. To train the network, the authors of [23] combined the image and camera model to construct the dataset. Lopez et al. [24] used the SUN360 panoramic dataset [25] to manually generate training images and then used independent regressors that shared the same pretrained network structure in order to estimate the camera's internal parameters. Unlike the previous two methods, Yin et al. [26] used a depth network to remove distortions from fisheye camera images. In autonomous driving systems, sensor fault detection can be accomplished by combining information from different sensors and removing mismatched measuring values [27–31]. These approaches are usually able to detect faults when they occur, but they rely heavily on redundant information about the sensor settings.

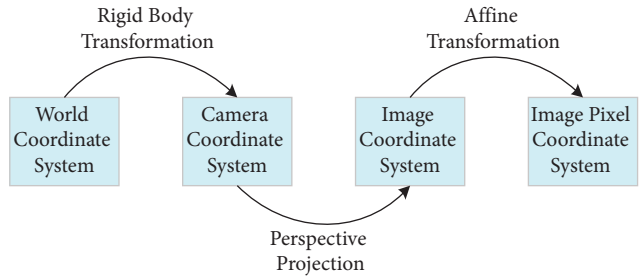

Figure 1: Coordinate transformation process of projecting the point cloud onto the image. The points in space are first converted from the world coordinate system to the camera coordinate system by rigid body transformation, then projected to the imaging plane (image coordinate system) by perspective projection, and finally, the data in the imaging plane are converted to the image pixel coordinate system by an affine transformation.

LiDAR and camera are two commonly used sensors in autonomous vehicles. In the process of point cloud and image fusion, the calibration of internal parameters of the camera and external parameters between the camera and the lidar is essential, and their calibrated internal and external parameters, once determined, will be in a constant state during the whole operating cycle of the autonomous vehicles.

However, during long-term deployment and use, due to the aging of the camera, transient changes, and interference in the external environment, the camera's internal parameters that were initially correctly calibrated are no longer applicable to the current environment and need to be recalibrated. Periodically recalibrating parameters can be labor-intensive; it would be more sensible to only recalibrate the camera when the system detects a calibration error, which makes identifying when the camera needs recalibration a pressing issue.

To address this problem, in 2020, Cramariuc [32] et al. proposed a method that uses deep learning to identify the errors in the calibration of the camera's internal parameters; this method is one of the first works to propose the use of deep learning methods to detect whether the miscalibration of the RGB camera internal parameters. In their paper, the authors designed a neural network to extract the features of

the input data by using ReLU as the activation function of the network. Due to the problem of neuron death in the ReLU activation function, the network cannot be updated properly in training.

In summary, to address these problems, we propose a data-driven RGB camera miscalibration detection method to detect the miscalibrated camera internal parameters, and we design a feature extraction network using Leaky ReLU as the activation function, which can solve the neuron death problem of the ReLU activation function used in traditional methods. The Leaky ReLU activation function has a small positive slope in the negative half-axis, so it can also perform better backpropagation when the input data are negative. At the same time, in the process of generating the miscalibrated image dataset, we introduce a random disturbance factor to perturb the correctly calibrated camera internal parameters so as to obtain the miscalibrated camera internal parameters. This allows efficient and accurate identification of when the camera needs to be recalibrated.

3. System Design

3.1. Image Datasets for Error Correction. To obtain the image data of the correction error, you first need to obtain the miscalibration camera's internal parameters. Using the wrong internal parameters to calibrate the camera's raw image produces miscalibrated image data. In general, using a different lens to change the camera's internal parameter is one way to acquire miscalibrated camera's internal parameters. However, this production process is cumbersome because every lens change requires a new offline calibration. Calibrating the raw image with wrong calibration parameters changes the position relative to calibrating with the right ones. Based on such an idea, we use the raw data in the KITTI dataset and the correctly calibrated camera internal parameters to generate the miscalibrated image data by adding a random perturbation factor to the correct camera internal parameters, so that the correctly calibrated internal parameters become miscalibrated. Since the lens of the camera is not perfectly parallel to the imaging plane, lines that are straight in the real world become curved when projected onto the 2D plane through the camera and need to be corrected for distortion by using a calibrated distortion factor (as shown in Figure 2). Therefore, in the process of generating image datasets with calibration errors, we considered pinhole camera models with radial and tangential distortions [33].

The paper [32] generates the miscalibrated image data by first fluctuating a fixed magnitude from left to right on the basis of the correctly calibrated camera internal parameters given in KITTI, so that the miscalibrated calibrated camera internal parameters are randomly selected from this parameter range (The value range of the parameter for the focal length miscalibration is 5% decrease to the left and 20% increase to the right on the basis of the original correct calibration parameter; the value range of the parameter for the optical center miscalibration is 5% decrease to the left and 5% increase to the right on the basis of the original correct calibration parameter, and the value range of the parameter for the aberration coefficient miscalibration is 15% decrease to the left and 15% increase to the right on the basis of the original correct calibration parameter). Since the KITTI [34] dataset has a total of 5 days of raw image data and the camera calibration parameters are different for each of the 5 days, this approach can lead to a large difference in the miscalibrated camera parameters generated each day.

Figure 2: (a) The raw data from KITTI [34]. The three vertical lines that are pointed by the three arrows should normally be straight lines, but it is obvious that they are not straight, but curved with some curvature. (b) The result of calibrating the raw image with the correctly calibrated camera internal parameters, all three vertical lines have become straight.

To address the abovementioned problem, we propose a method of adding a random perturbation factor to generate the miscalibrated camera internal parameters to reduce the gap between the miscalibrated camera internal parameters generated each day. The operation process is as follows: first, the calibration file of September 26 is used as the reference, the range of values of the miscalibrated internal parameters is fixed according to the method in the literature [36]; then, the corresponding miscalibrated camera internal parameters are obtained from the range of values of each parameter by random sampling, and the original image is calibrated with these internal parameters to obtain the miscalibrated image data. The distance between the pixel position of the miscalibrated image and the correctly calibrated image is calculated. Using this distance as a standard, a random perturbation factor is added to the calibration parameters for the other days, so that the pixel difference between the generated miscalibrated image and the pixel difference on September 26 is controlled within a fixed threshold.

We assume that the original image captured by the camera is I; the corrected image I' is obtained by using the true calibration parameters $\Theta = \{f_u, f_v, u_c, v_c, k_r, k_t\}$ of the pinhole camera model by mapping function $M' = f(\Theta)$. At this moment, each pixel in I' is associated with a position in the original image, but not every pixel in I' can find a corresponding position in the original image. Therefore, we define the largest rectangular area of valid pixels in image I' as R, then crop the region R, resize it to the size of the original image I according to a certain ratio, and finally obtain the sample image $\widehat{I}$.

Usually, the calibration parameters of the camera are difficult to obtain. In order to obtain samples of incorrectly calibrated images, we get the incorrectly calibrated internal parameter Θ^m by sampling each internal reference in its corresponding range of values independently at several random times, thus obtaining many incorrectly calibrated images $\widehat{I}^m$ and incorrectly corrected mapping functions $\widehat{M}^m$. Thus, by calibrating the sensor correctly, a large number of incorrectly calibrated images can be generated by acquiring only one set of raw image data, which are used to detect whether the internal parameters of the camera is miscalibrated. The variation of the corrected position of the raw image with correct and incorrect calibration parameters is shown in Figure 3.

3.2. Metric for the Degree of Error in the Calibration of the Camera's Internal Parameters. Because the calibration of the input image is performed in the first stage of the perception system of the autonomous vehicle, the extent of the camera's internal parameters calibration error needs to be portrayed by using miscalibrated images. Thus, we use an average pixel position difference (APPD) [32] metric to reflect the extent of error of the calibration parameters. The APPD is the average of all pixel position deviations on an image, and its expression is shown below:

$$\delta = \frac{1}{H \times W} \sum_{p \in I} \| \widehat{M}^* (p) - \widehat{M}^m (p) \|_2, \tag{1}$$

where $H \times W$ is the size of the image. p is the pixel coordinate (u, v), $\widehat{M}^*$ is the correct mapping function, and $\widehat{M}^m$ is the error mapping function and I is the raw image.

To accurately detect if the camera calibration is wrong, we designed a convolutional neural network to extract features. The input data of the network are the picture $\widehat{I}^m$ with the incorrect calibration parameters. The final output of the network is the APPD metric, and the structure of the network is shown in Figure 4.

In contrast to reference [32], we use the Leaky ReLU activation function after each convolutional layer, which solves the neuron death problem that exists with ReLU. The leaky ReLU activation function still has a small positive slope in the negative half-axis, thus allowing better backpropagation when the input data are negative. The ReLU and the leaky ReLU activation functions are shown in Figure 5.

The ReLU activation function can avoid gradient disappearance during backpropagation, shield negative values, and prevent gradient saturation, but it also has its own drawbacks. When the learning rate is too high, some neurons will die permanently, resulting in the network cannot be updated properly later on. The neural network weight update formula:

$$\mathscr{W}' = \mathscr{W} - \eta \Delta \mathscr{W}, \tag{2}$$

where η is the learning rate, $\Delta \mathscr{W}$ indicates the gradient of the current parameter obtained by derivation (generally positive), when the learning rate is too high, it will cause $\eta \Delta \mathscr{W}$ to rise up, when $\eta \Delta \mathscr{W}$ is greater than $\mathscr{W}$, the updated $\mathscr{W}'$ will become negative. When the weight parameter becomes

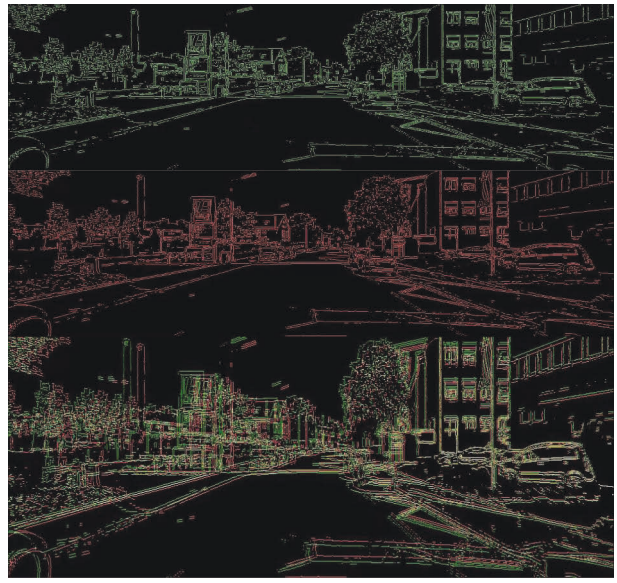

Figure 3: (a) The result of image edge detection after calibrating the raw image with the correct calibration parameters. (b) The result of image edge detection after calibrating the raw image with miscalibration parameters. (c) The result of overlapping the two images, you can see that the position of the corresponding pixel points in the two images has changed.

negative, the positive value of the input network will be multiplied by the weight and will also become negative. According to the image of the ReLU function in Figure 5, the negative value will output 0 after passing ReLU; if W has a chance to be updated to a positive value at a later stage, there will not be a big problem, but when the output value of Relu function is 0, the derivative of ReLU will also be 0, so it will lead to the later $\Delta \mathscr{W}$ to be 0 all the time. In turn, this leads to the fact that $\mathscr{W}$ will never be updated and therefore will lead to the permanent death of this neuron (always output 0).

As shown in Figure 5, the output value of Leaky ReLU is also less than 0 when the input is less than 0, thus having the opportunity to update to a positive value if $\mathscr{W}$ is less than 0.

Since the acquisition platform for the KITTI [34] dataset uses two cameras, we trained a neural network model for each of these two cameras and deployed it next to them. This can be considered a complement to the calibration procedure. The purpose of describing the dataset generation process is to reduce the amount and type of data required to train the model. With this method, when the correct calibration is known, only one dataset is enough; any manual annotation is not required.

In the training process, each parameter in the camera's internal parameter is sampled uniformly and independently within its corresponding value range, so that the calculated APPD values follow an approximately uniform distribution. We selected a certain percentage of correctly calibrated samples, i.e., those with zero APPD values. We use the mean error loss function of the output of the neural network and the true label value (as shown in (3)) for training:

$$\text{loss} = \frac{1}{n} \sum_{n} \| y\_\text{pred} - y\_\text{true} \|^2, \tag{3}$$

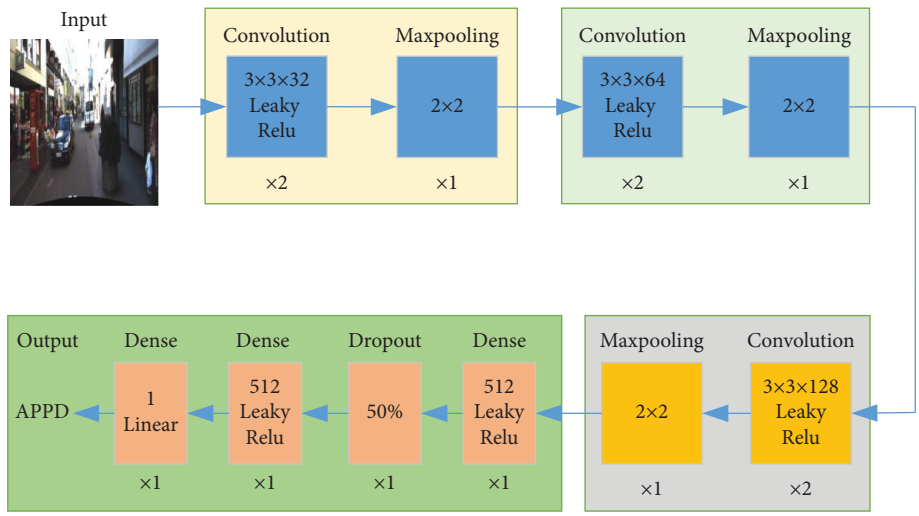

Figure 4: The structure of the network.

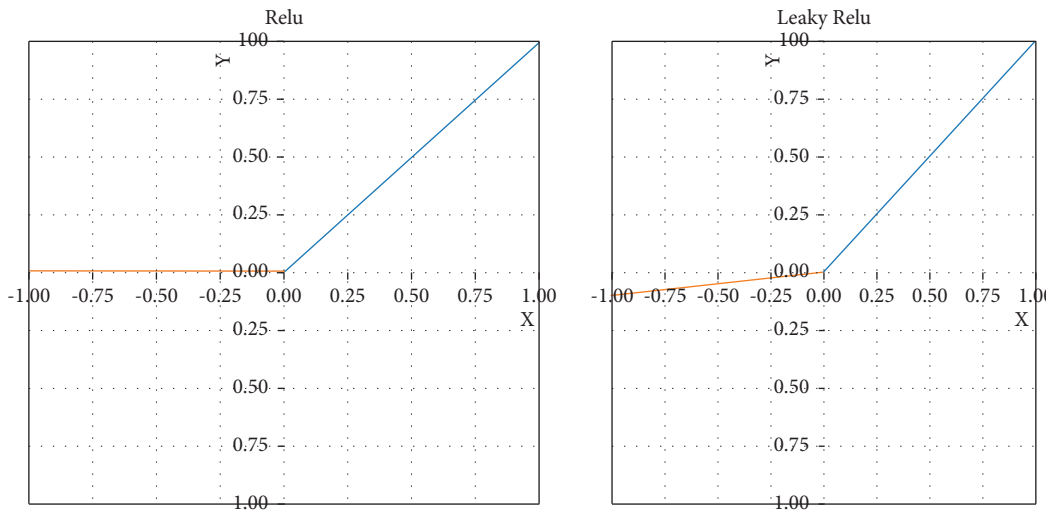

Figure 5: ReLU and leaky ReLU activation functions.

where n is the batch_size, y_pred is the APPD value of the neural network output, and y_true is the true APPD value of the training sample.

4. Experiments

4.1. Setting Experimental Parameters. The collection platform for the KITTI dataset [34] has two RGB cameras (e.g., Cam2 and Cam3 in Figure 6). We first divide the data from September 26, 2011, into a training set and a validation set. The trained model was tested with data from September 28, September 29, September 30, and October 03.

The dataset includes a total of 5 days of data and the calibration files are different for each day. The reason for this situation is not clear. Therefore, there is no standard correct calibration when evaluating the performance of network models. We use the calibrated parameters from September 26, 2011, as reference values. The network model is trained with batch_size set to 4, epoch set to 10, and learning rate set to 0.0001, and its loss curve for training and validation is shown in Figure 7.

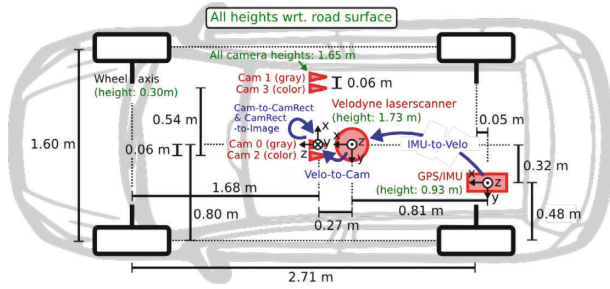

Figure 6: Top view of the data collection platform for the KITTI [34] dataset.

The two curves in Figure 7 show the convergence of the loss function of the network model during the training process. By analyzing the two curves in the figure, it can be concluded that the convergence of the loss function during the training process is fast. The trend of the validation loss and training loss remains basically the same. When the training reaches the 4th epoch, the loss value is basically close to 0 and the network model converges.

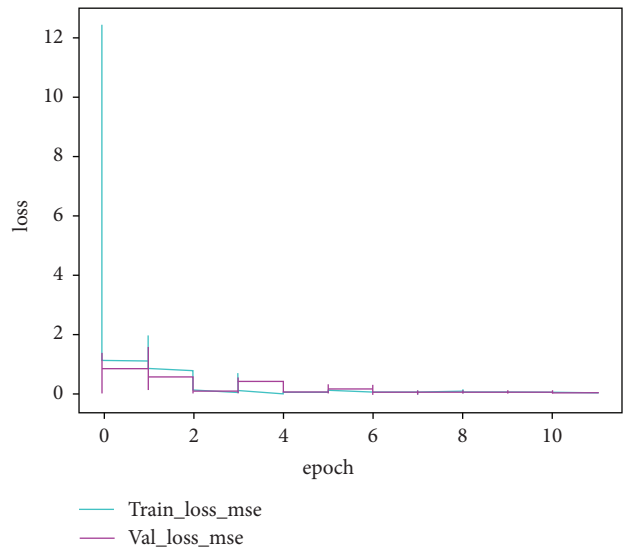

Figure 7: Convergence curves for training and validating loss functions.

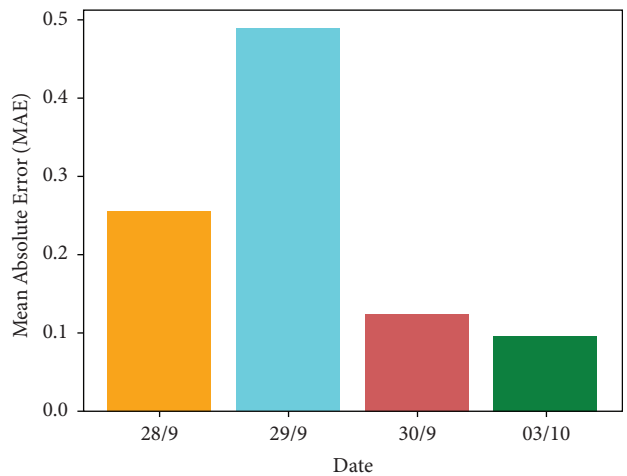

Figure 8: Test results of the network model.

4.2. Analysis of Prediction Results. We tested the trained model by using data from dates other than September 26, and the results are shown in Figure 8.

As can be seen from Figure 8, the average absolute error of our trained model on the data of other dates is within 0.5, which indicates the feasibility of our proposed method in detecting whether there is an error in the calibration of the internal reference of the camera. The worst evaluation result was obtained for the data on September 29, with an average absolute error close to 0.5. The possible reason for this is the high diversity of the data on September 29.

We trained the network models for the Cam2 camera data and Cam3 camera data on September 26 and tested the two network models with data from other dates, and the test results are shown in Tables 1 and 2.

Tables 1 and 2 show the prediction results of the trained network models for the Cam2 data and Cam3 data, respectively, which shows that both network models have a good generalization ability to previously unseen images and environments. Although both network models are powerful in detecting miscalibration relative to the reference parameter set they were trained on, it can be seen from Table 1 that the network of Cam2 performs better. It is also evident from Tables 1 and 2 that the trained models perform worse at both very low and very high APPD values.

In the KITTI data collection platform, Cam2 and Cam3 are from the same brand, and their operating environment and location in the car are horizontal, so we evaluated the Cam3 data with the model trained on the Cam2 data (Figure 9), and the Cam2 data with the model trained on the Cam3 data (Figure 10).

By analyzing Figures 9 and 10, we can see that the model trained on Cam2 can be well generalized to Cam3. While the model trained on Cam3 does not generalize well to Cam2. This illustrates the importance of reference calibration, which is another reason why Cam3 may be inconsistently calibrated.

To verify the effectiveness of the Leaky ReLU activation function we used in this experiment, we trained the data collected by using Cam2 on September 26 by using both ReLU and Leaky ReLU activation functions, respectively, and tested them with data that are not used in training. The test results are shown in Figure 11.

From Figure 11, it can be seen that when the learning rate is 0.000001, the network model with Leaky ReLU is not as effective as the network model with ReLU. As the learning rate increases, the network model with Leaky ReLU is significantly better than the network model with ReLU. When the learning rate is 0.0001, the average absolute error of the trained network model is the smallest, and the network model performs the best.

4.3. Generalization to New Environments and Cameras. Since the KITTI dataset is limited by the variation of the scene as well as the camera sensor positions (both cameras are forward facing with only horizontal offset between them); therefore, to test the potential generalization capability of the proposed method, we trained the same model by using our proposed method on some daytime scenes recorded by using the forward-facing cameras of the Waymo dataset [35] and tested the trained model, the results of which are shown in Figure 12.

From Figure 12, it can be seen that the network model trained on the Waymo dataset still has great generalization ability, but there are big errors in the prediction for both small and large APPD values. The data of their test results are generally similar to those of the network model trained on the KITTI dataset, respectively. It shows that the method proposed in this paper is still applicable to other datasets.

Table 1: Train on cam2, test on cam2.

	True APPD											
Prediction	0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75
0.00~0.20	476	215	4	2	0	0	0	0	4	0	5	0
0.21~0.40	58	538	27	0	6	0	0	5	2	0	43	0
0.41–0.65	133	153	805	17	31	2	0	0	0	0	32	0
0.66~0.90	0	6	41	835	25	34	3	0	0	0	109	0
0.91~1.15	7	0	37	23	826	6	7	13	0	0	0	45
1.16~1.40	0	0	0	0	43	875	26	14	9	0	0	85
1.41~1.65	0	3	0	3	16	14	877	8	38	0	0	0
1.66~1.90	43	0	0	9	0	3	53	905	23	0	0	7
1.91~2.15	0	0	0	0	0	12	3	32	883	12	152	0
2.16~2.45	0	0	0	2	0	0	0	13	15	931	32	103
2.46~2.65	0	0	3	0	3	1	0	0	2	16	497	163
2.66~3.00	0	0	0	0	0	1	3	0	6	0	42	462

Table 2: Train on cam3, test on cam3.

	True APPD											
Prediction	0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75
0.00~0.20	432	18	0	13	0	0	0	0	0	0	0	0
0.21~0.40	25	364	23	0	0	3	0	0	0	0	0	0
0.41~0.65	69	16	536	62	35	0	0	6	0	7	3	0
0.66~0.90	103	0	124	749	12	54	7	0	0	0	0	0
0.91~1.15	0	0	32	53	820	32	3	0	7	0	0	64
1.16~1.40	54	24	0	21	43	834	13	16	0	0	16	0
1.41~1.65	74	86	6	0	27	17	853	23	8	1	0	0
1.66~1.90	11	43	22	0	0	8	8	842	4	5	0	15
1.91~2.15	3	6	0	0	0	3	3	31	857	3	24	43
2.16~2.45	0	0	0	2	0	0	0	9	13	894	231	0
2.46~2.65	0	0	0	23	0	3	0	7	3	9	527	125
2.66~3.00	26	0	0	2	0	5	2	2	0	14	75	446

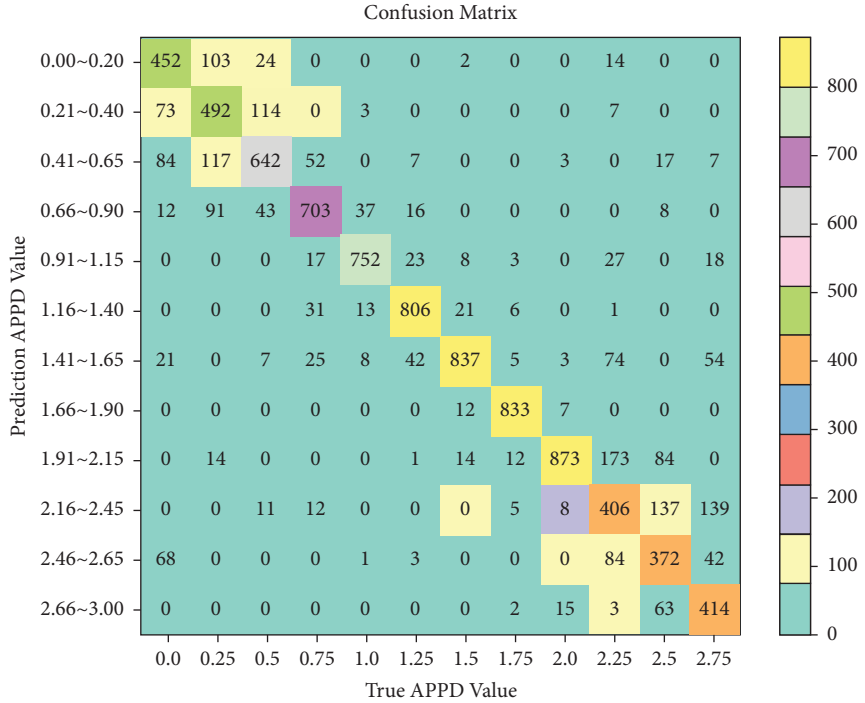

Figure 9: Result of testing cam2 trained model on cam3.

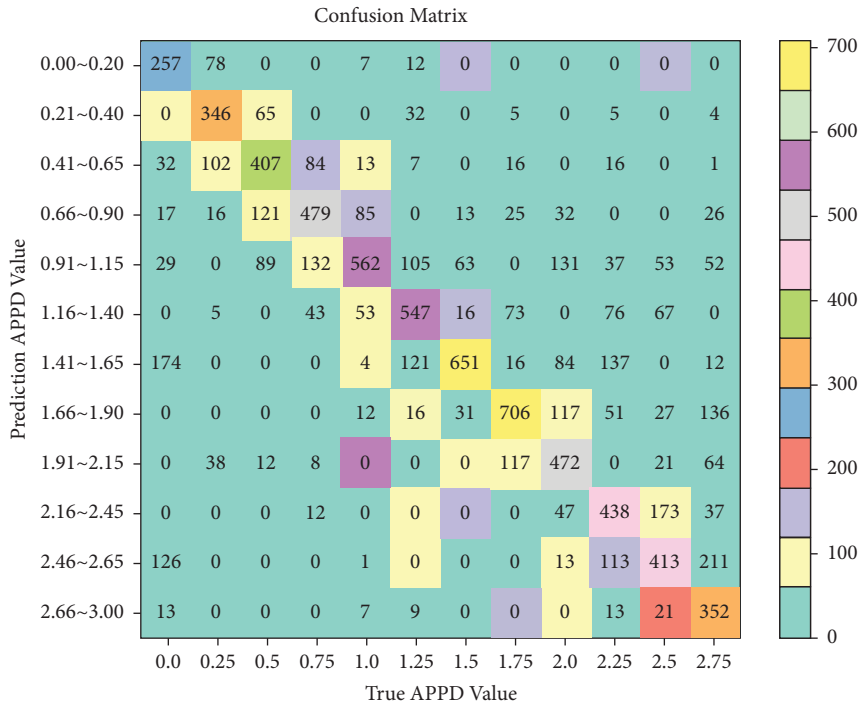

Figure 10: Result of testing cam3 trained model on cam2.

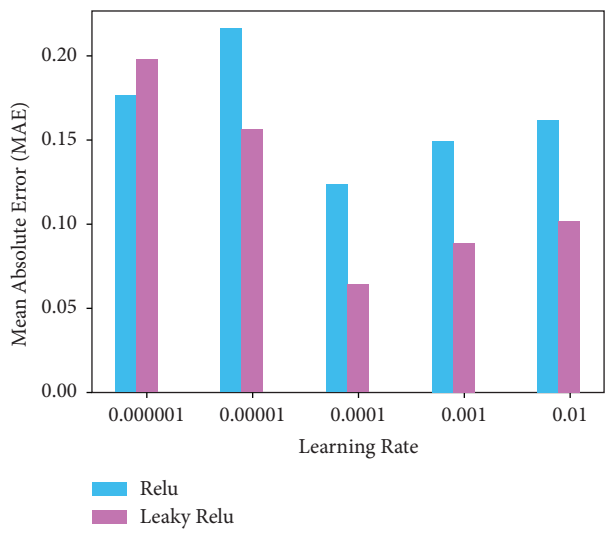

Figure 11: Performance of network models with different activation functions at different learning rates.

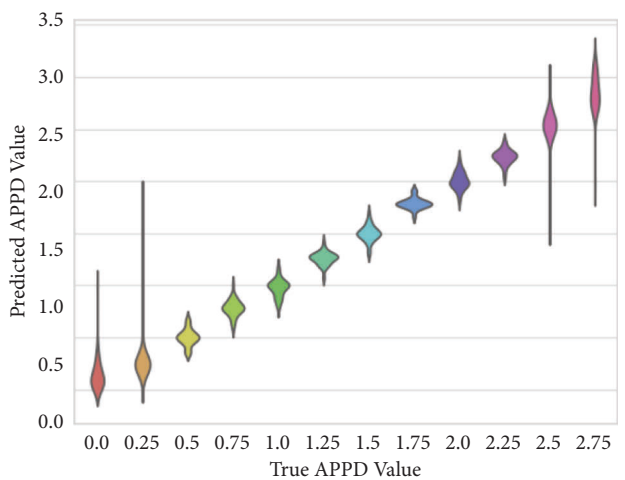

Figure 12: Test results of the network model on the Waymo dataset [36].

5. Conclusion

In this paper, we propose a data-driven method for detecting vehicle camera miscalibration by generating a set of miscalibrated data on the KITTI dataset by using a random perturbation factor as the training data for the neural network. Also, we designed a lightweight feature extraction network to extract the features of miscalibrated images. Finally, the metric of average pixel position difference mentioned in reference [32] was used to measure the degree of error in the calibration of the camera's internal reference. Our method solves the problem that it is not necessary to recalibrate the camera internal reference periodically but only to recalibrate the camera's internal reference when an internal reference calibration error is detected. The current problem with our approach is that for different autonomous vehicle platforms, depending on the camera and radar mounting positions and angles, a separate model needs to be trained for each camera, thus ensuring that the camera's internal parameters can be recalibrated when the camera position changes. However, this also makes it impossible to apply models from one acquisition platform to another. In our next work, we intend to mix multiple datasets from different acquisition platforms to train the network model, so that the network model can be adapted to multiple acquisition platforms with better generalization capability.

Data Availability

All data and programs included in this study are available upon request by contact with the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (61971007 and 61571013).

References

[1] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, "LiDAR camera calibration using 3D-3D point correspondences," *ArXiv eprints*, vol. 65, p. 199, 2017.

[2] W. Zhen, Y. Hu, J. Liu, and S. Scherer, "A joint optimization approach of LiDAR-camera fusion for accurate dense 3-D reconstructions," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3585–3592, 2019.

[3] T. Schneider, M. Li, M. Burri, J. I. Nieto, R. Siegwart, and I. Gilitschenski, "Visual-inertial self-calibration on informative motion segments," *IEEE International Conference on Robotics and Automation*, pp. 6487–6494, 2017.

[4] J. A. Preiss, K. Hausman, G. S. Sukhatme, and S. Weiss, "Simultaneous self-calibration and navigation using trajectory optimization," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1573–1594, 2018.

[5] H.-J. Chien, R. Klette, N. Schneider, and U. Franke, "Visual odometry driven online calibration for monocular LiDAR-camera systems," in *Proc. The 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2848–2853, 2016.

[6] J.-K. Huang and J. W. Grizzle, "Improvements to target-based 3D LiDAR to camera calibration," *IEEE Access*, vol. 8, pp. 134101–134110, 2020.

[7] S. Verma, J. S. Berrio, and S. Worrall, "Automatic extrinsic calibration between a camera and a 3D Lidar using 3D point and plane correspondences," *IEEE Intelligent Transportation Systems Conference (ITSC). IEEE*, pp. 3906–3912, 2019.

[8] Z. Tang, M. Naphade, M. Y. Liu, and X. D. Yang, "A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification," vol. 72, p. 944, Proc. CVPR, 2019.

[9] B. Boudine, "A flexible technique based on fundamental matrix for camera self-calibration with variable intrinsic parameters from two views," *Journal of Visual Communication and Image Representation*, vol. 39, pp. 40–50, 2016.

[10] M. Shan, J. S. Berrio, and S. Worrall, "Probabilistic egocentric motion correction of LiDAR point cloud and projection to camera images for moving platforms," in *Proceedings of the IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–8, IEEE, Rhodes, Greece, 20-23 September 2020.

[11] Q. Liao, Z. Chen, Y. Liu, Z. Wang, and M. Liu, "Extrinsic calibration of LiDAR and camera with polygon," *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, in *Proceedings of the 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 200–205, Kuala Lumpur, Malaysia, 12-15 December 2018.

[12] J. Jiang, P. Xue, S. Chen, Z. Liu, X. Zhang, and N. Zheng, "Line feature based extrinsic calibration of LiDAR and camera," *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, in *Proceedings of the 2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pp. 1–6, Madrid, Spain, 12-14 September 2018.

[13] T. M. Nguyen, Q. H. Pham, L. B. Doan, H. V. Trinh, and V. A. Nguyen, "Contrastive learning for natural language-based vehicle retrieval," *Computer Vision and Pattern Recognition*, vol. 65, p. 2898, 2021.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, China, 2016, http://www.deeplearningbook.org.

[15] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *ICLR*, vol. 3898, pp. 18–34, 2015.

[16] E. Denton, S. Chintala, A. Szlam, and R. Fergus, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Proceedings of the 29th Annual Conference on Neural Information Processing Systems (NIPS)*, Spain, 13-23 July 2019.

[17] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision*, pp. 132–149, ECCV, 2018.

[18] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *Advances in Neural Information Processing Systems*, vol. 33, p. 53, 2020.

[19] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirectional transformers for language understanding," 2018, http://arxiv.org/abs/1810.04805.

[20] L. Dong, N. Yang, W. H. Wang et al., "Unified language model pre-training for natural language understanding and generation," in *Advances in Neural Information Processing Systems*, pp. 13063–13075, 2019.

[21] T. Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, 2017.

[22] S. Workman, C. Greenwell, M. Zhai, R. Baltenberger, and N. Jacobs, "Deepfocal: a method for direct focal length estimation," in *IEEE International Conference on Image Processing*, pp. 1369–1373, 2015.

[23] K. Wilson and N. Snavely, "Robust global translations with 1dsfm," in *Computer Vision - ECCV 2014 European Conference on Computer Vision*, pp. 61–75, Springer, 2014.

[24] M. Lopez, R. Mari, P. Gargallo, Y. Kuang, J. Gonzalez-Jimenez, and G. Haro, "Deep single image camera calibration with radial distortion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 817–825, 2019.

[25] Z. Tang and J. N. Hwang, "MOANA: an online learned adaptive appearance model for robust multiple object tracking in 3D," *IEEE Access*, vol. 7, no. 1, pp. 31934–31945, 2019.

[26] X. Yin, X. Wang, J. Yu, M. Zhang, P. Fua, and D. Tao, "FishEyeRecNet: a multi-context collaborative deep network for fisheye image rectification," in *Proceedings of the European Conference on Computer Vision*, pp. 469–484, 2018.

[27] J. P. Mendoza, M. Veloso, and R. Simmons, "Mobile robot fault detection based on redundant information statistics," in *IROS Workshop on Safety in Human-Robot Coexistence and Interaction Vilamoura, Portugal,* vol. 945, p. 7, Citeseer, 2012.

[28] E.-S. Kim and S.-Y. Park, "Extrinsic calibration of a camera-LiDAR multi sensor system using a planar chessboard," *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, IEEE, in *Proceedings of the 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 89–91, 02-05 July 2019.

[29] J. Jiao, Q. Liao, Y. Zhu et al., "A novel dual-LiDAR calibration algorithm using planar surfaces," 2019, http://arxiv.org/abs/1904.12116.

[30] S. Verma, J. S. Berrio, S. Worrall, and E. Nebot, "Automatic extrinsic calibration between a camera and a 3D LiDAR using 3D point and plane correspondences," 2019, http://arxiv.org/abs/1904.12433.

[31] T. Schneider, M. Li, C. Cadena, J. Nieto, and R. Siegwart, "Observability-aware self-calibration of visual and inertial sensors for ego-motion estimation," *IEEE Sensors Journal*, vol. 19, no. 10, pp. 3846–3860, 2019.

[32] A. Cramariuc, A. Petrov, R. Suri, M. Mittal, R. Siegwart, and C. Cadena, "Learning camera miscalibration detection," *arXiv e-prints*, vol. 45, p. 190, 2020.

[33] L. Gan, R. Zhang, J. W. Grizzle, R. M. Eustice, and M. Ghaffari, "Bayesian spatial kernel smoothing for scalable dense semantic mapping," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 790–797, 2020.

[34] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," *IEEE Conference on Computer Vision & Pattern Recognition (CVPR)*, vol. 289, p. 190, 2012.

[35] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, and P. Tsui, "Scalability in perception for autonomous driving: Waymo open dataset," *Conference on Computer Vision & Pattern Recognition*, IEEE, vol. 47, p. 108, 2019.

[36] P. Sun, H. Kretzschmar, X. Dotiwalla et al., "Scalability in perception for autonomous driving: waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Seattle, WA, USA, June 2020.