ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

Improvement of Direction Decision Algorithm based on Bird's Eye View Transform for Self-Driving Cars

Hyeon-Seok Jeong[1] Seok-Ho Im[1]Hyeon-Ju Yoon[2]

U.G. Students, Department of Computer Engineering, Kumoh National Institute of Technology, Gumi-City, Republic of Korea[1]

Associate Professor, Department of Computer Engineering, Kumoh National Institute of Technology, Gumi-City, Republic of Korea[2]

ABSTRACT: Lane detection is one of the most fundamental functions for self-driving cars to decide the steering direction. There are several approaches to detect lane marks by processingthe images from front or side video cameras attached to the car. In this paper, we adopt the bird's eye view transform and improved the algorithm with EPM(Expected Perspective Mapping) and modified sliding window algorithm to accommodate the limited hardware facility of embedded system. Proposed algorithm is able to get the same lane mark extraction result as existing method with about 30% less computation, so it is well suited to the real-time environment of self-driving car.

KEYWORDS: Bird's eye view transform,Expected Perspective Mapping,Lane DetectionSelf-Driving Car, Sliding Window

I. INTRODUCTION

Lane marks on roads are among the most important and fundamental feature in the process of autonomous driving. On-board cameras mounted on the front of autonomous vehicles capture road scene images, then the images are processed with several image processing and computer vision techniques to be able to extract the lane information. Simplified binary images with features such as line segments are used to decide the steering direction and angle.

Vision-based lane detection systems usually consist of three main procedures, which are image pre-processing, lane detection and lane tracking[1]. In the pre-processing step, the raw image is transformed and prepared to make next procedures easy, which includes color space conversion, vanishing point detection, ROI(region of interest) selection, noise removal, perspective view transform, segmentation, etc. Once the input images have been pre-processed, lane features such as the colors and edge features can be extracted. The Hough Transform algorithm is one of the most widely used algorithms for lane detection, but this is designed to detect straight lines and does not show good performance for curve lanes. Curve lanes can often be detected based on model fitting techniques such as RANSAC(RANdom SAmple Consensus) algorithm, which fits lane models by recursively evaluating to find the optimal model parameters. If the more elaborate lane tracking ability is needed, tracking algorithms such as Kalman filter or particle filters to refine the detection results and predict lane positions.

In this paper, we developed a lane detection algorithm which would be used in a tiny model car with very limited computing power. We adopt and combined typical and representative techniques such as color space conversion, Canny edge detection with Sobel filter, bird's eye view transform, and edge-linking algorithm to extract the lane direction and angle very fast.

According to the classification of Xing et al.[1], our algorithm can be categorized in feature-based method and we referred excellent work of previous researchers, applied and combined their method into ours. In [2], noisy lane edge feature were detected using the Sobel operator and the road images were divided into multiple subregions along the vertical direction. Collado et al.[3] created a bird-view of the road image and proposed an adaptive lane detection and classification method based on spatial lane features and the Hough transform algorithm. Lin et al.[4] uses an extended edge linking algorithm in the lane speculation stage to detect lane edges and decide the lane direction and curvature.In many works, The YUV video images were converted to HSV format to increase the contrast, then the binary feature image were processed using the threshold method[5].We modified bird's eye view transform and edge-linking algorithm to reduce the computationamount.

II. Related Work

We develop a lane detection algorithm to use for the small model car with low-ability embedded system, and the test course uses only yellow solid line as lane marks. Therefore, the focus of our work is rather rough and speedy than robust or accurate lane detection or tracking. Overall stages are organized with simple and universal well-known approaches, and we propose the computationally improved method for bird's eye view transform and lane direction decision by feature extraction using sliding window algorithm. Figure 1 is the flowchart of the lane detection process.

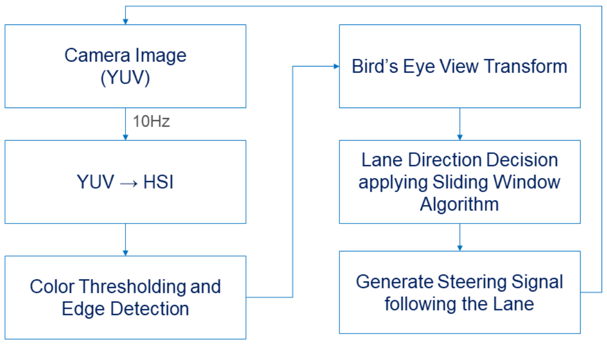
Camera Image (YUV)
10Hz
YUV → HSI
Color Thresholding and Edge Detection
Bird's Eye View Transform
Lane Direction Decision applying Sliding Window Algorithm
Generate Steering Signal following the Lane

Figure 1. Lane Detection Process

The real-time video image is provided at regular speed (e.g. 10Hz) from the on-board camera. First, YUV format of the image is converted into HSI format to focus on the hue and saturation of each pixel while decreasing the influence of intensity. Next, the line segments are extracted from the converted image using Canny edge detection algorithm[6], and the result is binary image.

Then,perspective mapping is performed to transform the perspective view into bird's eye view. Left image of Figure 2 is transformed into right image as we look down from the sky. Transformed image is more convenient to extract features than the original perspective view image. The transformation uses four-points correspondence between two images.

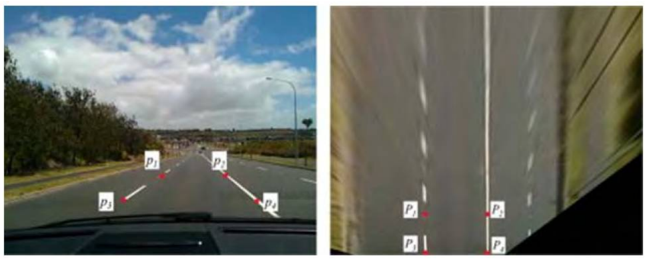

Figure 2. Bird's Eye View Transform

WPM(Warp Perspective Mapping)[7] assumes the road is flat and plain and selects 4 anchor points P_W at the camera calibration stage, which is viewed as P_I in the input image. The mapping from P_I to P_W can be achieved by an affine matrix A as follows:

$$P_W = A \cdot P_I \quad \text{eq. (1)}$$

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

Once the matrix is calculated from the anchor points, then the whole input image is mapped pixel-by-pixel using the matrix.

IPM(Inverse Perspective Mapping)[8, 9] is more widely used in real world lane detection. It uses the camera parameters such as camera's position, viewing direction, aperture, and resolution. IPM is modelled as a projection from a 3D Euclidean space onto a planar 2D subspace, and the mapping is presented as a 3×3 homography matrix. Once all camera parameters are acquired and the matrix is calculated, the mapping is calculated pixel-by-pixel as WPM.

After candidate line segments are extracted through the image transformation and edge detection process, feature extraction should be performed to decide the direction of lane and control the wheel. We use an edge-linking[4, 10] algorithm to link the feature points into line-segments so that noise points can be removed by checking the directions and length. Compared to the other methods such as Hough transform or RANSAC, this approach is not so accurate but sufficient to simple lane of test course and it takes much less time so suitable for our system.

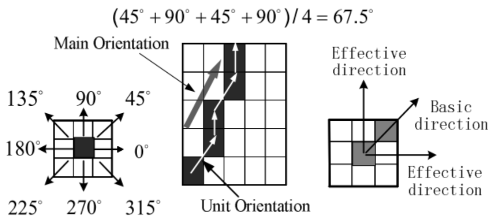

Figure 3. Finding Link Direction[Lin]

A binary image is divided into several small unit, we call it a "window". The windows slide from bottom to top finding the black pixels. In each unit, the unit direction is decided according to the orientation of one pixel's 8 neighborhoods as shown in Figure 3. The main direction of edge-link is defined as the mean value of all unit directions between every two neighboring units. The algorithm start from the bottom units finding the starting point of a link, and traces pixels associated with the starting point. While tracing, basic and effective direction in a unit is a guide for next pixel search. The basic direction is the unit direction between the first two pixels in a link, and effective unit-direction is within ±45° to the basic direction. The sliding window stops if no more connected pixels can be found or the unit direction of the next neighboring pixel falls out of effective direction range. If there are two or more starting points, the same process is repeated for every starting point.

III. Computation Improvements

A. *Bird's Eye View Transform by Expected Perspective Mapping*

For bird's eye view transform, we propose the EPM(Expected Perspective Mapping) which can utilize the simple road environment and run with the limited computing power. After the image passes through the color space conversion and edge detection, we can get a black-and-white image with one or two straight lines. Instead of product operation every pixel with the affine matrix, we get the x-coordinatevalue of each pixel from the line equation and interpolation because the y-coordinate is the same on both images. From the 4 anchor points of perspective image, we calculate two line equations(Figure 4). We call the line equation as "Expected Function" while *a* and *a'* is slope of the line, *b* and *b'* is the y-intercept value.

On both images the y-coordinate value is identical and the x-coordinate of start and end point of each by eq. (2). Then other points are calculated with bilinear interpolation

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

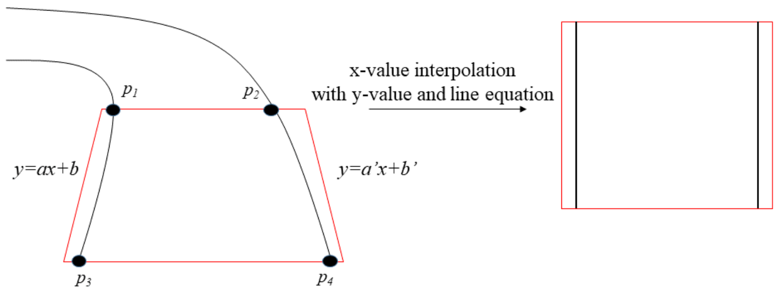

Figure 4. Expected Perspective Mapping

$$x = \frac{y-b}{a}, \quad x' = \frac{y'-b'}{a'} \qquad \text{eq. (2)}$$

Our algorithm can acquire the same transformed image as the WPM method with much less computation.Table 1 shows the time complexity of IPM, WPM and our EPM method, while n is the number of pixels in a trapezoid. IPM and WPM should access all pixels with 3×3 homography matrix, so basic time complexity is $9\times O(n)$. In addition to that, IPM takes more time because it considers other factors including camera position, angle and the real distance from camera center direction. EPM applies the homography matrix just once with the both end points of line, and other pixels are computed with interpolation by the line equation. Therefore the complexity is $O(n)$.

Table 1. Comparison of Transform Algorithms

Algorithm	Time Complexity
IPM (Inverse Perspective Mapping)	$9\times O(n) + \alpha$
WPM (Warp Perspective Mapping)	$9\times O(n)$
EPM (Expected Perspective Mapping)	$O(n)$

B. *Modified Sliding Window Algorithm for Direction Decision*

With the result of bird's eye view transform, we have to decide the direction of lane for steering control. Our algorithm is based on the sliding window algorithm introduced in the previous section, while applying the window adaptively instead of searching every unit on the image.

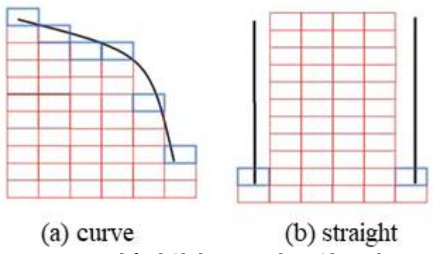

Figure 5. Modified Sliding Window Algorithm

Assume that the window size is 40×20 on the 320×240 image. Starting with the 8 windows from the bottom, it extracts the line segment in a unit window. If black pixels are detected, the window stops there not growing up. Then we can know the direction and the angle of the lane from the number of uppermost windows having the line segments and y-axis value of each window. Figure 5 shows the result of left curve lane and straight lane.

Worst case is that there is no line segments in the image, then every pixel should be tested as the original sliding window algorithm, but it doesn't happen because we are working with model car and test driving course. On average, we could reduce about 30% of computation with modified algorithm. On the other hand, our algorithm is not robust for sharp curve and camera noise. We tried to solve the problem by decreasing the car speed with significant amount, but other exception handling or noise reduction in pre-processing would be better solution.

IV. EXPERIMENT RESULTS

We developed this algorithm for self-driving model car competition which is a part of the Embedded Software Contest 2017[11]. Hyundai Autron[12] provides a small model car and basic system software including OS, SDKand device drivers.The contest participants developan embedded software program including image processing and recognition and hardware control algorithms to drive fast and safely on the given test course(Figure 8). The specification of provided hardware is shown in Table 2.

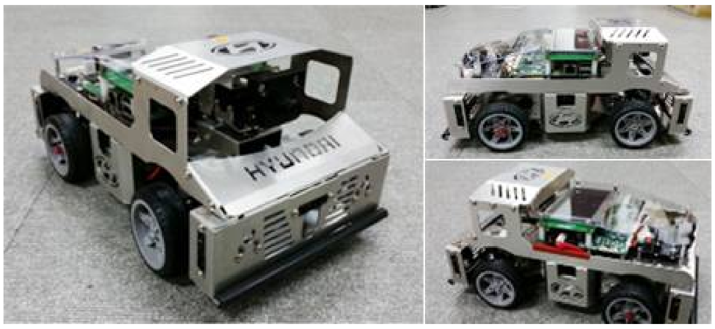

Figure 6. Model car for self-driving test

Table 2. Model Car Specification

Component	Specification
Size	330 × 190 × 160mm
Weight	2.3 kg
Drive motor	DC Servo motor Encoder (12V-12W)
Steering and Camera tilt motor	DC Servo motor-3EA (6V)
Camera	CMOS VGA(640×480), CVBS output
Computation Processor	NVIDIA Tegra3(ARM Cortex-A9 quadcore 900MHz)
Control board	Arduino
Debugging tool	Ethernet(10/100) and mini-USB
Distance sensor	Infrared sensor, 6 sets, 4cm ~ 30cm
Battery	NIMH 12V-3000mA
Software	Linux OS, Custom SDK, Device drivers

Figure 7 shows the lane mark extraction result processed on the model car. Figure 7(a) presents a straight lane but inclined to the left side, so the steering control signal is issued to turn right slightly then we can get the straight lane image again. Figure 7(b) shows the curved lane. With this image, the lane is detected in 5 or more windows at the upper right direction so we can determine the car is on the curved lane. The exact steering angle and motor speed for curved lane were acquired by empirical study because it depends on the hardware conditions such as battery level. Our algorithm reduces the amount of computation so it could run relatively fast, but it sometimes went off the lane at the steep curve due to the insufficient deceleration.

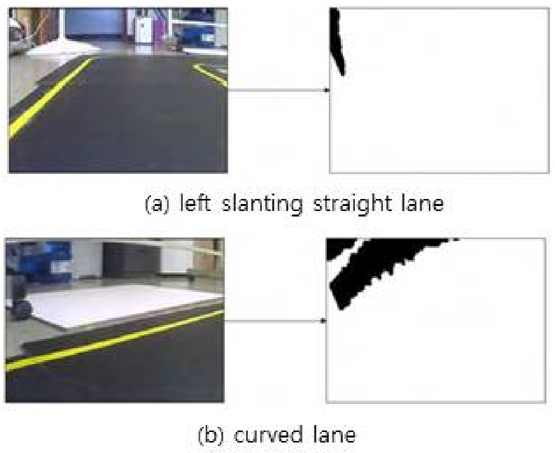

Figure 7. Lane mark extraction result

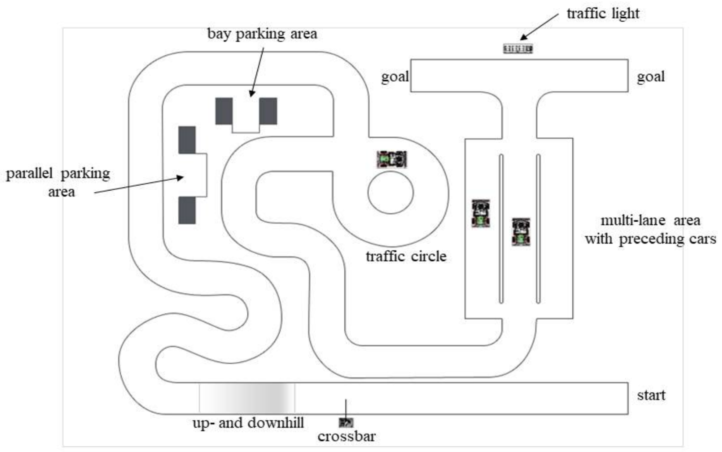

Figure 8. Driving Course

V. CONCLUSION

This paper proposed some improvement of lane mark detection and steering direction decision algorithm based on the bird's eye view transform to apply on the self-driving model car with very limited embedded computing facility. Our algorithm showed the same result compared to the previous methods while it can reduce the computation amount up to 30%. We participated in the self-driving model car section of 2017 Embedded Software Contest with this program, and run on the competition course with relatively high speed.

It may not so much appropriate for real world autonomous cars because real cars can have more powerful computing devices and the road situation is far more complex. But accumulated small enhancement would make the progress in the future.

REFERENCES

1. Xing, Y., Lv, C., Chen, L., Wang, H., Wang, H., Cao, D., Velenis E. and Wang, F.-Y., "Advances in Vision-Based Lane Detection: Algorithms, Integration, Assessment, and Perspectives on ACP-Based Parallel Vision", IEEE/CAA Journal of Automatica Sinica, Vol. 5, No.3, pp. 645-661, May 2018.
2. Kang, D. J. and Jung, M. H., "Road lane segmentation using dynamicprogramming for active safety vehicles," Pattern Recognition Letters, vol. 24,no. 16, pp. 3177−3185, Dec. 2003.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 5, May 2018

3. Collado, J. M., Hilario, C., de la Escalera, A. and Armingol, J. M., "Adaptative road lanes detection and classification," in Proceedings of 8th International Conferenceof Advanced Concepts for Intelligent Vision Systems, Springer,pp. 1151−1162, 2006.
4. Lin, Q., Robust Lane Detection Method using Bird's-eye View Transform, MS Thesis, Soongsil University, 2011.
5. Cela, A. F., Bergasa, L. M., Sanchez, F. L. and Herrera, M. A., "Lanesdetection based on unsupervised and adaptive classifier," in Proceedings of 5th International Conference of Computational Intelligence, Communication Systems andNetworks (CICSyN), pp. 228−233, 2013.
6. Canny J., "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.41, No.8, pp.2512-2524, August 2008.
7. Kim, Z. W., "Robust Lane Detection and Tracking in Challenging Scenarios", IEEE Transactions on Intelligent Transportation Systems, Vol.9, No.1, pp. 16-26, 2008.
8. Mallot, M. A., Hulthoff, H. H. B., Little, J. J., and Bohrer, S., "Inverse Perspective Mapping Simplifies Optical Flow Computation and Obstacle Detection", Biological Cybernetics, Vol. 64, pp. 177-185, 1991.
9. Muad, A.M., Hussain, A., Samad, S.A., Mustaffa, M.M., and Majlis, B.Y.,"Implementation of Inverse Perspective Mapping Algorithm for the Development of an Automatic Lane Tracking System", In Proceedings of IEEE Region 10 Conference TENCON, pp. 207-210, November 2004.
10. Ghita. O. and Whelan, P., "Computational approach foredge-linking".Journal of Electronic Imaging.Vol.11, No.4, pp.479-485, 2002.
11. The 15th Embedded Software Contest, http://swcontest2017.wayhome.kr/htm/collusion_04.php
12. http://hyundai-autron.com