Lane Detection Based on Inverse Perspective Transformation and Kalman Filter

Yingping Huang, Yangwei Li*, Xing Hu, Wenyan Ci
School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai, China
[e-mail: 2531222095@qq.com]
*Corresponding author: Yangwei Li

Received April 13, 2017; revised July 10, 2017; revised August 10, 2017; accepted September 21, 2017; published February 28, 2018

Abstract

This paper proposes a novel algorithm for lane detection based on inverse perspective transformation and Kalman filter. A simple inverse perspective transformation method is presented to remove perspective effects and generate a top-view image. This method does not need to obtain the internal and external parameters of the camera. The Gaussian kernel function is used to convolute the image to highlight the lane lines, and then an iterative threshold method is used to segment the image. A searching method is applied in the top-view image obtained from the inverse perspective transformation to determine the lane points and their positions. Combining with feature voting mechanism, the detected lane points are fitted as a straight line. Kalman filter is then applied to optimize and track the lane lines and improve the detection robustness. The experimental results show that the proposed method works well in various road conditions and meet the real-time requirements.

Keywords: Lane detection, inverse perspective transformation, feature extraction, Kalman filter, line fitting

http://doi.org/10.3837/tiis.2018.02.006 **ISSN : 1976-7277**

1. Introduction

Recent years, Advanced Driver Assistance System (ADAS) has attacted widespread attention of researchers. ADAS can effectively assist the driver and reduce the incidence of traffic accidents. Lane detection and tracking are a key technology for autonomous vehicle and ADAS applications such as Lane Departure Warning and Lane Change Assist. The task of lane extraction is to separate the traffic lanes from the background within an image.

Existing lane detection methods can be classified into two classes: feature-based approach and model-fitting approach. The feature-based approach extracts lanes by analyzing the low-level characteristics of the image. Several methods based on characteristics have been proposed. Yang et al. [1] proposed a lane detection method based on improved Hough transform. They applied a polar angle constraint to locate the position of the lane and then used dynamic Region of Interest to track the lane line. Chen et al. [2] presented a line scanning method based on imaging model to remove the interruption of non-lane road markings. The local maximum value of the edge contribution function is calculated after the edge point is extracted and then the lane line is fitted into a straight line. Yi et al. [3] extracts firstly the effective edge by using the image gradient, and then clusters and classifies the effective feature points into lane lines. Gao et al. [4] employed Hough transform to extract line features within an image and then used the direction and strength information of extracted line to determine the lane marking. In [5], the authors utilized spatiotemporal images to determine lane points, and fitted the detected lane points to a cubic curve. Lindner et al. [6] utilized multi-level features to determine geometric features as candidates that were further classified for the distinction between true and false lane marking points. In [7], authors employed a Kalman filter to track the lane lines detected by the Hough transform, and measurement updating of the Kalman filter was conducted by identifying the vanishing point, a left marking point and a right marking point. The feature-based approach can be influenced by factors such as lighting, lane breakage, tree shadows and road marking. Aly [8] presented a real-time method for detecting lane markers in urban scenarios. The method generated a top-view image of the road, and employed RANSAC line fitting method to give initial estimation for fitting Bezier Splines. Yoo et al. [9] proposed a lane detection method based on vanishing point estimation. Line segments that pass through the vanishing point were located. Candidate line segments for lanes were selected according to geometric constraints, and the host lanes were determined by using a score function. Mammeri et al. [10] used the Maximally Stable Extremal Region method with a three-stage refinement to determine lane line area. And then, the Progressive Probabilistic Hough Transform was employed to detect lane markings.

The mode-fitting approach employs a predefined curve model such as a line model or parabolic model to match the line features within an image, and therefore a lane detection issue becomes the process of calculating the parameters of the model. References [11, 12] presented a method utilizing generalized curve parameters model to fit lane lines. Adaptive random Hough transformation (ARHT) and the tabu search algorithm were used to calculate the parameters in the lane model. In [13], a B-snake lane model was used to

convert the problem of lane detection into the problem of finding the control points which determine the spline curve. As an improvement, Li et al. [14] presents a parallel-snake model combining with Kalman filter to achieve a more robust and accurate lane detection and tracking. Zhou et al. [15] proposed a deformable template model to match the lane boundaries, and used the Tabu search algorithm with the maximum posterior probability to estimate the parameters of the template. They have also employed a particle filter to recursively estimate the lane shape. Zhou et al. [16] used the main direction and the edge direction of lines to estimate parameters of the fitting model, and the Gabor filter was employed to further select the best lane marking parameters. In [17], a lane detection algorithm based on a hyperbola-pair lane boundary model was proposed. By fitting points on pair road boundaries into this model, the method was able to make full use of road boundaries with existence of partial occlusion. In [18], lane boundaries were initially detected using a linear–parabolic lane model, pairs of local maxima–minima of the gradient were used as cues to identify lane markings, and a Bayesian classifier based on mixtures of Gaussians was applied to classify the lane markings. In [19], lane lines were assumed to be cubic curves and fitted using the ransac algorithm. The model-based approach benefits for detection of a bending lane line, but requires complicated algorithm to determine the model parameters.

In [20], a machine learning based approach has been proposed for lane detection. Both structural and statistical features of the extracted bright shape are applied to the Neural Network for finding correct lane marks. In [21], convolutional neural networks demonstrated superior performance for image enhancement and lane detection.

This paper proposes a novel and simple line fitting method to extract lane lines using road feature points. The processing is conducted on the top-viewing image obtained from the inverse perspective transform, and Kalman filter is then employed to track the lane to ensure the detection robustness. The framework of the method is shown in **Fig. 1**.

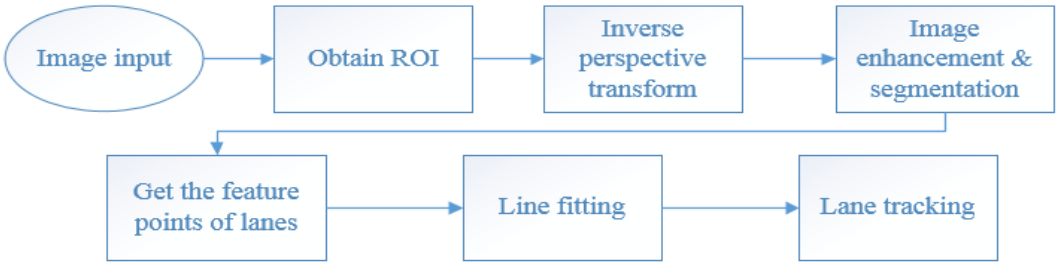


Fig. 1. Framework of the proposed method

2. The image pre-processing

2.1 Obtaining the region of interest (ROI)

The image obtained by the car camera contains a large non-road area, such as sky, trees on roadside, etc. Globally processing of the image will increase the computational complexity and reduce the real-time capability. Furthermore, the invalid area will interfere the lane information and affect the detection accuracy. Therefore, the valid region within the image should be selected to eliminate the invalid information. The area shown in the red box of **Fig. 2** is the region of interest (ROI) for lane detection, which reaches to the vanishing point and takes about half of the area in the bottom of the image depending on camera installation. In some cases that the front hood of the equipped car has been captured in the bottom of the image, the ROI must remove the bottom area containing the front hood.

Fig. 2. Region of interest

2.2 Inverse perspective transformation

Due to the imaging perspective effect, the traffic lanes in **Fig. 2** present as two non-parallel lines. The inverse perspective transformation is to eliminate the perspective effect and produce a top view image that is consistent with the actual situation. In the resulting top view image, the lane lines are vertical and parallel to each other, which facilitate the identification in subsequent algorithms.

Strict reverse perspective transformation requires the knowledge like the tilt angle and height of camera installation, the camera aperture size and other physical quantities. In this work, a simple trapezoidal transformation is used to produce the top view image. The ROI in **Fig. 2,** the rectangular region, should be converted into an inverted trapezoidal shape after the transformation. The inverse perspective transformation is simplified as the process that convert a rectangular into an inverted trapezoidal. Letting (u, v) denote the ROI

coordinate system, (x, y) denote the transformed top view coordinate system, the mapping relationship can be expressed as:

$$P = QM \tag{1}$$

where $P = [x', y', w']$ is homogeneous coordinates of image points in the ROI coordinate system, and $x = \frac{x'}{w'}$, $y = \frac{y'}{w'}$. $Q = [u, v, 1]$ is the homogeneous coordinates of image points in the transformed top view coordinate system. $M = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & 1 \end{bmatrix}$ is the transformation matrix from the rectangular to the inverted trapezoidal.

The matrix operation gives:

$$x = \frac{x'}{w'} = \frac{a \cdot u + b \cdot v + c}{g \cdot u + h \cdot v + 1}, \; y = \frac{d \cdot u + e \cdot v + f}{g \cdot u + h \cdot v + 1} \tag{2}$$

Equation (2) indicates that the transformation matrix M can be solved by locating four corresponding points between the ROI coordinate system and the top view coordinate system.

In practice, the points A, B, C, and D in the ROI image and the points A, E, F, D in the transformed image as shown in **Fig. 3** are selected as the corresponding points between the transformation to determine the matrix M. The top view image obtained from the transformation is shown in **Fig. 4**. It can be seen from the figure that the transformed top view image appears as an inverted trapezoid, and the lane lines present as parallel and vertical (or close to parallel and vertical).

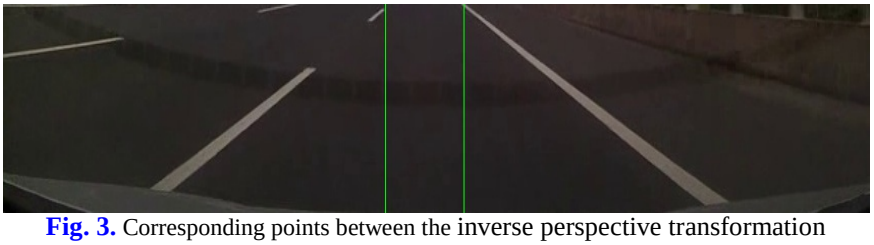


Fig. 3. Corresponding points between the inverse perspective transformation

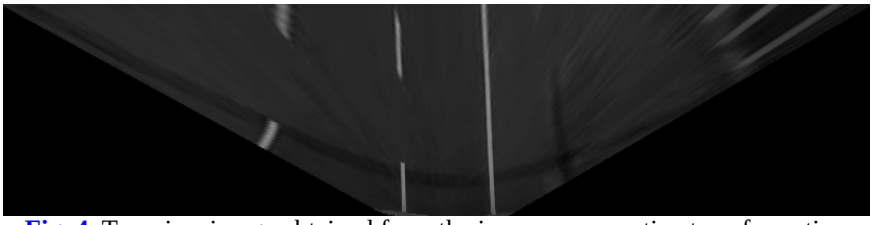


Fig. 4. Top view image obtained from the inverse perspective transformation

2.3 Lane enhancement

In order to highlight the lane lines and weaken the peripheral irrelevant area, a two dimensional Gaussian kernel function is used to convolute the image, the expression is as follows:

$$f_u(x) = \frac{1}{\sigma_x^2} \exp\left(-\frac{x^2}{2\sigma_x^2}\left(1 - \frac{x^2}{\sigma_x^2}\right)\right) \tag{3}$$

$$f_v(y) = \exp\left(-\frac{1}{2\sigma_y^2} y^2\right) \tag{4}$$

Equation (3) is the Gaussian kernel response function in the horizontal direction, and equation (4) is the Gaussian kernel response function in the vertical direction. The horizontal kernel is a second-derivative of Gaussian, whose σ_x is adjusted according to the width of the lanes (set to the equivalent of 8cm in the top-view image). The vertical kernel is a smoothing Gaussian, whose σ_y is adjusted according to the height of lane segment (set to the equivalent of 1 meter in the top-view image). Using this separable kernel allows for efficient implementation, and is much faster than using a non-separable kernel.

Fig. 5 shows the result of the Gaussian convolution. The left image is the 2D Gaussian kernel used for image convolution. It can be seen that the lane markers are enhanced, the surrounding area is weakened and the vertical response is stretched.

Fig. 5. Lane enhancement.

2.4 Image binarization

To distinguish lane lines from background, image is binarized according to the gray scale of the image pixels. The key to binarize an image is how to select the threshold. Since illumination varies under deferent road scenes, it is impossible to distinguish the lane lines from background using a fixed threshold. In this work, an iterative method is used to determine an optimal threshold to extract lane lines. The optimal threshold is capable of adapting to varied illumination.

The image can be divided into two regions of A and B according to the threshold $Th^k(k = 0,1,...n)$, and the average gray value of the two regions can be calculated as follows:

$$g_A = \frac{\sum_{g(i,j)\geq Th^k} g(i,j)}{\sum_{g(i,j)\geq Th^k} 1}, g_B = \frac{\sum_{g(i,j)<Th^k} g(i,j)}{\sum_{g(i,j)<Th^k} 1}, k = 0,1,\ldots,n \quad (5)$$

where $\sum_{g(i,j)\geq Th^k} 1$ denotes the number of pixels whose gray value is greater than or equal to the threshold Th^k, and $\sum_{g(i,j)<Th^k} 1$ denotes the number of pixels whose gray value is less than the threshold Th^k. $\sum_{g(i,j)\geq Th^k} g(i,j)$ is the sum of grey value of pixels whose gray value is greater than or equal to the threshold Th^k, and $\sum_{g(i,j)<Th^k} g(i,j)$ represents the sum of gray value of pixels whose gray value is less than the threshold Th^k.

Thus, a new threshold can be calculated as follows:

$$Th^{k+1} = \frac{g_A + g_B}{2}, k = 0,1,,\ldots,n \quad (6)$$

Repeat the above steps until $Th^k = Th^{k+1}$, and the resulting threshold is the optimal threshold. Initially, the threshold is set as:

$$Th^0 = \frac{g_{\max} + g_{\min}}{2} \quad (7)$$

where, $g_{\max}$ and $g_{\min}$ are the maximum and minimum gray value within the original image respectively. The binarized image is shown in **Fig. 6**.

Fig. 6. Image Binarization

3. The lane detection algorithm

3.1 Extraction of lane characteristics

In the top view image obtained from the reverse perspective transformation, the lane lines become vertical and parallel as their actual state. The following searching method is used to determine the location of the lane lines.

Step 1: In the binarized image, starting at the image center to search for the lane point to the left and the right. For each column, the points whose grey value is not zero are counted. As a result, the number of these points $f(x)$ is recorded as the function of the abscissa x.

Step 2: $f(x)$ will be an extreme value at a vertical line. However, due to a lane line having a certain width and not being perfectly vertical, a lane line may be identified as multiple straight lines. That is to say there are multiple extreme points within the lane's width range. A Gaussian filter is applied on $f(x)$ to merge the extreme points located closely.

Step 3: Some extreme points can't be merged by the Gaussian filter because they are far apart from each other. Setting a neighborhood α and merging these extreme points within α by the following processing. Assuming that two extreme points are found at x_1 and x_2, the combined abscissa of extreme points is calculated as follows:

$$x = \frac{x_1 f(x_1) + x_2 f(x_2)}{f(x_1) + f(x_2)} \tag{8}$$

The combined abscissa is calculated as a weighted average of the two extreme points with $f(x)$ as the weights. The position of the combined extremum is more biased towards the point whose $f(x)$ value is larger.

By this processing, each lane line will be determined as a single extremum. They are stored in an array H, and arranged in descending order. The extreme point response is plotted in **Fig. 7**.

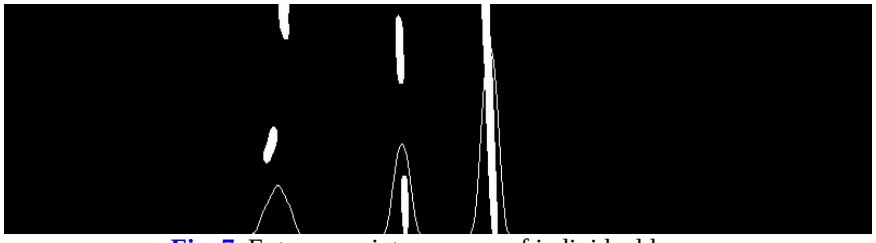


Fig. 7. Extreme point response of individual lanes

3.2 Line fitting

Each element in *H* denotes an extreme point corresponding to one lane line. Each lane line can be fitted according to its extreme point by the following steps:

Step 1: For each x in the array H, choose point $P_1(x_{b1},0)$ and $P_2(x_{b2},h)$ with $x-b<x_{b1}<x+b$ and $x-b<x_{b2}<x+b$, where b is the pixel number within the width of a lane line**,** and *h* is the height of the image. Connecting p_1 and p_2 generates a line.

Step 2: Step 1 will generate $(2b-1)^2$ straight lines in total for each *x* in the array H. Count the pixels with non-zero gray scale in a line as the polling score of the line.

Step 3: Take the line with the largest polling score as the lane line.

Step 4: Repeat the above 3 steps and obtain all lane lines within the image

This process will enable each element x in the array H to generate a straight line. In order to obtain the left and right lane lines of the current vehicle, it is assumed that the vehicle is traveling at the middle of the two lines, and the interference information in the middle of the lane are removed by the lane spacing constraint, such as the pedestrian line, as well as other straight lines outside the driveway.

4. The lane tracking algorithm

The damage of lane lines, road surface stains, shadows and the interference of other vehicles may lead to miss-detection in a frame or a few frames. In order to predict the lane position in the case where the lane is not recognized, the Kalman filter is used to track the lane. The lane tracking mechanism can greatly reduce the interference of various factors on the image and improve the robustness and speed of detection.

4.1 Lane tracking based on Kalman filter

Kalman filter is a high efficiency auto-regressive filter. It can optimize and predict the location of the target based on the measurements. The Kalman filter used a feedback mechanism to realize the filter estimation, mainly relying on a state equation and an observation equation. The two equations are as follows:

$$\begin{cases} X(k) = AX(k-1) + W(k) \\ Z(k) = HX(k) + V(k) \end{cases} \tag{9}$$

where, $X(k)$ is M-dimensional state vector, $Z(k)$ is N-dimensional state vector, A is state transition matrix of $M \times M$ dimension and H is observation matrix of $N \times M$ dimension. $W(k)$ is M-dimension process noise vector, $V(k)$ is N-dimension measurement noise vector, $W(k)$ as well as $V(k)$ can be modeled as Gaussian white noise and satisfy the following normal distribution:

$$W(k) \sim (0, Q), V(k) \sim (0, R) \tag{10}$$

where Q and R denote the process noise covariance matrix and measurement noise covariance matrix, respectively.

Based on the minimum mean square error theory, Kalman filter uses the optimized value $x(k-1)$ and the measured value $z(k)$ to solve the optimized value $x(k)$. This process is divided into two steps of prediction and correction.

First step: prediction (Time updating)

The estimated value $\hat{x}(k)$ is obtained from the optimized value $x(k-1)$ according to the state equation:

$$\hat{x}(k \mid k-1) = Ax(k-1) \tag{11}$$

The error covariance matrix can be presented as follows:

$$P(k \mid k-1) = AP(k-1)A^T + Q \tag{12}$$

The observed value at k th moment obtained from observation equation can be calculated as follows:

$$\hat{z}(k) = H\hat{x}(k \mid k-1) \tag{13}$$

Second step: Correction (Measurement updating)

The optimized value $x(k)$ can be calculated by correcting the estimated value $\hat{x}(k)$ with the observed value $z(k)$ and the gain matrix $G(k)$.

$$\begin{aligned} x(k) &= \hat{x}(k \mid k-1) + G(k)\left[z(k) - \hat{z}(k)\right] \\ &= \hat{x}(k \mid k-1) + G(k)\left[z(k) - H\hat{x}(k \mid k-1)\right] \end{aligned} \tag{14}$$

where, the gain matrix can be expressed as follows:

$$G(k) = P(k \mid k-1)H^T[HP(k \mid k-1)H^T + R]^{-1} \quad (15)$$

Updating the error covariance matrix as follows:

$$P(k) = [1 - G(k)H]P(k \mid k-1) \quad (16)$$

where, $x(k)$ and $P(k)$ are used for optimization of next frame, which is an iterative process for continuous prediction and updating.

In this work, the state vector $x(k)$ is defined as $[x_1, x_2, x_3, x_4]$, where x_1 and x_2 denote the upper and lower coordinates of the left lane, x_3 and x_4 denote the upper and lower coordinates of the right lane. In our method, the state transition matrix A and the observation matrix H are defined as $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, which means that the state vector $x(k)$ is regarded as unchanged between consecutive frames, and the observed values and estimated values are the same variables. The process noise covariance matrix Q is has little effect in the tracking process. In order to facilitate the calculation, we define $Q = \begin{bmatrix} 10^{-5} & 0 & 0 & 0 \\ 0 & 10^{-5} & 0 & 0 \\ 0 & 0 & 10^{-5} & 0 \\ 0 & 0 & 0 & 10^{-5} \end{bmatrix}$. The measurement noise has a greater effect than the process noise in the tracking process. The measurement noise covariance matrix is set as $R = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}$, which is an empiric value obtained from the experiments by recording the errors between real lane position and measured lane position. The iteration processing can reduce the effect of the measurement noise to give an optimized estimated value.

In order to obtain a fast convergence of the tracking, the autocorrelation value is set as 100, so the initial error covariance matrix is set as $P(0) = 100\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

In most cases, the equipped vehicle drives in the middle of the two lanes, and the curvature of the lanes is small, which means the lane in the near field can be approximated as a straight line. The lane's position under these situations is considered as an ideal state and used to initialize the Kalman filter. In the subsequent detection, if one or two lane lines are not detected successfully, the following process will be executed:

(1) For the case that the left lane is only detected, the right lane is set as the position of the left lane plus the width between two lanes. The positions of the two lanes are passed to the Kalman filter as the measured values.

(2) For the case that the right lane is only detected, the left lane is set as the position of the right lane plus the width between two lanes. The positions of the two lanes are passed to the Kalman filter as the measured values.

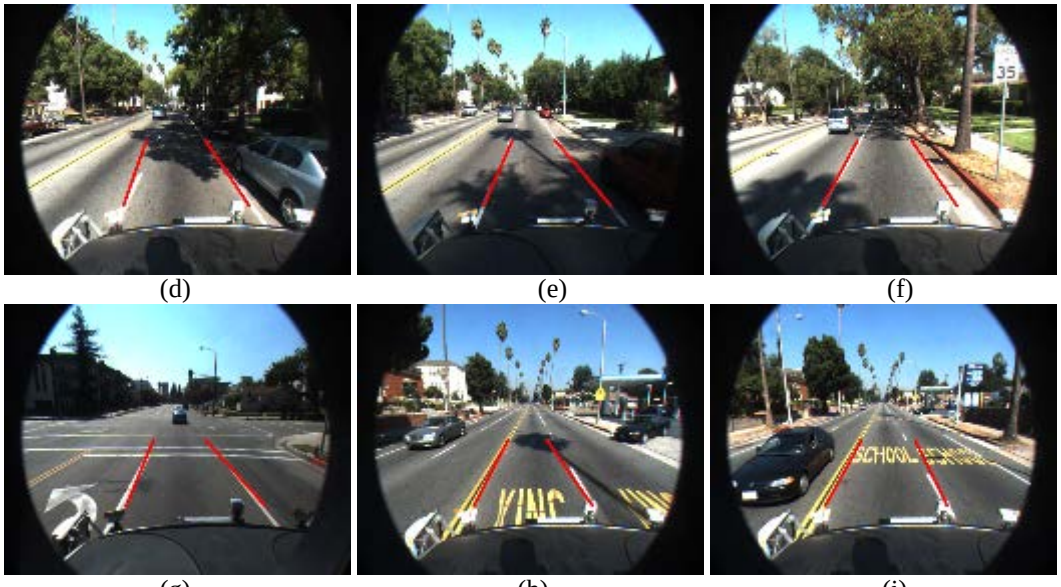
(3) For the case that both lanes are not detected, the locations of the ideal lanes are passed to the Kalman filter as the measured values.

5. Experiments and Results

5.1 Lane detection experimental results

Experiments have been conducted on Caltech image database [8], one of the popular benchmark platforms for lane detection in urban scenes, established by the California Institute of Technology. The results are shown in **Fig. 8**. It can be seen that the proposed method can tolerate the interferences like zebra marking (a, b), vehicles and shadows (c, d, e and f) and road signs (g, h and i), and shows robustness in these scenes.

(a) (b) (c)



(d) (e) (f)

(g) (h) (i)

Fig. 8. Lane detection results in urban scenes

Experiments have also been conducted on image sequences captured from highway and rural roads. The results are shown in **Fig. 9**. For highway scenarios, the system works well under poor lighting condition **Fig. 9** (a) (tunnel), yellow line interference **Fig.9** (b), and breakage of lane line **Fig.9** (c). Experiments also show that the system is capable of working in rural roads as shown in **Fig.9** (d), (e) and (f) interference. In summary, the proposed method is immune to various interferences and works well in various road conditions. The method has no problems like wide lane lines being identified as plurality of straight lines or dashed lanes being identified as discontinuous lines.

(a) (b) (c)

(d) (e) (f)

Fig. 9. Lane detection results in highway and rural road scenes

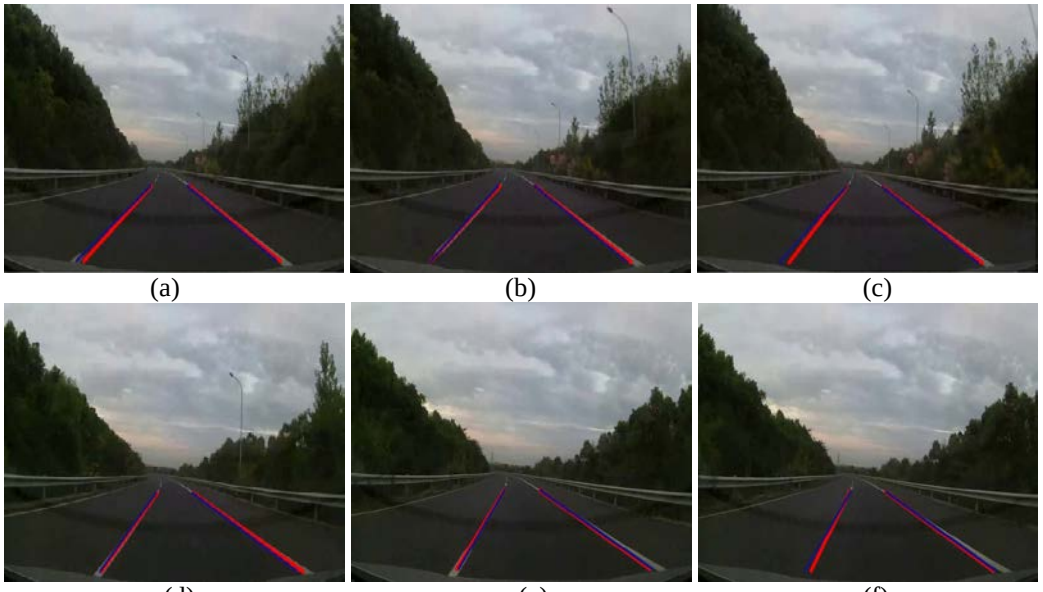
5.2 Lane tracking experimental results

In the cases that the lanes are not or wrongly detected, the Kalman tracking algorithm can take effects to compensate and correct the detection results. In the following images, the red lines are the results obtained from the detection algorithm while the blue lines represent the lanes obtained from the tracking algorithm. **Fig. 10** shows three special cases where the left lane line is broken (a), the shadow interference causes a detection failure of the right lane (b), and the road signs cause a deviation of the lane detection (c). In these cases, if we only rely on the lane detection algorithm, we will lose some lane lines (a) and (b) or get inaccurate lane position (c). But, with the effect of Kalman filter tracking, the lost lane lines can be retrieved and the deviated detection can be corrected according to the temporal consistency.

(a) (b) (c)

Fig. 10. Comparison of detection and tracking for three special cases

Fig.11 shows the lane detection and tracking results for a sequence of images in a highway scenario. The right lane is a long solid line, leading to a stable detection results. However, the left lane is a long broken line; thus, the detection results present a small range of fluctuations. With the effect of the Kalman filter, the left lane line becomes smooth and stable.

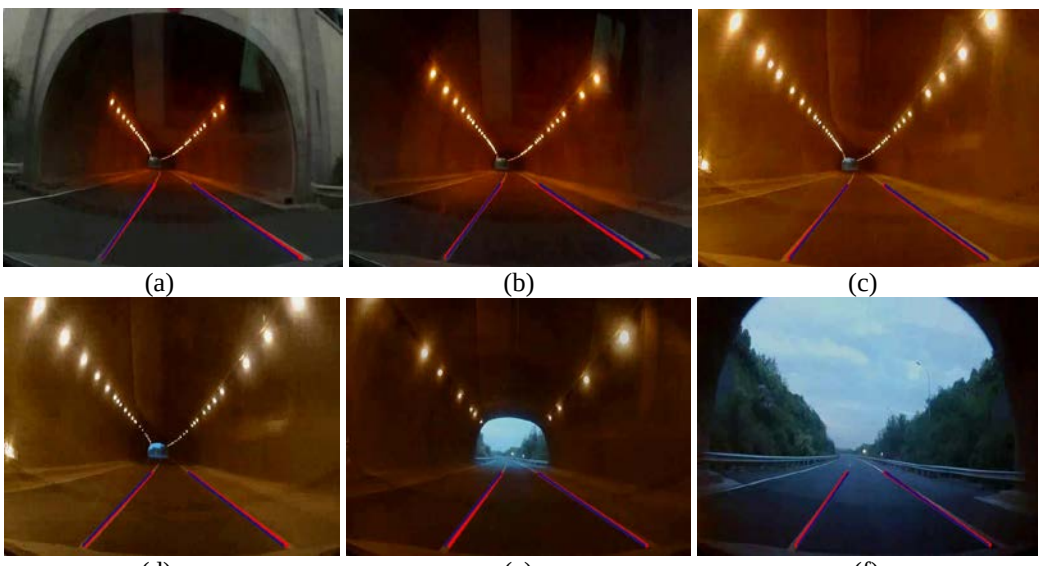


(a) (b) (c)

(d) (e) (f)

Fig. 11. Lane tracking results in highway scenario

Fig. 12 shows the lane tracking results when the vehicle passes through a tunnel. It can be seen that, thanks to the Kalman filter tracking algorithm, we can always get stable lanes even if the lighting condition is poor.



(a) (b) (c)

(d) (e) (f)

Fig. 12. Lane tracking results in poor lighting conditions

5.3 Evaluation and Comparison with other works

The proposed method has been compared to the approaches presented in [8, 9] by using the standard Caltech database. The database contains four clips: Cordova1 with 250 frames, Cordova2 with 406 frames, Washington1 with 336 frames, and Washington2 with 232 frames. Cordova1 has road signs on the street; Cordova2 contains various pavement markings with some of the frames lack of lane markings; Washington1 has shadows and passing cars with some double yellow lines; and Washington2 contains road signs, double yellow lines and vehicles as well. Overall, the database includes scenes with various interferences like pavement markings, broken lane markings, shadows, and so on. For every frame, each detected lane is compared to manually drawn ground truth. The criteria to determine if it is a correct or false lane detection is the same as the one used in [9].

Table 1 presents a comparison of lane detection rate on Caltech database. Aly's method detected lanes independently in each frame without tracking. As improvement, Yoo's method considers interframe similarity for location of the detected lanes and the estimated vanishing point in consecutive frames, so Yoo's method outperformed Aly's method. Our method combines the tracking mechanism with lane detection and presents a similar performance with Yoo's method. In most cases, the detection rate exceed 90%, except cordova2, because there are a large number of missing road markings in cordova2. There exist a lot of shadow interference in washington1, so the detection rate is relatively low. On average, lane detection rates of Aly's method is 87.96%, Yoo's method is 92.43%, and our method is 93.05%.

Table 1. Comparison of Lane detection rate (%) on Caltech database

	Aly [8]	Yoo [9]	The Proposed method
Cordova1	91.60	94.40	94.20
Cordova2	75.37	87.44	88.32
Washington1	92.43	89.61	91.45
Washington2	92.46	98.28	98.21
Mean	87.96	92.43	93.05

The system was implemented in Visual Studio 2013 with Opencv3.0 in a PC quipped with a 2.40-GHz Intel Dual Core i5 processor and 4GB of RAM. The image resolution is 640 × 480 pixels. In general, we can achieve a processing rate of 18-23 FPS depending on complexity of the images.

6. Conclusion

This paper proposes an efficient, robust and real-time algorithm for lane detection by using line fitting on the top-view image generated from the inverse perspective transformation. Kalman filter is adopted to optimize and track the lane lines and improve the detection accuracy and robustness. The experiment results show that the proposed method is robust against broken/missing lines, shadows/road sign interference and poor lighting conditions, and can effectively detect lanes under various road conditions. The implemented system

also meets the real-time requirement.

References

[1] X. N. Yang, J. M. Duan, D. Z. Gao and B. G. Zheng, "Research on Lane Detection Based on Improved Hough Transform," *Computer Measurement & Control*, vol. 18, No. 2, pp. 292-294, 2010. Article (CrossRef Link).

[2] L. Chen, Q. Q. Li and Q. Z. Mao, "Lane Detection and Following Algorithm Based on Imaging Model," *China Journal of Highway and Transport*, vol. 24, No. 6, pp. 96-102, November, 2011. Article (CrossRef Link).

[3] S. C. Yi, K. Q. Li, J. B. Guo and X. L. Gao, "Lane Marking Detection based on Edge Distribution and Feature Clustering," *Automotive Engineering*, vol. 36, No. 10, pp. 1210-1215, 2014. Article (CrossRef Link).

[4] Z. F. Gao, B. Wang, Z. Q. Zhou and F. F. Xu, "A Robust Algorithm for Lane Detection on Unplannar Road," *Transactions of Beijing Institute of Technology*, vol. 33, No. 1, pp. 73-78, January, 2013. Article (CrossRef Link).

[5] S. Jung, J. Youn and S. Sull, "Efficient Lane Detection Based on Spatiotemporal Images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, No. 1, pp. 289-295, January, 2016. Article (CrossRef Link).

[6] P. Lindner, S. Blokzyl, G. Wanielik, U. Scheunert, "Applying multi-level processing for robust geometric lane feature extraction," in *Proc. of IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)* , pp. 248-254, September 5-7, 2010. Article (CrossRef Link).

[7] A. Petrovai, R. Danescu and S. Nedevshi, "A stereovision based approach for detecting and tracking lane and forward obstacles on mobile devices", in *Proc. of IEEE Conference on Intelligent Vehicles Symposium (IV)*, pp. 634-641, June 28-July 1, 2015. Article (CrossRef Link).

[8] M. Aly, "Real time Detection of Lane Markers in Urban Streets," in *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 7-12, June 4-6, 2008. Article (CrossRef Link).

[9] J. H. Yoo, S. W. Lee, S. K. Park and D. H. Kim. "A Robust Lane Detection Method Based on Vanishing Point Estimation Using the Relevance of Line Segments," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1-13, March 2017. Article (CrossRef Link).

[10] A. Mammeri, A. Boukerche, Z. Tang, "A real-time lane marking localization, tracking and communication system," *Computer Communications*, vol. 73, pp. 132-143, January 2016. Article (CrossRef Link).

[11] W. H. Zhu, F. Q. Liu, Z. P. Li , X. H. Wang and S. S. Zhang, "A Vision Based Lane Detection and Tracking Algorithm in Automatic Drive," in *Proc. of IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, pp. 799-803, December 19-20 , 2008. Article (CrossRef Link).

[12] L. Qing, N. N. Zheng and H. Cheng, "Spingrobot: A Prototype Autonomous Vehicle and Its Algorithms for Lane Detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, No. 4, pp. 300-308, December 2004. Article (CrossRef Link).

[13] Y. Wang, E. K. Teoh and D. Shen, "Lane detection and tracking using B-Snake," *Image & Vision Computing*, vol. 22, No. 4, pp. 269-280, April 2004. Article (CrossRef Link).

[14] X. Y. Li, X. Z. Fang, Ci. Wang and W. Zhang, "Lane Detection and Tracking Using a Parallel-sanke Approach," *Journal of Intelligent & Robotic Systems*, vol. 77, No. 3, pp. 597-609, March 2015. Article (CrossRef Link).

[15] Y. Zhou, R. Xu, X. Hu and Q. Ye, "A robust lane detection and tracking method based on computer vision," *Measurement Science & Technology*, vol. 17, No. 4, pp. 736-745, February 2006. Article (CrossRef Link).

[16] S. Zhou, Y. Jiang, J. Xi and J. Gong, "A novel lane detection based on geometrical model and Gabor filter," in *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 59-64, June 21-24, 2010. Article (CrossRef Link).

[17] Q. Chen and H. Wang, "A Real-time Lane Detection Algorithm Based on a Hyperbola-Pair Model," in *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 510-515, June 13-15 , 2006. Article (CrossRef Link).

[18] M. B. D. Paula and C. R. Jung, "Automatic Detection and Classification of Road Lane Markings Using Onboard Vehicular Cameras," *IEEE Transactions on Intelligent Transportations System*, vol. 16, No. 6, pp. 3160-3169, December 2015. Article (CrossRef Link).

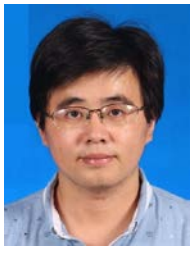
[19] M. Haloi and D. B. Jayagopi, "A robust lane detection and departure warning system," in *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 126-131, June 28-July 1, 2015. Article (CrossRef Link).

[20] S. Lim, D. Lee and Y. Park, "Lane Detection and Tracking Using Classification in Image Sequences," *Ksii Transactions on Internet & Information Systems*, vol. 8, No. 12, pp. 4489-4501, December, 2014. Article (CrossRef Link).

[21] J. Kim, J. Kim, J. G. Jang and M. Lee, "Fast learning method for convolutional neural networks using extreme learning machine and its application to lane detection," *Neural Networks*, vol. 87, pp. 109-121, March, 2017. Article (CrossRef Link).

Yingping Huang is a professor at University of Shanghai for Science and Technology, Shanghai, China. His research interests involve Intelligent Vehicles, Image Processing, Computer Vision and Vehicle Electronics.

Yangwei Li is a master student at University of Shanghai for Science and Technology, Shanghai, China. His research interests involve Image Processing and Computer Vision.



Xing Hu is a lecturer at University of Shanghai for Science and Technology, Shanghai, China. He received his PhD degree from Shanghai Jiaotong University in 2017. His research interests involve Machine Learning and Computer Vision.

Wenyan Ci is a PhD student at University of Shanghai for Science and Technology, Shanghai, China. His research interests involve Pattern Recognition and Vehicle Electronics.