The Visual Computer
https://doi.org/10.1007/s00371-024-03275-9

ORIGINAL ARTICLE

Camera calibration for the surround-view system: a benchmark and dataset

Leidong Qin[1] · Chunyu Lin[1] · Shujuan Huang[1] · Shangrong Yang[1] · Yao Zhao[1]

Accepted: 8 January 2024

Abstract
Surround-view system (SVS) is widely used in the advanced driver assistance system (ADAS). SVS uses four fish-eye lenses to monitor real-time scenes around the vehicle. However, accurate intrinsic and extrinsic parameter estimation is required for the proper functioning of the system. At present, the intrinsic calibration can be pipeline by utilizing checkerboard algorithm, while extrinsic calibration is still immature. Therefore, we proposed a specific calibration pipeline to estimate extrinsic parameters robustly. This scheme takes a driving sequence of four cameras as input. It firstly utilizes lane line to roughly estimate each camera pose. Considering the environmental condition differences in each camera, we separately select strategies from two methods to accurately estimate the extrinsic parameters. To achieve accurate estimates for both front and rear camera, we proposed a method that mutually iterating line detection and pose estimation. As for bilateral camera, we iteratively adjust the camera pose and position by minimizing texture and edge error between ground projections of adjacent cameras. After estimating the extrinsic parameters, the surround-view image can be synthesized by homography-based transformation. The proposed pipeline can robustly estimate the four SVS camera extrinsic parameters in real driving environments. In addition, to evaluate the proposed scheme, we build a surround-view fish-eye dataset, which contains 40 videos with 32,000 frames, acquired from different real traffic scenarios. All the frames in each video are manually labeled with lane annotation, with its GT extrinsic parameters. Moreover, this surround-view dataset could be used by other researchers to evaluate their performance. The dataset will be available soon.

Keywords Surround-view system · Advanced driver assistance system (ADAS) · Automatic extrinsic calibration

1 Introduction

Surround-view is increasingly popular in advanced driver assistance system (ADAS) [1, 2]. A four-camera SVS system is shown in Fig. 1. The surround-view generates a 360-degree image around the vehicle, providing the driver with a comprehensive view of the environment without any blind spots. In addition, the surround-view system is widely used in various automatic driving computer vision tasks, including traffic sign recognition parking space detection [3–5]. Typically, the onboard cameras are installed at the front, rear, left, and right sides of the vehicle, near the license plate. The images captured by onboard cameras are then used to generate auxiliary views of the vehicle environment.

Achieving seamless stitching of all captured images requires accurate calibration of the multi-camera system. Inaccurate calibration parameters can lead to a false perception of surroundings, which is dangerous for vehicle control. While intrinsic calibration techniques are mature, extrinsic parameters are varied due to the tiny camera motion. As the vehicle moves, the camera can slowly accumulate extrinsic changes from vibrations such as engine vibration, door opening and closing, extreme wind, and road bumps. Therefore, the pose and position of the onboard camera need to

✉ Chunyu Lin
cylin@bjtu.edu.cn

Leidong Qin
21120295@bjtu.edu.cn

Shujuan Huang
shujuanhuang@bjtu.edu.cn

Shangrong Yang
sr_yang@bjtu.edu.cn

Yao Zhao
yzhao@bjtu.edu.cn

[1] Institute of Information Science, Beijing Jiaotong University, Beijing Key Laboratory of Advanced Information Science and Network, Beijing 100044, China

Published online: 16 February 2024

be re-estimated to ensure stitching performance. In most commercial solutions, drivers have to rely on professional factories or workers for extrinsic calibration, which can be time-consuming and labor-intensive. Manufacturers have a demand for effective extrinsic calibration without human intervention. In recent years, there are some self-calibration schemes for various realistic scenarios [6–9]. However, the dataset of traffic scenarios with surround-view is still insufficient. To address this research gap, we collect a new dataset that consists of traffic scenes with lane lines and proposed a self-calibration pipeline. In summary, our contributions mainly consist of three aspects:

(1) An extrinsic calibration scheme is proposed, which utilizes lane line and ground texture. First, our method detects and filters lane lines nearby the vehicle. Based on geometric constraints, we makes a rough estimation for camera pose by utilizing the direction of lane lines. Second, we align the ground texture of adjacent cameras to achieve accurate extrinsic estimation.
(2) Our method corrects pose through mutual iterating of lane line projection constraints and lane re-detection. Rough lane detection results in inaccurate pose estimation. By projecting the frame to the rough ground plane with the rough pose, lane marking can be relocated more accurately. As a result, the pose can be iterated with updated lane marking. Our method is robust and can maintain sub-pixel accuracy for long-range lane marking detection.
(3) We propose a surround-view video dataset with ground truth (GT). The dataset is collected from various traffic scenarios with lane lines in different environmental conditions. It contains 40 sets of videos collected by fish-eye cameras. All video frames are manually annotated with high-quality lane points and camera extrinsic parameters.

The proposed method requires following assumptions as prerequisites:

(1) Intrinsic parameters and the position of the camera are already known.
(2) The road surface is flat and straight; besides, the driving direction is always parallel to the lane line. The angle between the front direction of the vehicle and the lane lines is supposed to be from -2 to 2 degree. In our calibration pipeline, the range of roll and yaw angle between front direction of the vehicle and the lane lines was supposed to be from -10 to 10 degrees. The range of pitch angle is from 20 to 90 degrees. For each camera, The intrinsic parameters (including distortion coefficients) are required, and the error of the x-coordinate should be within -0.2 and $+0.2$ ms. The error of the z-coordinate (the height of camera) should be within -0.2 and $+0.2$ ms.

SVS system is usually consist of four or six fish-eye cameras. Our paper is aimed at the four-camera system. The proposed method is fully automatic. In traffic scenarios, it can maintain stable estimation under various environmental conditions.

2 Related works

2.1 Surround-view datasets

The Woodscape dataset [10] is the first fish-eye dataset that includes about 10,000 images with instance-level semantic annotation. It was collected from the street road and parking lot. Notably, the images from the dataset are not continuous frame by frame and are selected periodically. The entire dataset were collected by the same vehicle, resulting in only one set of extrinsic parameters.

The Tongji surround-view dataset [11] is a large-scale campus scene collected by an electric car equipped with four cameras. It mainly composed of two parts: calibration site images and natural scenes. The original dataset contains 19,078 groups of fish-eye images with only one set of extrinsic parameters.

2.2 Lane detection

Lane markings are general elements in traffic scenes. Utilizing the geometric constraints, pose can be estimated by a pair of lane marking in frames.

Early lane detection methods were primarily based on handcrafted features, such as color [12, 13], edge [14, 15], and texture [16]. Recently, the methods of deep learning [17, 18] have significantly improved lane detection performance. In VPGNet [19], vanishing points were utilized to conduct multitask network training for lane detection. SCNN [20] specifically considers the thin-and-long shape of lanes by passing messages between adjacent rows and columns at a feature layer. SAD [21] and inter-region affinity KD [22] further adopt the knowledge distillation to improve lane detection. PolyLaneNet [23] formulates the instance-level lane detection as a polynomial regression problem, and UFSA [24] provides ultra-fast lane detection by dividing the image into grids and scanning grids for lane locations.

Due to the limitation of computation performance, onboard processors cannot apply to deep leaning tasks. To achieve a more accurately extrinsic calibration, we propose a method that provide sub-pixel lane detection in the middle distance.

2.3 Pattern-based calibration methods

A pattern-based approach estimates camera parameters using special patterns, including corners, circles, or lines. Since a pattern-based approach uses precisely drawn patterns with

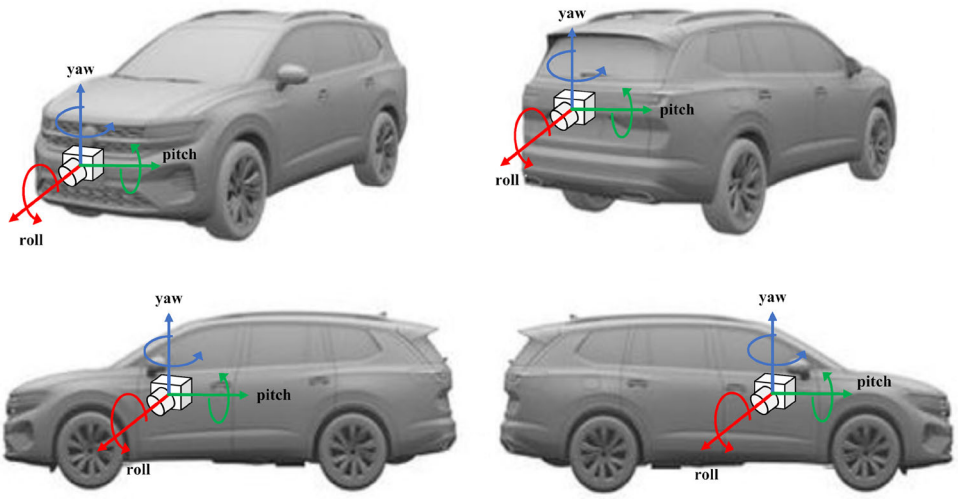

Fig. 1 A four-camera SVS system

known configurations, it is possible to accurately estimate the camera parameters, making it suitable for accurate calibration for a surround-view camera system. Some pattern-based calibration methods place calibration patterns in the overlapped fields of the cameras [25, 26]. Methods in [27] use factorization-based methods by placing calibration patterns between adjacent groups of cameras. However, these calibration method requires a checkerboard, making them suitable for the factory but not for a private use, as they involve lots of human interventions.

2.4 Self-calibration methods

In [28], Zhao et al. first detected multiple vanishing points of lane markings on the road via the weighted least-squares method. With the estimated vanishing points, the pose of the multi-camera system relative to the world coordinate system was solved. Choi et al. [6] also designed a lane line-based extrinsic self-calibration pipeline for the surround-view case, in which the SVS was calibrated by aligning lane markings across images of adjacent cameras. However, this method relies on high accuracy of lane marking detection and world coordinate of cameras.

There are some self-calibration schemes that are applicable to the SVS, including Liu et al.'s method [7] and Zhang et al.'s [8, 9]. They deeply dissected the online extrinsic correction problem and offered effective solutions. Liu et al. [7] proposed two models, the "Ground Model" and the "Ground-Camera Model," which correct extrinsic by minimizing photometric errors with the steepest descent [7]. Zhang et al. [8] designed a novel model, the bi-camera model, to construct the least-squares errors [29] on the imaging planes of two adjacent cameras and then optimize camera poses by the LM (Levenberg–Marquardt) algorithm [30]. And they further improved their work in [9] by utilizing multiple frames selected in a local window rather than a single frame to build the overall error, thus improving the system's robustness. However, the above three studies [7–9] focused on online correction rather than calibration, and required a rough initial extrinsic as the input, which limited the application.

3 Proposed method

Figure 2 shows the flowchart of the proposed method. First, the method detects lane markings from images captured by four cameras. After locating the lane markings, the method calibrates the extrinsic parameter the of the front and rear cameras. Then it processes the left and right cameras using the estimated of the front and rear cameras. In the following section, further details of each step will be introduced.

3.1 Lane detection and selection

We utilize Kannala–Brandt's distortion model to undistort fish-eye images and Canny's edge to detect lines. The proposed method employs Hough line detection on the undistorted image to detect lane marking. Since the detection results may include falsely detected lane markings, a series of measures are taken to filter the required lane markings. First, we used a vanishing point (VP) to reject the outer lines. The VP can be calculated by [31]. The Lane marking whose perpendicular distance to the vanishing point is less than a specified threshold are preserved. The overall aim is to reject the lines that are far away from the vanishing point. Considering the left and right cameras cannot capture the lane markings below the vehicle, a RoI (region of interest) is set to reject such markings. Since the front and rear cameras required only one pair of lane marking, the pair of lane marking nearest to the camera is selected for estimation.

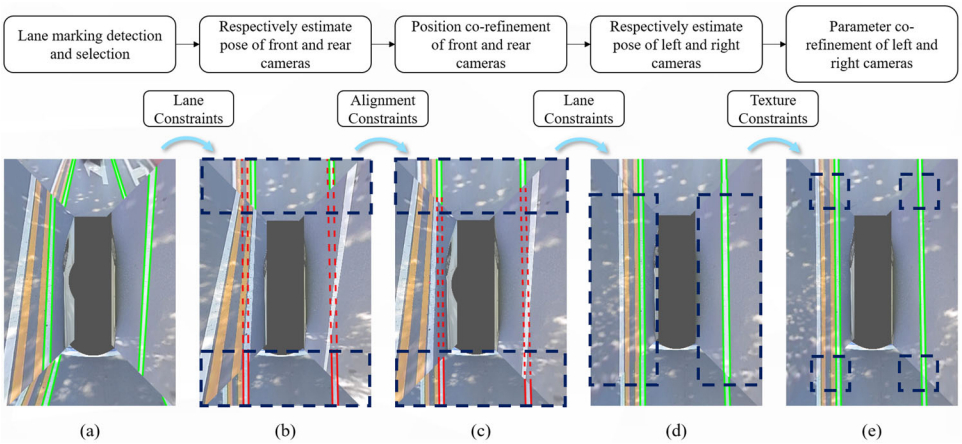

Fig. 2 Flowchart of the proposed method. **a**–**e**, respectively, show the surround-view generated by extrinsic parameters of each step

As for the left and right cameras, it only requires one lane marking.

3.2 Front and rear cameras Calibration

The proposed method uses geometric constraints of lane lines in the projection plane to iteratively estimate camera pose.

We propose an algorithm to iteratively estimate camera pose. The algorithm consisted of three sub-step, and each sub-step separately estimates pitch, yaw and roll angles. Assuming P' is the point in the undistorted image and the corresponding point in the surround-view image is P, and P can be transformed by the homography H as follows,

$$P^{'} = H^{-T}P = \left(K_2 R K_1^{-1}\right)^{-T} P, \tag{1}$$

where K_1 is a 3x3 matrix representing the real camera's intrinsic parameters and K_2 is a 3x3 matrix representing the intrinsic parameters of the virtual camera on the surround-view plane. R is the rotation matrix. $I^{'}$ is the point on the surround-view plane.

We design three kinds of cost function to estimate pitch, roll and yaw angle separately. Figure 3 illustrates their functionality. Our algorithm first minimizes the cost c_φ by updating rotation matrix iteratively with,

$$R_1^{'} = R_\varphi R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} R, \tag{2}$$

where φ is pitch angle.

The pitch angle is estimated by maximizing the cost c_φ as follows,

$$\begin{aligned} c_\varphi &== \left\| \overline{a_1 b_1} + \overline{a_2 b_2} + \overline{a_3 b_3} + \overline{a_4 b_4} \right\|, \\ where\ \overline{a_i b_i} &= \overrightarrow{a_i b_i} / \left\| \overrightarrow{a_i b_i} \right\| \ (i = 1, 2, 3, 4). \end{aligned} \tag{3}$$

As Fig. 3 shows, point a_i and b_i (i=1,2,3,4) correspond to the end points of the extracted markings in the surround-view image. a_1, a_2, a_3, and a_4 are the intersection point with line $y = 0$. $\overline{a_i b_i}$ (i=1,2,3,4) is the unitized direction vector of $\overrightarrow{a_i b_i}$, i.e., the direction vector of the lane marking. In this sub-step, we can approach the unique extreme value of c_φ by iterating calculating rotation matrix with Eq. (2).

In the second sub-step, the iterative strategy is updating rotation matrix with,

$$R_2^{'} = R_\omega R = \begin{bmatrix} \cos\omega & 0 & -\sin\omega \\ 0 & 1 & 0 \\ \sin\omega & 0 & \cos\omega \end{bmatrix} R, \tag{4}$$

where ω is yaw angle. In most cases, the width of the right lane line is equal to the left lane line. As a result, the yaw angle is estimated by minimizing the cost c_ω by,

$$c_\omega = \left| |a_1 a_2| - |a_3 a_4| \right|. \tag{5}$$

where $|a_1 a_2|$ corresponds to length of $a_1 a_2$ and $|a_3 a_4|$ corresponds to length of $a_3 a_4$. Similarly, we can approach the only extreme value of c_ω by iterating the rotation matrix with Eq. (4).

In the third sub-step, the iterative strategy can be expressed as,

$$R_3^{'} = R R_\theta = R \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

where θ is roll angle. Since the direction of the vehicle parallels to the lane marking, the projection line of the lane marking also parallels to the axis of the projection plane. The roll angle is estimated by minimizing the cost, c_θ, as,

$$c_\theta = \sum_{i=1}^{n} \left| \cos\left\langle \widehat{a_i b_i}, \overrightarrow{e_1} \right\rangle \right|, \tag{7}$$

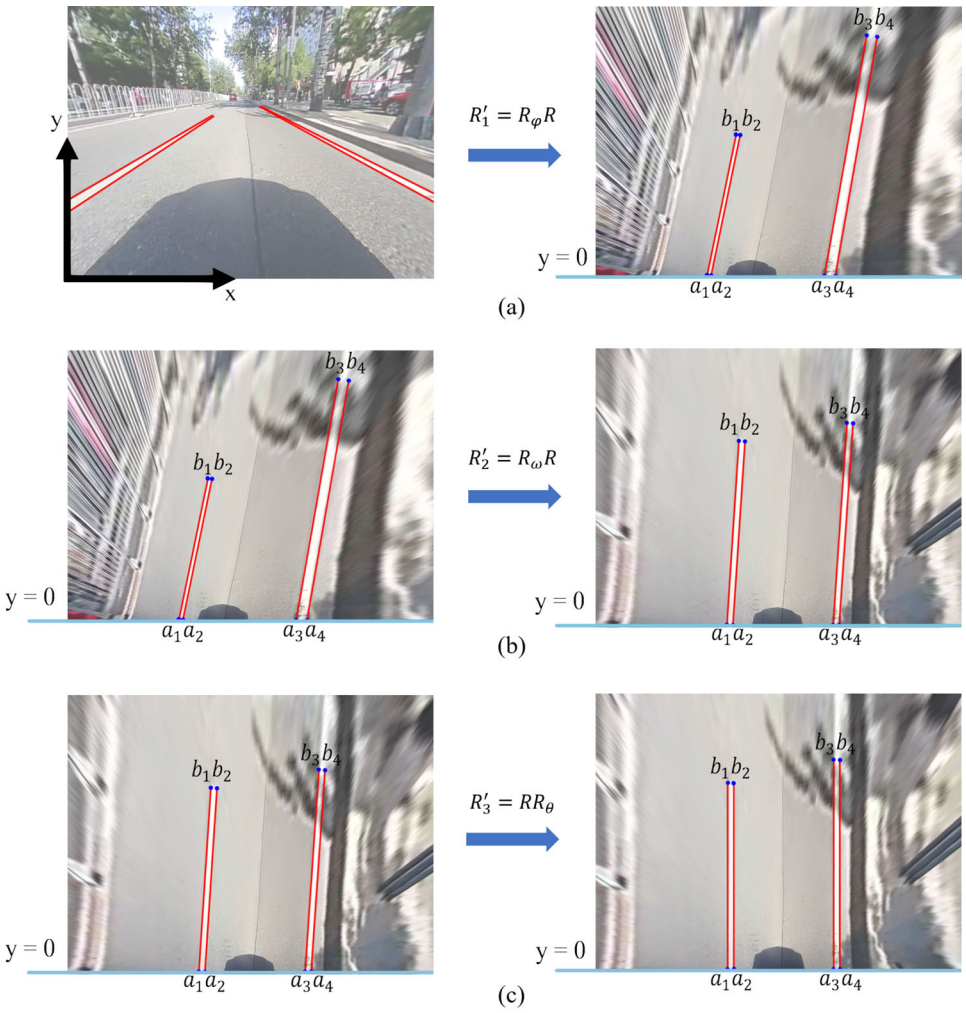

Fig. 3 Algorithm of pose estimation for front and rear camera. **a** Sub-step of approaching pitch angle. **b** Sub-step of approaching yaw angle. **c** Sub-step of approaching roll angle

where $\overrightarrow{e_1}$ is the vector equals [1, 0] in the surround-view plane and n equals 4. Similarly, we can approach the only extreme value of c_θ by iterating the rotation matrix with Eq. (6).

Overall, the motivations for each of the "cost functions" are not well described. Eventually, it becomes clear that the first makes all the lines "point the same direction," the second normalizes the line widths, and the third makes the lines point "up" in the image. The total algorithm is shown in Algorithm 1.

In the sub-step, phi, omega, and theta will be updated by determining the gradient direction of cost function. Ignoring the error of line detection, the first derivative in c_φ of φ is monotonic, so as to the derivative in c_ω of ω, derivative in c_θ of θ. Each sub-step iteratively approaches the extreme value of the cost functions, and camera pose is able to be gradually estimated.

Based on these cost functions, camera pose can be roughly estimated. This method can achieve an accurate camera pose estimation under precise lane marking detection. Consider-

Algorithm 1 Single Camera Pose Correction

Require: P, R, K_1, K_2
Ensure: R
Function
$c_{angle} \leftarrow F(R, P, K_1, K_2)$ angle $\in \{\varphi, \omega, \theta\}$

Initialize R, P, K_1, K_2
while $iter < iter\_max$ **do**
  while $c_\varphi < c_{\varphi,last}$ **do**
    $R \leftarrow R_\varphi R$
    $c_\varphi \leftarrow F(R, P, K_1, K_2)$
  end while
  while $c_\omega < c_{\omega,last}$ **do**
    $R \leftarrow R_\omega R$
    $c_\omega \leftarrow F(R, P, K_1, K_2)$
  end while
  while $c_\theta < c_{\theta,last}$ **do**
    $R \leftarrow RR_\theta$
    $c_\theta \leftarrow F(R, P, K_1, K_2)$
  end while
  $iter \leftarrow iter + 1$
end while
return R

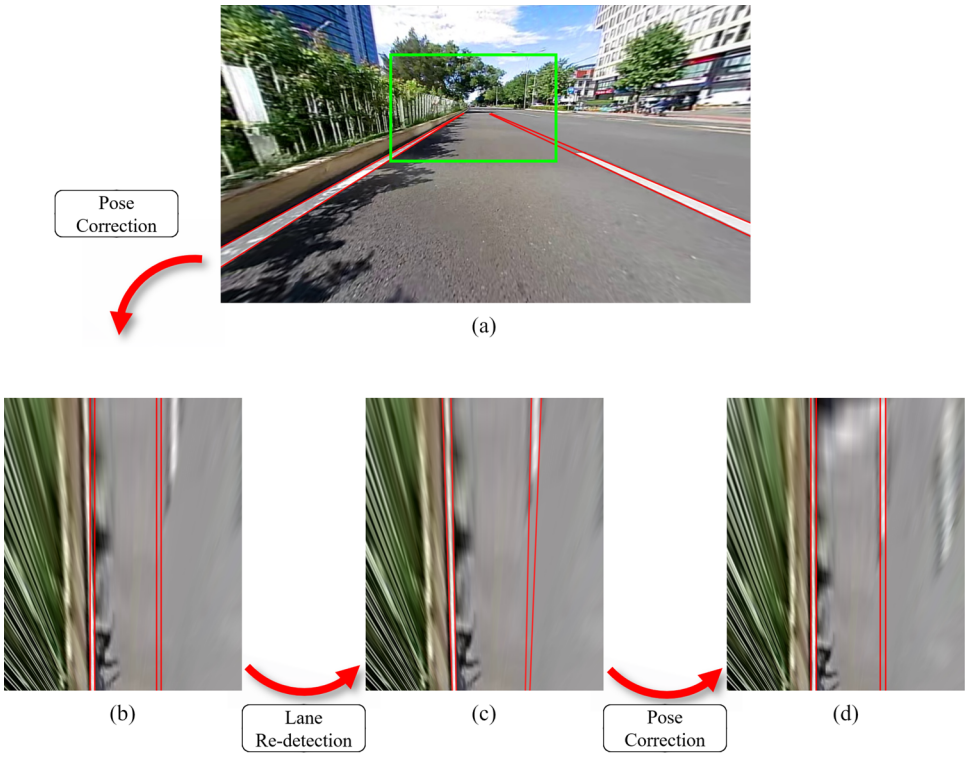

Fig. 4 **a** Straight lane line found in the undistorted image. **b** A rough projection after a rough calibration of the front camera. **c** Lane marking correction in the rough projection images. **d** After pose re-correction

ing the complex traffic scenario in reality, the line detection needs to be further refined in the subsequent step.

3.3 Lane marking refinement

The accuracy of lane marking detection is crucial for calibration. For the left and right cameras, since lane marking is quite close to the vehicle, the detection method mentioned in Sect. 3.4 is sufficient to meet the accuracy requirements. However, in environmental conditions, such as extreme light and fragmentary lane marking, the above detection method yields unsatisfactory result for the front and rear cameras.

As shown in Fig. 4b, the method is unstable in edge detection when lane marking is far away from the camera. This problem can be solved under the projection plane. After the rough calibration based on the lane line cost function, the preliminary estimated pose is utilized to generate a projection image based on homography. As illustrated in Fig. 4c, in the projection image, lane marking is accurately estimated by line detection, such as Hough line detection. The originally selected lane line is updated by searching for the nearest detected line. After correcting lane marking, the front and rear cameras are re-calibrated base on the lane marking cost function.

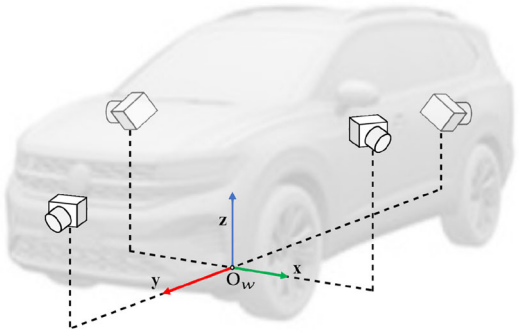

Fig. 5 World coordinate system of SVS. The y-axis direction is the direction of vehicle. The original point is the cross center of four cameras on the ground plane

3.4 Co-refinement of front and rear camera

As shown in Fig. 5, after the pose of front and rear cameras are, respectively, estimated, the x coordinate of the front and rear cameras in the world coordinate system can be optimized.

As shown in Fig. 6, in order to robustly generate the surround-view images, camera position error and pose error need to be taken into account. Therefore, the front and rear

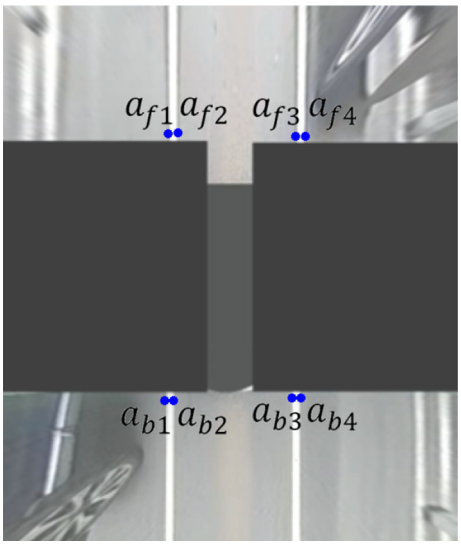

Fig. 6 Explanations of c_{fb} cost functions. The image of front and rear camera will be projected to the surround-view plane. The positions will be approached by aligning lane lines

cameras will be simultaneously refined by minimizing the cost, c_{fb}, as

$$c_{fb} = c_{\omega,\text{front}} + c_{\omega,\text{behind}} + \sum_{i=1}^{n} \left| x_{fi} - x_{bi} \right|, \tag{8}$$

where x_{fi}, x_{bi} is x coordinate of a_{fi} and a_{bi} (i=1,2,3,4). The cost function c_{fb} reflected the relative aligned error of front and rear camera. Therefore, we only optimizing the x coordinate of rear camera in the world coordinate system.

3.5 Calibrate left and right cameras

The calibration of the left and right cameras is similar to that of the front and rear cameras.

As Fig. 7 shows, the method firstly estimates the yaw (Fig. 7a–b) and roll angles (Fig. 7b, c) of the side camera. The yaw and roll angles of the side camera are similarly estimated by the method mentioned in Sect. 3.1.

As Fig. 7 shows, point a_{s1}, a_{s2}, b_{s1}, and b_{s2}, respectively, correspond to the end points of the extracted markings in the surround-view image, i.e., the intersection point with line $y = 0$ and line $y = y_0$.

The yaw angle is estimated by minimizing the cost $c_{\omega,\text{side}}$ by,

$$c_{\omega,\text{side}} = \left| |a_{s1}a_{s2}| - |b_{s1}b_{s2}| \right|, \tag{9}$$

where $|a_{s1}a_{s2}|$ corresponds to length of $a_{s1}a_{s2}$ and $|b_{s1}b_{s2}|$ corresponds to length of $b_{s1}b_{s2}$. Similarly, we can approach the only extreme value of $c_{\omega,\text{side}}$ by iterating the rotation matrix with Eq. (4).

The roll angle is estimated by maximizing the cost, $c_{\theta,\text{side}}$ as,

$$c_{\theta,\text{side}} = \cos\left\langle \widehat{a_{s1}b_{s1}}, \overrightarrow{e_1} \right\rangle + \cos\left\langle \widehat{a_{s2}b_{s2}}, \overrightarrow{e_1} \right\rangle, \tag{10}$$

where $\widehat{a_{s1}b_{s1}}$ and $\widehat{a_{s2}b_{s2}}$, respectively, correspond to the unitized direction vector of $\overrightarrow{a_{s1}b_{s1}}$ and $\overrightarrow{a_{s2}b_{s2}}$, i.e., the direction vector of the lane marking. $\overrightarrow{e_1}$ is the vector equals [1, 0] in the surround-view plane. Similarly, we can approach the only extreme value of $c_{\theta,\text{side}}$ by iterating the rotation matrix with Eq. (6).

The optimization of pitch angle relies on the calibration results of the front and rear cameras. Once the estimation of front and rear camera is completed, the width of lane marking can be simply calculated. The pitch angle is estimated by minimizing the cost $c_{\varphi,\text{side}}$ by,

$$c_{\varphi,\text{side}} = \left| |a_{s1}a_{s2}| - d_l \right|, \tag{11}$$

where $|a_{s1}a_{s2}|$ corresponds to length of $a_{s1}a_{s2}$ and d_l corresponds to the width of lane marking calculated in previous the section. Similarly, we can approach the only extreme value of $c(\varphi, \text{side})$ by iterating the rotation matrix with Eq. (2).

However, due to the limited lane width accuracy of front and rear calibration, pitch estimation is unstable. This represents a limitation of calibration based on lane marking. As a result, a texture-based method is proposed to robustly estimate the pitch of the side camera.

3.6 Co-refinement of left and right camera

In this step, the pipeline estimates pitch and position by contracting the pixel color error of RoI in the projection plane. Figure 8 provides an example of how adjacent cameras are processed by RoI texture. The texture alignment error considers the color difference and edge gradient of pixels. Before calculating the pixel color error, the projected images of adjacent cameras need to be normalized on three color channels, which reduces the impact of adjacent cameras under different lighting conditions.

Assume that the boxed projection map of two adjacent cameras is I_2 and I_1. Normalization from I_2 to I_1 is,

$$I_2^{'} = \frac{\sigma_1}{\sigma_2}(I_2 - \mu_2) + \mu_1, \tag{12}$$

where σ_1 and μ_1 are standard deviation and mean value of I_1, and σ_2 and μ_2 are standard deviation and mean value of I_2.

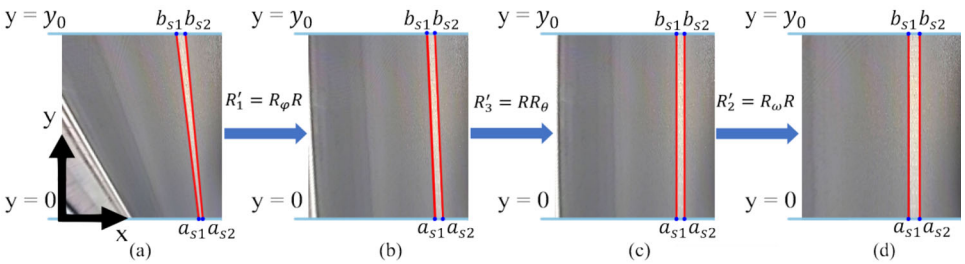

Fig. 7 Algorithm of pose estimation for front and rear camera. **a** Original surround-view. **b** Sub-step of approaching yaw angle. **c** Sub-step of approaching roll angle. **d** Sub-step of approaching pitch angle

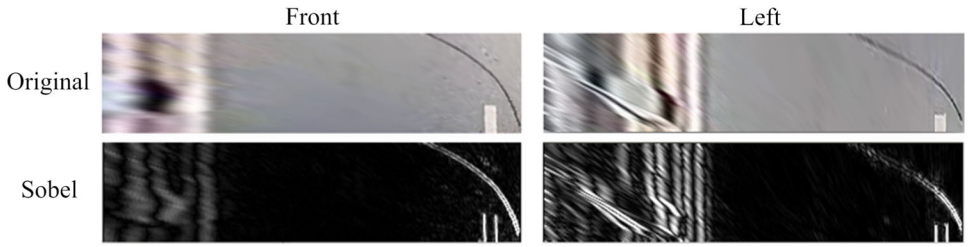

Fig. 8 Comparison of the adjacent camera projection. The pitch angle and position of bilateral camera can be estimated by traversal algorithm

The pitch angle and position can be estimated by minimizing the cost, c_{text} as,

$$c_{\text{text}} = \frac{1}{hw}\sum_{i=1}^{h}\sum_{j=1}^{w} G_y(i,j)(I_1(i,j) - I_2^{'}(i,j))^2, \tag{13}$$

where h is the height of the RoI image. w is the width of the image, G_y is the longitudinal edge gradient of the pixel, and G_y is calculated by Sobel operator. Considering the characteristics of the fish-eye lens, the clarity of projection images of two adjacent cameras is different. Therefore, G_y is calculated by the camera whose RoI is closer to its principal point. For our dataset, we select left and right camera to calculate G_y. Due to the rough estimation of the previous step, the traversal range has been reduced. The pitch angle and position can be approached by traversal algorithm and the time cost is acceptable.

3.7 Parameter summarizing in sequence

Sections 3.1–3.6 introduce the calibration at a certain moment. To apply the calibration to a long image sequence, the camera parameters are repeatedly estimated whenever a sufficient number of lane markings are collected. Using this approach, multiple parameter sets are obtained. A parameter set includes 12 camera angles (three angles for each camera) and can be represented as a 12-dimensional vector. To select the most appropriate parameter set and prevent overfitting from a specific place, this paper uses the mean parameter set $a^{'}$ of multiple parameter sets (a_1,a_2,. . .,a_N) as,

$$a^{'} = \min_{b\in\{a_1,a_2,\ldots,a_N\}} \sum_{i=1}^{N} k_i d(a_i, b), \tag{14}$$

where N is the number of parameter sets and $d(a_i, b)$ indicates the Euclidean distance between two parameter sets (a_i and b). k_i is the confidence as,

$$k_i = \frac{c_{i,text}}{\sum_{j=1}^{N} c_{j,text}}, i = 1, 2, 3, \ldots, N, \tag{15}$$

where $c_{(i,text)}$ is the texture cost function of a time stamp.

4 Our dataset

Our dataset was collected by four fish-eye cameras mounted on cars. The dataset consists of 40 groups of videos. Each group of videos includes four video streams, front, back, left, and right. The frame rate is 25 fps.

Our dataset is captured by multiple sets of fish-eye cameras with different camera parameters, and the camera system is installed on several vehicles for collection. As a result, the GT of the extrinsic parameters in the dataset is various, as illustrated in Fig. 9.

The frames of each video share the same set of extrinsic parameters and GT, which are estimated using a chessboard.

The type of cameras is 2-million-pixel high-definition panoramic waterproof lens from Shenzhen Zhongke Nanguang Technology Co., Ltd. The FOV of the camera is 190 degrees, and the Image sensor type is IMX307 from Sony.

Lane lines, as well as the vehicle intrinsic and extrinsic parameters of each frame, are stored in the corresponding JSON file. Besides, all lane line annotations are manual. Each lane line is represented by two sets of edge points and is marked with the its type, which includes single white solid line, single white dotted line, single yellow solid line, single yellow dotted line, double white solid line, double yellow

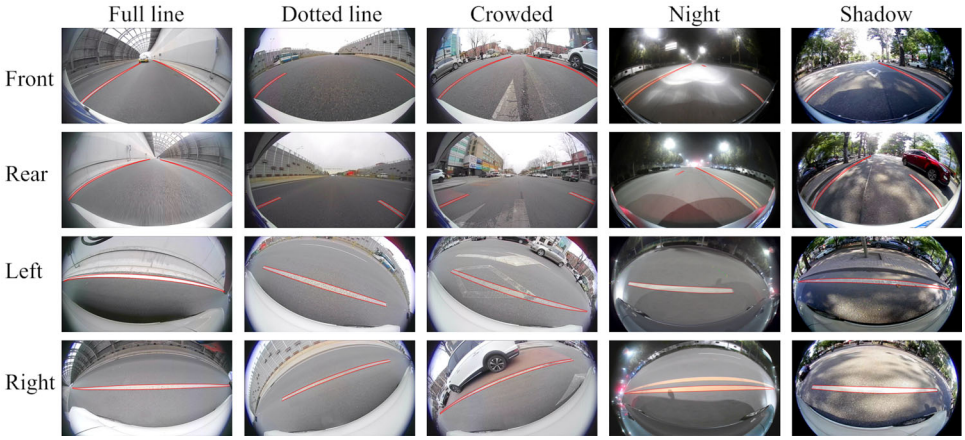

Fig. 9 The proposed video surround-view dataset contains different real traffic scenarios and provides the lane annotations

solid line, double yellow dotted line, double white solid dotted line, double white solid line, and white yellow solid line.

5 Experiments

In real-world traffic scenario, when driving in the dotted lane, four cameras cannot capture the valid lane at the same time. Our dataset includes the solution. We reserve the lane lines that appeared in a short period. This is because driving direction can be assumed to be straight for a short period when diving to a straight lane.

We tested the pipeline with our dataset. We select clips from various environments as input. Partial results are shown in Fig. 10. As can be seen from the results, our method has stable performance in various traffic scenarios.

In order to verify the effectiveness of the proposed method, we reproduce the scheme of Choi et al. [6], OCPO [7] and WESNet [11] for comparison. Each method initializes the position parameter with 5-centimeter error from GT in each world axis. Table 1 shows the result. The average camera angle error equals the average of three Euler angle. The average position error equals Euclidean distance between estimation and ground truth in the world coordinate system.

We provide detailed evaluation results of the calibration methods for each type of scenario. Table 1 shows the average angle error and average position error. The angle unit and length unit are degree and meter.

The best results are highlighted in bold. Table 1 shows that compared with all counterparts, our method shows an overwhelming performance on all data groups. Choi's method utilized lane marking and only estimated the camera angles. For the left and right camera, Choi's method performs unstably due to the error of camera height. In contrast, our method estimates height and pitch together by constraining the width of projected lane marking and aligning the projected texture. Specifically, OCPO only takes the photometric error as the guidance to correct camera poses. However, in low-texture environment, the photometric error is mainly affected by noise rather than inaccurate poses of cameras. In this case, OCPO can yield extrinsic parameters with low photometric errors, but their accuracy cannot be guaranteed.WESNet follows a weakly supervised learning framework. In the fine-tuning stage, WESNet utilizes photometric error to train the network, which will result in similar errors to OCPO. In conclusion, the excellent accuracy of extrinsic parameter estimation and the generalization capability of our method has been nicely demonstrated.

In Table 2, we conducted the ablation experiment to prove the effectiveness of each component we propose. The lane marking re-detection improves extrinsic estimation by enhancing lane marking detection accuracy of each camera. The texture procedure also contributes to calibration.

Robustness to intrinsic disturbance In order to evaluate the robustness of our method of the accuracy of intrinsic parameters, we empirically alter the intrinsic parameters of each camera. The disturbance can be represented as an intrinsic disturbance factor d. We added it to the focal length of the camera. The disturbed intrinsic matrix $K^d_{C_i}$ of camera C_i can be expressed as

$$K^d_{C_i} = \begin{bmatrix} f_x + d & 0 & c_x \\ 0 & f_y + d & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{16}$$

We test the mentioned method and our scheme with different d's setting and recorded the result. The result is shown in Fig. 11. It illustrates that our method is able to calculate a more accurate result as long as the intrinsic disturbance is higher. The reason is that the intrinsic parameter are disturbed, and the pixel is farther from principal point and yields higher distortion error. Our scheme utilizes information about lane as a strong constraint. Lane lines mostly distributed in the center of fish-eye image while the RoI distributed at the side of fish-eye image. Therefore, lane-based calibration is more robust than texture-based calibration. Besides, based on

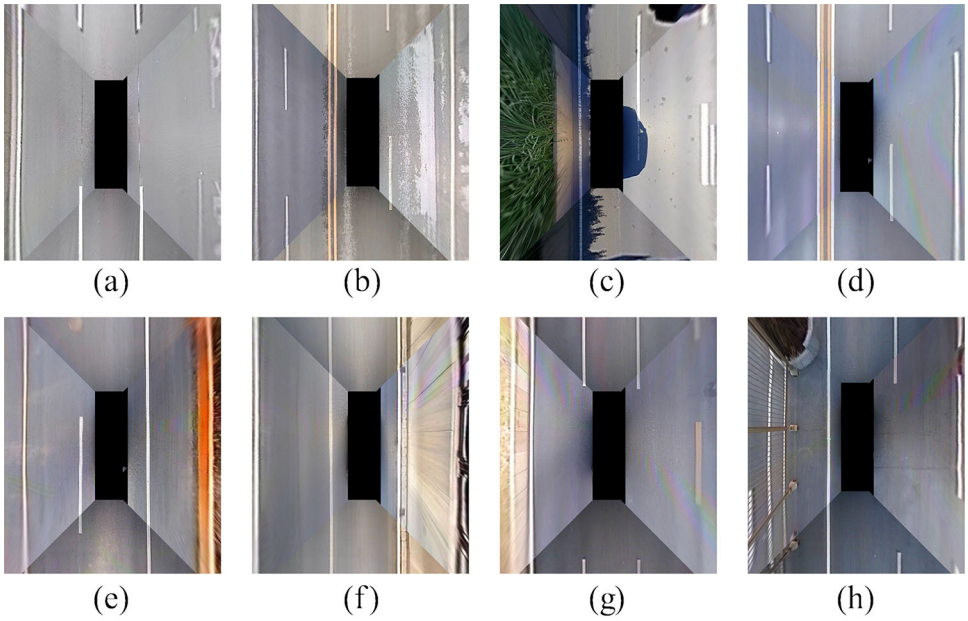

Fig. 10 Examples of representative traffic scenarios. Note that there is a wall or a footstep on the side of the vehicle in **c**, **e**, **f**, and **h**. The 3D point that is not on the ground plane will be transformed incorrectly

Table 1 Average angle error (degree) and average position error (meter) for different methods

Group	Method	Front camera		Rear camera		Left camera		Right camera	
		Angle	Position	Angle	Position	Angle	Position	Angle	Position
Full line	Choi et al. [6]	0.261	none	0.279	none	1.542	none	2.964	none
	OCPO [7]	0.631	0.019	0.504	0.021	0.702	0.049	0.737	0.037
	WESNet [11]	0.369	0.030	0.265	0.019	0.424	0.029	0.465	0.021
	Ours[1]	**0.136**	**0.003**	**0.150**	**0.006**	**0.213**	**0.009**	**0.253**	**0.009**
Dotted line	Choi et al. [6]	0.337	none	0.330	none	2.974	none	3.513	none
	OCPO [7]	0.834	0.029	0.688	0.050	1.038	0.088	1.229	0.051
	WESNet [11]	0.501	0.032	0.549	0.038	0.870	0.040	0.989	0.024
	Ours[2]	**0.153**	**0.004**	**0.233**	**0.005**	**0.206**	**0.010**	**0.349**	**0.011**
Crowded	Choi et al. [6]	0.289	none	0.291	none	3.321	none	3.950	none
	OCPO [7]	0.977	0.035	0.840	0.074	1.481	0.122	1.916	0.088
	WESNet [11]	0.884	0.036	0.958	0.057	1.833	0.099	1.259	0.087
	Ours[3]	**0.326**	**0.012**	**0.398**	**0.011**	**0.608**	**0.018**	**0.631**	**0.017**
Night	Choi et al. [6]	1.856	none	2.769	none	3.229	none	4.125	none
	OCPO [7]	1.109	0.057	1.889	0.110	2.538	0.139	1.638	0.117
	WESNet [11]	0.835	0.087	1.594	0.084	2.217	0.108	1.526	0.062
	Ours[4]	**0.781**	**0.062**	**0.960**	**0.022**	**1.013**	**0.080**	**1.097**	**0.055**
Shadow	Choi et al. [6]	0.308	none	0.323	none	3.591	none	3.911	none
	OCPO [7]	0.529	0.031	0.497	0.063	1.517	0.109	1.270	0.077
	WESNet [11]	0.448	0.033	0.341	0.036	1.381	0.082	1.053	0.043
	Ours[5]	**0.257**	**0.011**	**0.146**	**0.009**	**0.534**	**0.027**	**0.398**	**0.021**
Average	Choi et al. [6]	0.371	none	0.424	none	2.710	none	3.484	none
	OCPO [7]	0.763	0.025	0.687	0.049	1.244	0.086	1.217	0.060
	WESNet [11]	0.486	0.035	0.513	0.037	0.965	0.053	0.887	0.030
	Ours[6]	**0.228**	**0.009**	**0.262**	**0.008**	**0.376**	**0.017**	**0.398**	**0.015**

Table 2 Result of ablation experiment

Method	Front camera		Rear camera		Left camera		Right camera	
	Angle (degree)	Position (meter)	Angle	Position	Angle	Position	Angle	Position
Ours[a]	0.367	0.023	0.485	0.028	0.648	0.034	0.674	0.029
Ours[b]	0.228	0.009	0.262	0.008	0.554	0.076	0.597	0.058
Ours	0.228	0.009	0.262	0.008	0.376	0.017	0.398	0.015

[a] without lane marking re-detection
[b] without texture procedure

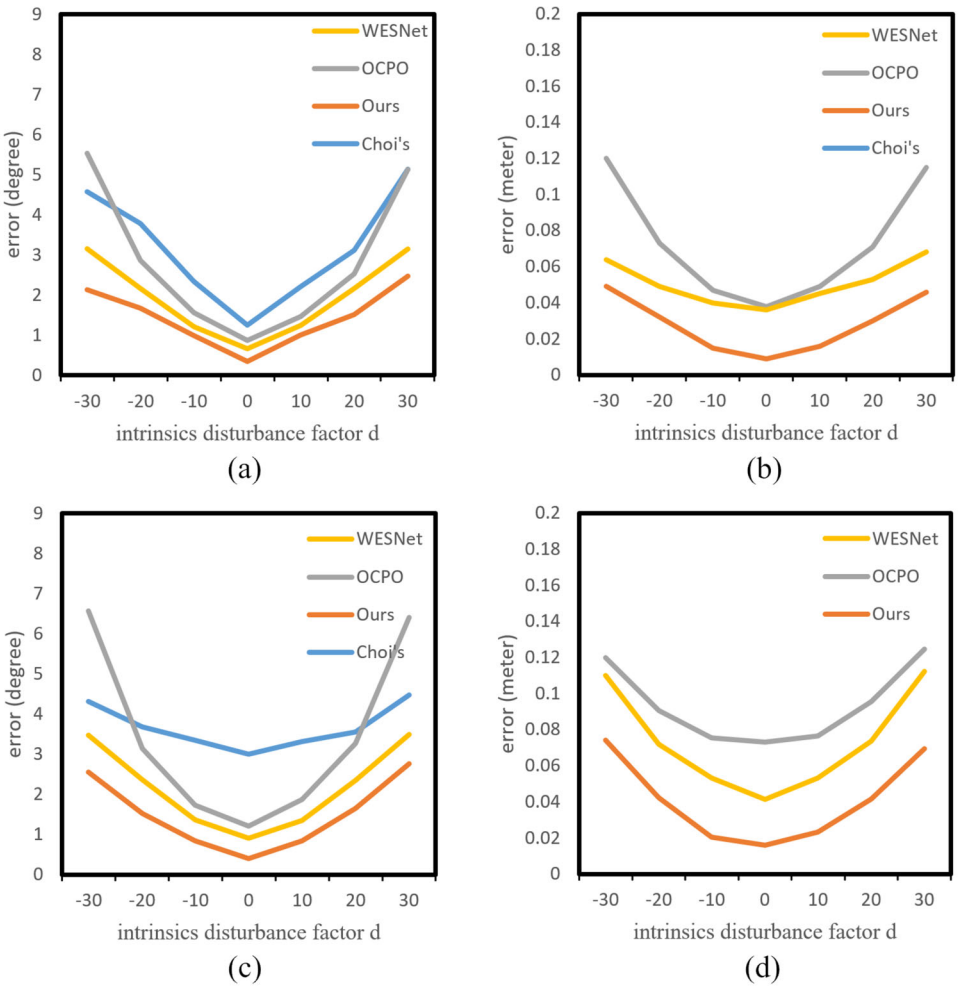

Fig. 11 Angle error and position error under different disturbance factor d's settings. **a** and **b**, respectively, show average angle error and position error of front and rear camera. **c** and **d**, respectively, show the error of left and right camera

our experience, the camera's focal length variation caused by the natural collisions or bumps is generally less than 5 pixels. Therefore, it can be concluded that our method is robust to the variations in intrinsic parameters.

Execution Time Table 3 shows the execution time for four main modules of the proposed method. These times were measured on an Intel Core i7-10875 CPU using only a single core that has 2.30 GHz. Note that among four modules the proposed method, only the lane marking detection module is required to be processed in real time. The other three modules need to be processed only once after a sufficient number of lane markings are gathered. In Table 3, the lane marking detection module requires 29.74 ms to process four images acquired from four cameras of the AVM system, which means that this module can process more than 30 frames per second in real time. The other three modules require a total execution time of 315.54 ms. Since those modules need to be processed only once after gathering lane markings, their execution times do not hinder the proposed method from operating in real time.

Table 3 Execution time

Module	Time(ms)
Lane marking detection	29.74
Lane marking selection	3.89
Parameter estimation	311.63
Parameter selection	0.02

6 Conclusion

In this paper, a new dataset and practical method are proposed to calibrate the camera attitude using lane markings and textures. To facilitate the research on self-calibration for the surround-view system, we collected a new dataset with high-quality lane annotation across all the frames. The proposed method offers several advantages. First, it is suitable for the natural driving situation of vehicles, whether the vehicle is moving at low speed or high speed. Second, the method uses the corresponding lane lines from adjacent camera images to relate each camera, generating a seamless aerial view of vehicles. The method is evaluated using image sequences captured under various actual driving conditions, and the results demonstrate good real-time performance.

The experiment uses a four-camera system. Therefore, this calibration method is also applicable to the six-camera system with six or more cameras. In the six-camera system, there will be two cameras at the front and two cameras at the rear of the vehicle when the vehicle is driving in the lane. Lane lines of two sides can be capture simultaneously. By treating the camera that captures two lane lines are as the "front and rear views angles" in the calibration process, and the camera that captures one side lane line as the "left and right views angles," this method can be used for camera calibration and circular projection splicing.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No.62172032, No. 62372036)

Data Availability The datasets associated with the current study are available upon reasonable request from the corresponding author.

Declarations

Conflict of interest The authors certify that there are no actual or potential conflicts of interest in relation to this article.

References

1. Baftiu, I., Pajaziti, A., Cheok, K.C.: Multi-mode surround view for ADAS vehicles. In: 2016 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS), pp. 190–193 (2016)
2. Gao, Y., Lin, C., Zhao, Y., Wang, X., Wei, S., Huang, Q.: 3-D surround view for advanced driver assistance systems. IEEE Trans. Intell. Transp. Syst. **19**(1), 320–328 (2018)
3. Chen, Y., Xiang, Z., Du, W.: Improving lane detection with adaptive homography prediction. Vis. Comput. **39**(2), 581–595 (2023)
4. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: BEVFormer: learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX, pp. 1–18 (2022)
5. Ma, Y., Liu, Y., Zhang, L., Cao, Y., Guo, S., Li, H.: Research review on parking space detection method. Symmetry **13**(1), 128 (2021)
6. Choi, K., Jung, H.G., Suhr, J.K.: Automatic calibration of an around view monitor system exploiting lane markings. Sensors **18**(8), 2956 (2018)
7. Liu, X., Zhang, L., Shen, Y., Zhang, S., Zhao, S.: Online camera pose optimization for the surround-view system. In: Proceedings of the 27th ACM International Conference on Multimedia, pp. 383–391 (2019)
8. Zhang, T., Zhang, L., Shen, Y., Ma, Y., Zhao, S., Zhou, Y.: Oecs: Towards online extrinsics correction for the surround-view system. In: 2020 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6 (2020)
9. Zhang, T., Zhao, N., Shen, Y., Shao, X., Zhang, L., Zhou, Y.: ROECS: a robust semi-direct pipeline towards online extrinsics correction of the surround-view system. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 3153–3161 (2021)
10. Yogamani, S., Hughes, C., Horgan, J., Sistu, G., Varley, P., O'Dea, D., Uricár, M., Milz, S., Simon, M., Amende, K., et al.: WoodScape: a multi-task, multi-camera fisheye dataset for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9308–9318 (2019)
11. Chen, Y., Zhang, L., Shen, Y., Zhao, B.N., Zhou, Y.: Extrinsic self-calibration of the surround-view system: a weakly supervised approach. IEEE Trans. Multimed. (2022). https://doi.org/10.1109/TMM.2022.3144889
12. Ma, C., Xie, M.: A method for lane detection based on color clustering. In: 2010 Third International Conference on Knowledge Discovery and Data Mining. pp. 200–203 (2010)
13. Wang, J., Mei, T., Kong, B., Wei., H.: An approach of lane detection based on inverse perspective mapping. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 35–38 (2014). https://doi.org/10.1109/ITSC.2014.6957662
14. Maya, P., Tharini, C.: Performance analysis of lane detection algorithm using partial Hough transform. In: Maya, P., Tharini, C. (eds.) 2020 21st International Arab Conference on Information Technology (ACIT), pp. 1–4 (2020)
15. Punagin, A., Punagin, S.: Analysis of lane detection techniques on structured roads using openCV. Int. J. Res. Appl. Sci. Eng. Technol. **8**, 2994–3003 (2020)
16. Teo, T.Y., Sutopo, R., Lim, J.M.-Y., Wong, K.: Innovative lane detection method to increase the accuracy of lane departure warning system. Multimed. Tools Appl. **80**, 2063–2080 (2021)
17. Haris, M., Hou, J., Wang, X.: Lane line detection and departure estimation in a complex environment by using an asymmetric kernel convolution algorithm. Vis. Comput. **39**(2), 519–538 (2023)
18. Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., Pazhayampallil, J., Andriluka, M., Cheng-Yue, R., Mujica, F., Coates, A.: An empirical evaluation of deep learning on highway driving. Comput. Sci. (2015). https://doi.org/10.48550/arXiv.1504.01716 https://doi.org/10.48550/arXiv.1504.01716
19. Lee, S., Kim, J., Yoon, J.S., Shin, S., Bailo, O., Kim, N., Lee, T.-H., Hong, H.S., Han, S.-H., Kweon, I.S.: VPGNet: vanishing point guided network for lane and road marking detection and recognition. In: IEEE International Conference on Computer Vision (ICCV), pp. 1947–1955 (2017). https://doi.org/10.1109/ICCV.2017.215

20. Pan, X., Shi, J., Luo, P., Wang, X., Tang, X.: Spatial as deep: Spatial CNN for traffic scene understanding. arXiv e-prints. (2018)
21. Hou, Y., Ma, Z., Liu, C., Loy, C.C.: Learning lightweight lane detection CNNs by self attention distillation. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1013–1021 (2019). https://doi.org/10.1109/ICCV.2019.00110
22. Hou, Y., Ma, Z., Liu, C., Hui, T.-W., Loy, C.C.: Inter-region affinity distillation for road marking segmentation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 12483–12492 (2020). https://doi.org/10.1109/CVPR42600.2020.01250
23. Tabelini, L., Berriel, R., Paixao, T.M., Badue, C., De Souza, A.F., Oliveira-Santos, T.: PolyLaneNet: lane estimation via deep polynomial regression. In: 2020 25th International Conference on Pattern Recognition (ICPR), 6150–6156 (2021)
24. Qin, Z., Wang, H., Li, X.: Ultra fast structure-aware deep lane detection. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16, pp. 276–291 (2020)
25. Hedi, A., Lončarić, S.: A system for vehicle surround view. IFAC Proc. Vol. **45**(22), 120–125 (2012)
26. Natroshvili, K., Scholl, K.-U.: Automatic extrinsic calibration methods for surround view systems. In: IEEE Intelligent Vehicles Symposium (IV), pp. 82–88 (2017)
27. Ueshiba, T., Tomita, F.: Calibration of multi-camera systems using planar patterns. Sensors **8**, 4 (2002)
28. Zhao, K., Iurgel, U., Meuter, M., Pauli, J.: An automatic online camera calibration system for vehicular applications. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 1490–1492 (2014)
29. Lourakis, M.I.: Sparse non-linear least squares optimization for geometric vision. In: Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5–11, 2010, Proceedings, Part II 11, pp. 43–56 (2010)
30. Moré, J.J.: The Levenberg–Marquardt algorithm: implementation and theory. In: Numerical Analysis: Proceedings of the Biennial Conference Held at Dundee, June 28–July 1, 1977, pp. 105–116 (2006)
31. Dubská, M., Herout, A.: Real projective plane mapping for detection of orthogonal vanishing points. In: BMVC (2013)

Leidong Qin received the BS degree in the Internet of Things engineering from Beijing Jiaotong University, Beijing, China, where he is currently pursuing the MS degree with the Institute of Information Science. His current resea rch interests include image processing, camera calibration, and computer vision.

Chunyu Lin received the PhD degree from Beijing Jiaotong University (BJTU), Beijing, China, in 2011. From 2009 to 2010, he was a Visiting Researcher at the ICT Group, Delft University of Technology, The Netherlands. From 2011 to 2012, he was a Postdoctoral Researcher with the Multimedia Laboratory, Ghent University, Belgium. He is currently a Professor at BJTU. His research interests include image/video compression and robust transmission, 3D video coding, virtual reality video processing, and ADAS.

Shujuan Huang received the BS degree in the Internet of Things engineering from Beijing Jiaotong University, Beijing, China, where he is currently pursuing the MS degree with the Institute of Information Science. His current resea rch interests include image processing, depth completion, and computer vision.

Shangrong Yang received the BS degree from Beijing Jiaotong University (BJTU), Beijing, China, in 2018, where he is currently pursuing the PhD degree in signal and information processing with the Institute of Information Science. His current research interests include transformer, unsupervised learning, and reinforcement learning.

Yao Zhao received the BS degree from the Radio Engineering Dep artment, Fuzhou University, Fuz hou, China, in 1989, the ME degr ee from the Radio Engineering Department, Southeast University, Nanjing, China, in 1992, and the PhD degree from the Institute of Information Science, Beijing Jiaotong University (BJTU), Beijing, China, in 1996. He became an Associate Professor at BJTU in 1998 and a Professor in 2001. From 2001 to 2002, he was a Senior Research Fellow with the Information and Communication Theory Group, Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands. He is currently the Director of the Institute of Information Science, BJTU. His current research interests include image/video coding, digital watermarking and forensics, and video analysis and understanding. He is a fellow of IET. He was named a Distinguished Young Scholar by the National Science Foundation of China in 2010. He was also elected as a Chang Jiang Scholar of Ministry of Education of China in 2013.