International Conference on Control, Automation and Systems 2008
Oct. 14-17, 2008 in COEX, Seoul, Korea

New Lane Detection Algorithm for Autonomous Vehicles Using Computer Vision

Quoc-Bao Truong[1] and Byung-Ryong Lee[2]

[1] Department of Mechanical and Automotive Engineering, University of Ulsan, Ulsan, Korea
(Tel : +82-52-259-2147; E-mail: baotruong@mail.ulsan.ac.kr)
[2] School of Mechanical and Automotive Engineering, University of Ulsan, Ulsan, Korea
(Tel : +82-52-259-2861; E-mail: brlee@mail.ulsan.ac.kr)

Abstract: For navigation tasks it is necessary to determine position of the ego vehicle relative to the road. One of the principal approaches is to detect road boundaries and lanes using a vision system in the vehicle. This paper presents a simple and robust method designed to detect and estimate the curvature of road lane boundaries from images provided by a monocular camera. First, we use Vector-lane-concept and Non-uniform B-Spline (NUBS) interpolation method to construct the boundaries road lane. Based on lane detection result, we estimate the curvature of left and right lane boundaries for Autonomous Guided Vehicle systems application. Some experimental results based on real world road images are presented. These simulation results show the efficiency, feasibility and robustness of the algorithm.

Keywords: Lane detection, road lane curvature, Vector-lane-concept, Non-uniform B-Spline (NUBS) interpolation, Autonomous Guided vehicle.

1. INTRODUCTION

The potential applications of autonomous vehicles are widely varied and quite important such as in hazardous environment and intelligent highway system. Autonomous control of vehicles may help relieve traffic congestion by the alleviation of unnecessary stop-and-go behavior that occurs due to high vehicle density.

Lane detection is crucial to vision-based lateral control as well lane departure warning for autonomous driving. Vision based lane detection methods can be categorized in the following three classes region-based, feature-based and model-based methods.

In region-based methods the lane detection problem is considered as a classification problem. A classification problem consists of feature extraction, feature de-correlation and reduction, and clustering and segmentation [1].

Feature-based methods: The meaningful structures of the road (lanes or lane markings) are constructed based on some features which are extracted from edge image. One examples of this approach is presented in [2].

Most of vision-based methods for lane detection are model based. They use a geometric model to characterize the lane. These approaches aim to match the observed images with a hypothesized road model. This technique is based on the assumption that the shapes lane can be represented by straight lines, parabolic curves, or snack-like curves which can be found in [2, 3, 4, 5].

Our lane detection and estimation algorithm is described in Figure 1. First, the road image is enhanced using some conventional image processing techniques. In the enhanced image, possible lane marking pixels are detected using Canny operator. Next, we apply parallel thinning algorithm [6, 7] into edge map to obtain skeleton image. Detected lane marking pixels after taking skeleton process, then, are utilized to select control points for NUBS interpolation to construct left and right road lanes. In our case, the right and left lanes to be detected are well separated, which means that each lane can be considered as separated from the other (unlike most of previous works which have lane models of a uniform width). We use vector-lane-concept to select and correct these control points, overcome noise problem to get more robust result of road lane boundaries. Finally, we introduce a simple mathematical model to estimate left lane and right curvature for autonomous vehicle system.

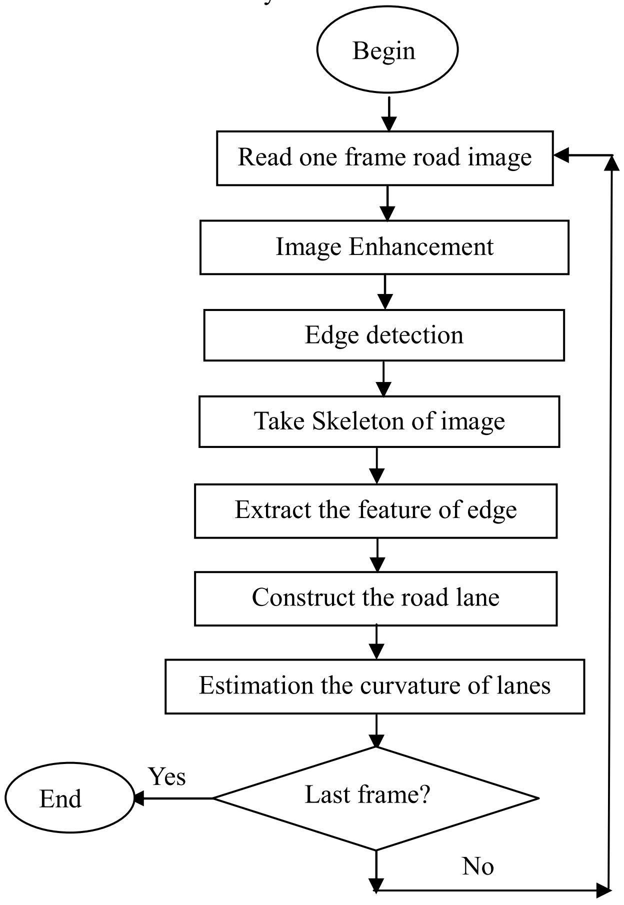


Fig. 1 The flow diagram of our method.

Section 2 summarizes these techniques that used in our work to obtain enhanced image, edge image and skeleton image by using parallel thinning algorithm. The road model uses NUBS interpolation approach is depicted in section 3. Then, the lane detection algorithm based on vector-lane-concept is proposed in section 4. Method to estimation lane-curvature and lane tracking is shown in section 5. Some simulation results are presented in section 6. Finally, the conclusion and future work are described in section 7.

2. IMAGE ENHANCEMENT, IMAGE EDGE DETECTION AND THINNING ALGORITHM

2.1 Image enhancement

Before taking edge detection, the capture image is enhanced using Gaussian and Median filter to reduce image noise. Next, histogram processing technique is applied to improve brightness and contrast of capture image as describe in equation 1:

$$O(x,y)=\frac{I(x,y)-\min[I(x,y)]}{\max[I(x,y)]-\min[I(x,y)]}*255 \qquad (1)$$

where $I(x,y)$ and $O(x,y)$ are input and output gray of pixel at (x,y) position, respectively.

2.2 Image edge detection

We apply Canny edge detection algorithm to obtain rough edge map from enhanced image. Subsequent, we use morphological operators for edge image output to remove potential holes between edge segments. Figure 2 describes one example of this task. From top to bottom and left to right are the original image, the image after apply reduce noise filter operations; edge image before and after applying morphological operators.

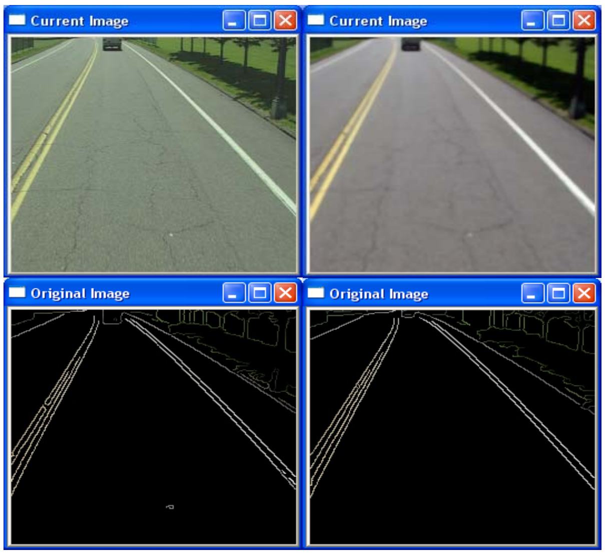


Fig. 2 Original image, images after apply enhancement, Canny edge detection and morphological operators

2.3 Thinning algorithm

Before taking edge detection, the capture image is enhanced using Gaussian and Median filter to reduce image noise. Next, histogram processing technique is applied to improve brightness and contrast of capture image as describe in equation 1:

In order to extract the most accuracy feature points of image, we use the parallel thinning algorithm is described in [8, 9] to get a skeleton of edge image.

Thinning is the process of reducing an object in a digital image to the minimum size necessary for machine recognition of that object. After thinning, analysis on the reduced size image can be performed and got the edge image with one pixel per edge lane road. Thinning is essentially a "pre-processing" step used in many image analysis techniques. Thinning is usually performed on monochrome, or two-color images. Figure 3 shows one example of a thinned handwritten "2".

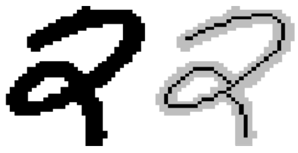


Fig. 3 Thinning a handwritten "2"

In parallel picture processing, the new value given to a point at the nth iteration depends on its own value as well as those of its eight neighbors at the (n - 1)th iteration, so that all picture points can be processed simultaneously. It is assume that a 3 x 3 window is used and that each element is connected to its eight neighboring elements as shown in table 1

Table 1 Designation of the Nine Pixels in 3 x 3 Windows

P_9 $(i-1,j-1)$	P_2 $(i-1,j)$	P_3 $(i-1,j+1)$
P_8 $(i,j-1)$	P_1 (i,j)	P_4 $(i,j+1)$
P_7 $(i+1,j-1)$	P_6 $(i+1,j)$	P_5 $(i+1,j+1)$

In the algorithm, each iteration is divided into two sub-iterations. In the first sub-iteration, the contour point P_1 is deleted from the digital pattern if it satisfies the following conditions:

$$\begin{array}{ll}(a).\ 2<B(P_1)<6 & (c).\ P_2*P_4*P_6=0 \\ (c).\ A(P_1)=1 & (d).\ P_4*P_6*P_8=0\end{array} \qquad (2)$$

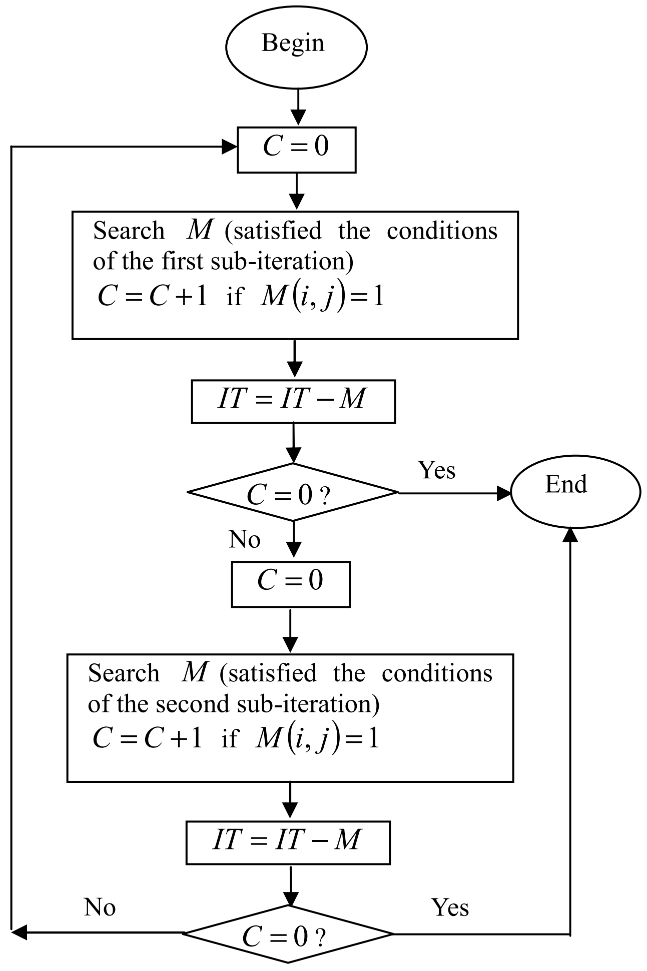
Where $A(P_1)$ is the number of 0-1 pattern in the ordered sets $P_2,P_3,\cdots P_8,P_9$ that are the eight neighbors of P_1 and $B(P_1)$ is the number of non-zero neighbors of P_1. That is:

$$B(P_1)=P_2+P_3+\cdots+P_8+P_9 \qquad (3)$$

In the second sub-iteration, only conditions (c) and (d) are changed as follows:

$$(c').\ P_2 * P_4 * P_8 = 0$$
$$(d').\ P_2 * P_6 * P_8 = 0 \quad (4)$$

and the rest remain is the same.



Begin
$C=0$
Search M (satisfied the conditions of the first sub-iteration)
$C=C+1$ if $M(i,j)=1$
$IT = IT - M$
$C=0$?
Yes
End
No
$C=0$
Search M (satisfied the conditions of the second sub-iteration)
$C=C+1$ if $M(i,j)=1$
$IT = IT - M$
No
Yes
$C=0$?

Fig. 4 The flow chart of parallel thinning algorithm

3. ROAD MODEL

As mentioned before, the model-based approach is the most appropriate for road detection and tracking application. In our case, we use non-uniform B-spline (NUBS) interpolation to construct left and right lanes of the road. NUBS interpolation is more complicated than uniform B-spline. However, we use NUBS interpolation is presented in [8, 9] for lane detection and tracking goal because we want to get the lane more exactly than uniform B-spline method does.

Non-uniform B-spline curve, denote as C, can be constructed by a set of $(n+1)$ control points $\{P_0, P_1, P_2, \cdots, P_n\}$, where $P_i(x_i, y_i), i = \overline{0,n}$. The points lie on curve C, are calculated from Equation 5:

$$C(u) = \sum_{i=0}^{n} B_{i,m}(u) P_i \quad (5)$$

where $t_{\min} \le u \le t_{\max}$, $2 \le m \le n+1$

$B_{i,m}(u)$ is called a basis B-spline that is a $(m-1)^{\text{th}}$ order polynomial function and defined as Equation 6:

$$B_{i,m}(u) = \begin{cases} \begin{cases} 1, t_i \le u \le t_{i+1} \\ 0, u \in [t_i, t_{i+1}] \end{cases}, & m=1 \\ \dfrac{u-t_i}{t_{i+m-1}-t_i} B_{i,m-1}(u) + \dfrac{t_{i+m}-u}{t_{i+m}-t_{i+1}} B_{i+1,m-1}(u) & ,m>1 \end{cases} \quad (6)$$

Considering Equation 6 to obtain value of $B_{i,m}(u)$, we need to specify a set $T = \{t_0, t_1, \cdots, t_{i+m}\}$ are knot vector and are $t_j, j = \overline{0, i+m}$ knot points.

When these knots are equidistant we say the B-spline is uniform otherwise is non-uniform. The more control points and higher order polynomial function, the more precise curve can be constructed. In our approach, we consider the following conditions:

(a). If we have more than 4 control points lay on the lane, we use NUBS interpolation with 3rd-order polynomial function for constructing the road lane boundaries,

(b). If we have 3 control points lay on the lane, we use NUBS interpolation with 2nd-order polynomial function for constructing the road lane boundaries.

4. LANE DETECTION ALGORITHM

4.1 Edge feature extraction

In constructing road land based on NUBS interpolation method, we need to find these control points for NUBS calculation and processing. Control point is position that intersects between the scan line and the most right lane (if we consider left lane) and most left lane (if we consider right lane). The lane model is described in Figure 5.

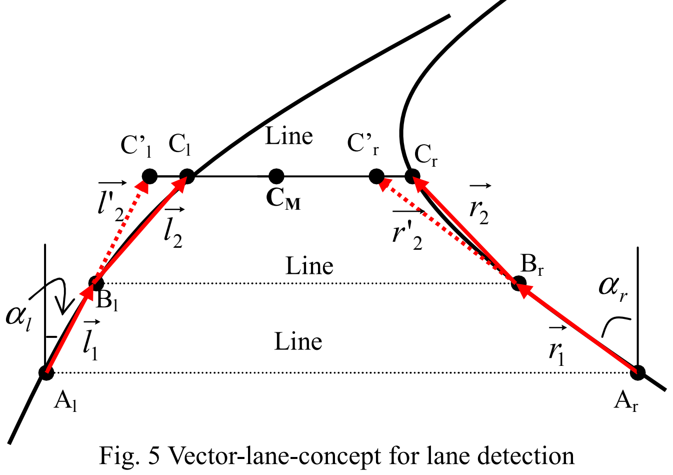


Fig. 5 Vector-lane-concept for lane detection

In this section, we propose new formulation that is called vector-lane concept to extract control points for NUBS interpolation processing. At the initialize step, we set up two scanning lines at the bottom of image space then find 2 first control points for each line, the left and right control points. The starting position for control-point scanning is selected as the central point for each scanning line. Then construct left lane vector $\vec{l}_1 = \overrightarrow{A_l B_l}$, right lane vector $\vec{r}_1 = \overrightarrow{A_r B_r}$, and calculate their slope angles α_l and α_r using formula (7) and (8):

Left lane: $$\alpha_l = arctg\left(\frac{x_{A_l} - x_{B_l}}{y_{A_l} - y_{B_l}}\right) \qquad (7)$$

Right lane: $$\alpha_r = arctg\left(\frac{x_{A_r} - x_{B_r}}{y_{A_r} - y_{Br}}\right) \qquad (8)$$

Using the parameters defined above, the control points of each lane for NUBS interpolation can be found as following steps:

Step 1: Divide the reminder of image space into 4 parts (based on the lane is longer length) via using 3 horizontal scan lines,

Step 2: Calculate position of $C_l^{'}$ based on vector lane $\vec{l}_2^{'}$, which is a extension vector along the vector $\vec{l}_1$.

Step 3: Calculate position of $C_r^{'}$ based on vector lane $\vec{r}_2^{'}$, which is a extension vector along the vector $\vec{r}_1$.

Step 4: Find the position of C_M the central point between $C_l^{'}$ and $C_r^{'}$.

Step 5: Scan from C_M to left for finding control point on the left lane and get C_l. If we can not find the control point (lost information because of noise), we will use $C_l^{'}$ instead of using C_l as shown in figure 5.

Step 6: Scan from C_M to right for finding control point on the right lane and get C_r. If we can not find the control point (lost information because of noise), we will use $C_r^{'}$ instead of using C_r.

4.2 Construct road lane boundaries

As mentioned earlier, based on these control points obtained in the last step, the algorithm will construct the left and right lane boundaries using NUBS interpolation method as following:

If we got more than 4 control points lay on the lane, we use NUBS interpolation with 3rd-order polynomial function for constructing that road lane boundary,

If we got 3 control points lay on the lane, we use NUBS interpolation with 2nd-order polynomial function for constructing that road lane boundary.

Because of the length of lane taken by camera is not balance, we choose the longer lane to divide the reminder of image. It is ensure that, we can detect almost all the lane. One example of proposed method is described in Figure 6.

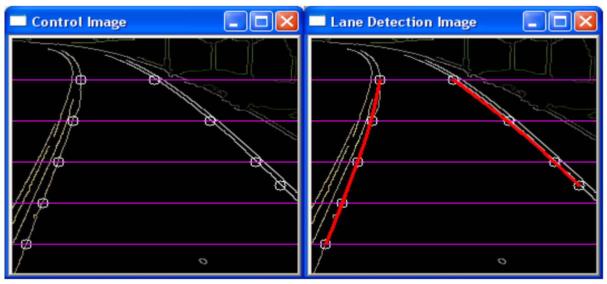


Fig. 6 Lane detection using Vector-lane-concept

To overcome the problem that some noises happen between two lanes, we set one preceding DELTA_ANGLE value. DELTA_ANGLE is the difference between two consecutive angles that belong to left or right lane. If DELTA_ANGLE value less than 15 degree we will continue go to left or right to find another control point for that lane. One example result of this method is demonstrated in Figure 7.

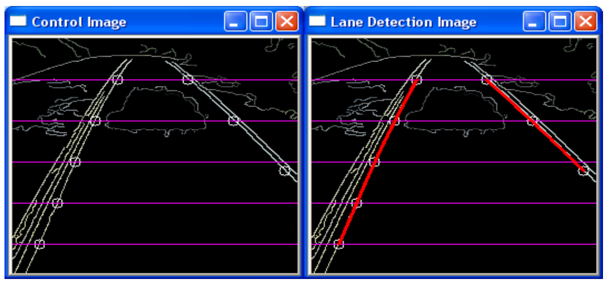


Fig. 7 Lane detection using Vector-lane-concept helps overcome noise problem.

5. ROAD LANE CURVATURE'S ESTIMATION AND LANE TRACKING

5.1 Road lane curvature's estimation

After lane boundaries detected, our next goal is to extract information to compute the directions and track the road lanes marking. To estimate the left and right lanes curvatures, we introduce the following steps:

Step 1: Using four scan lines from the bottom of image as shown in Figure 8 to calculate 3 angles for each lane.

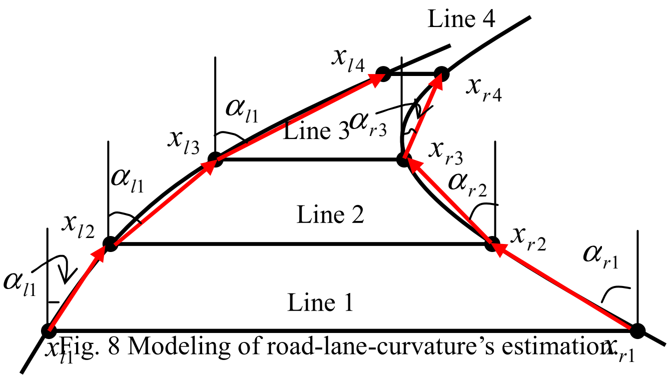


Fig. 8 Modeling of road-lane-curvature's estimation

(a) $\alpha_{l1}, \alpha_{l2}, \alpha_{l3}$ for left lane,

(b) $\alpha_{r1}, \alpha_{r2}, \alpha_{r3}$ for right lane.

Where the values of α_{li} and α_{ri}, $i = 1,2,3$ are calculated using Equation 9 and 10:

$$\alpha_{li} = arctg\left(\frac{x_{l(i+1)} - x_{li}}{y_{l(i+1)} - y_{li}}\right), \quad i = \overline{1,3} \tag{9}$$

$$\alpha_{ri} = arctg\left(\frac{x_{r(i+1)} - x_{ri}}{y_{r(i+1)} - y_{ri}}\right), \quad i = \overline{1,3} \tag{10}$$

Step 2: Calculate the different between these angles:

Step 2.1: For left lane:

$$\Delta\alpha_{l1} = \alpha_{l2} - \alpha_{l1} \text{ and } \Delta\alpha_{l2} = \alpha_{l3} - \alpha_{l2} \tag{11}$$

Step 2.2: For right lane:

$$\Delta\alpha_{r1} = \alpha_{r2} - \alpha_{r1} \text{ and } \Delta\alpha_{r2} = \alpha_{r3} - \alpha_{r2} \tag{12}$$

Step 3: Calculate the mean value of these angles:

Step 3.1: For left lane:

$$\alpha_l = a_l \Delta\alpha_{l2} + b_l \Delta\alpha_{l1} + c_l \alpha_{l1} \tag{13}$$

Step 3.2: For right lane:

$$\alpha_r = a_r \Delta\alpha_{r2} + b_r \Delta\alpha_{r1} + c_r \alpha_{r1} \tag{14}$$

Step 3.3: Choose the sign of these coefficients a_l, b_l, c_l using Equations 15a, 15b and 15c bellow:

$$\begin{cases} a_l \geq 0, if\ \Delta\alpha_{l2} \geq 0 \\ a_l < 0, if\ \Delta\alpha_{l2} < 0 \end{cases} \tag{15a}$$

$$\begin{cases} b_l \geq 0, if\ \Delta\alpha_{l1} \geq 0 \\ b_l < 0, if\ \Delta\alpha_{l1} < 0 \end{cases} \tag{15b}$$

$$\begin{cases} c_l \geq 0, if\ \alpha_{l1} \geq 0 \\ c_l < 0, if\ \alpha_{l1} < 0 \end{cases} \tag{15c}$$

Step 3.4: Choose the sign of these coefficients a_r, b_r, c_r using Equations 16a, 16b and 16c bellow:

$$\begin{cases} a_r \geq 0, if\ \Delta\alpha_{r2} \geq 0 \\ a_r < 0, if\ \Delta\alpha_{r2} < 0 \end{cases} \tag{16a}$$

$$\begin{cases} b_r \geq 0, if\ \Delta\alpha_{r1} \geq 0 \\ b_r < 0, if\ \Delta\alpha_{r1} < 0 \end{cases} \tag{16b}$$

$$\begin{cases} c_r \geq 0, if\ \alpha_{r1} \geq 0 \\ c_r < 0, if\ \alpha_{r1} < 0 \end{cases} \tag{16c}$$

The absolute value of $a_l, b_l, c_l, a_r, b_r, c_r$ are chosen depend upon real experiment. In our work, we set $|a_l| = |a_r| = 3$, $|b_l| = |b_r| = 2$ and $|c_l| = |c_r| = 1$.

α_l calculated from Equation 13 is the left-lane curvature, α_r calculated from Equation 14 is the right-lane curvature.

5.2 Lane tracking

In this sub-section we present one possible approach to extend the functionality of the lane-marking detection to tracking. The purpose of tracking is twofold: First, detection of the lane in previous frames should limit the search region for the lane in the current frame, thus reducing running time. Second, the information from previous frames serves as a consistency measure of the results of the current frame. This helps to prune wrong hypotheses. The geometric information the algorithm passed from frame to frame is the vector-lane-concept or the curvature of lane boundaries.

In the case of continuous lane-marking we utilize these values of left-lane and right-lane curvature that obtained in the last step of previous frame to construct the first vector-lane-concept for current frame and loop run edge feature extraction algorithm that presented in sub-section 4.1 from step 1 to step 6 to find these control points and recalculate values of left-lane and right-lane curvature.

In the case of non-continuous lane-marking we also utilize these values of left-lane and right-lane curvature that obtained in the last step of previous frame to construct the vector-lane-concept for current frame until we can get the new control point of lane-marking. Next, we construct new vector-lane-concept and recalculate values of left-lane and right-lane curvature step by step. Assume that, these values of left-lane curvature is $\alpha_{LeftOld}$ and the new value of left-lane curvature is $\alpha_{LeftNew}$, then we update the left-lane curvature value as follows:

$$\alpha_{LeftUpdate} = \frac{\alpha_{\text{LeftOld}} + \alpha_{\text{LeftNew}}}{2} \tag{17}$$

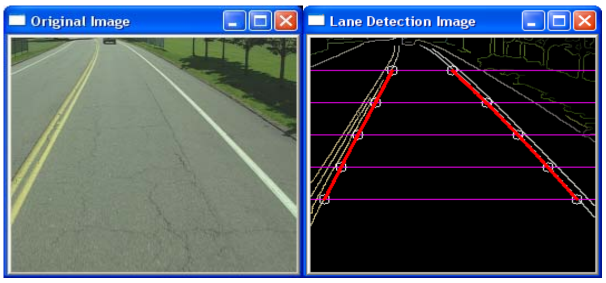
The same with right-lane boundary, we have:

$$\alpha_{RightUpdate} = \frac{\alpha_{\text{RightOld}} + \alpha_{\text{RightNew}}}{2} \tag{18}$$

6. SIMULATION RESULTS

In this section, we present some exemplary simulation result of our algorithm for lane boundary detection. The result was analyzed and simulated on Visual C++ program combine with Open CV tool and has been tested on some sequences of images grabbed by an on-board camera at different locations and at different times. Figure 9 below show some of our experimental results of lane boundary detection where estimated lane boundaries use vector-lane-concept. The images presented in this simulation are 256x240 color image and can be downloaded from the website:

http://vasc.ri.cmu.edu//idb/html/road/may30_90/index.html.



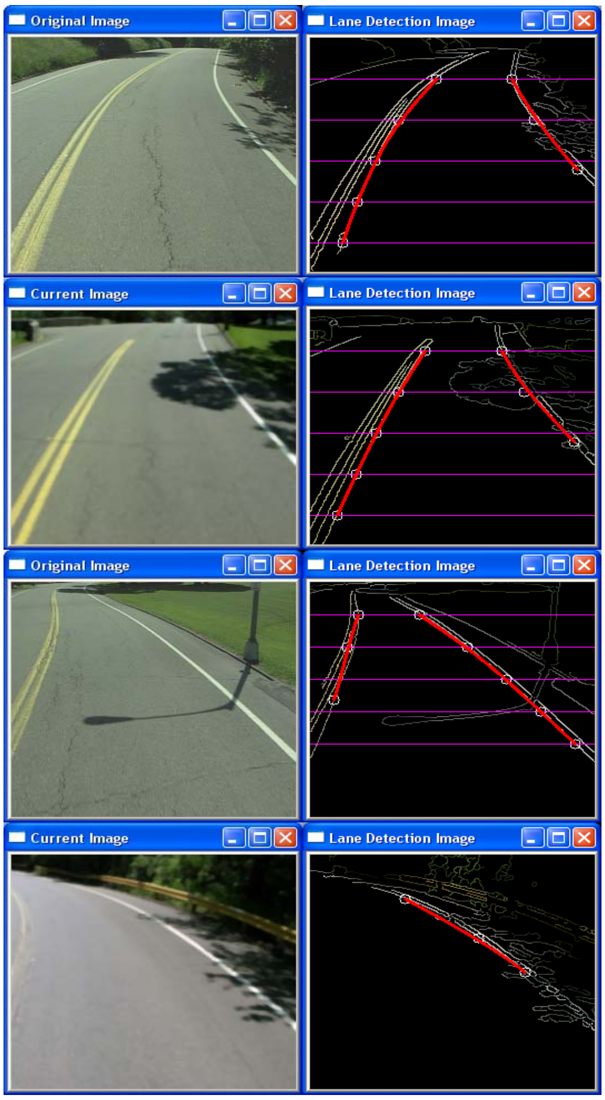


Fig. 9 Some example results of lane detection test

7. CONCLUSION

As shown in the above examples, our lane-detection algorithm well detects the lane marking line well even in the cases of severe noises. Therefore, the proposed algorithm can be applied to Autonomous Guided Vehicles (AGV), or driver-assistant applications of vehicles. In near future, we will apply some intelligent techniques such as fuzzy or neuron-network control to develop a mobile robot and test the AGV system with obstacle avoidance condition in real world roads.

ACKNOWLEDGEMENT

This work was supported by the 2007 Research Fund of University of Ulsan.

REFERENCES

[1] Tie Liu, Nanning Zheng, Hong Cheng, Zhengbei Xing, "A Novel Approach of Road Recognition Based on Deformable Template and Genetic Algorithm", International Journal, IEEE 2003, pp.1251–1256.

[2] Yue Wang, Eam Khwang Teoh, Dinggang Shen. "Lane detection and tracking using B-snake", Image and Vision Computing, powered by Science Direct 2004, pp. 269–280.

[3] Seung Gweon Jeong, Chang Sup Kim, Dong Youp Lee, Sung Ki Ha, Dong Hwal Lee, Man Hyung Lee, Hideki Hashimoto. "Real-time lane detection for autonomous vehicle", ISIE, Pusan, KOREA, 2001, pp. 1466–1471.

[4] M.Asif, M.R.Arshad, and P.A.Wilson, "AGV Guidance System: An Application of Simple Active Contour for Visual Tracking", Proceeding world academy of science, engineering and technology Vol. 6, 2005, pp. 74 –77.

[5] M.Boumediene, A. Ouamri, and N. Dahnoun, "Lane Boundary Detection and Tracking using NNF and HMM Approaches", Proceeding 2007 IEEE Intelligent Vehicles Symposium - Istanbul, Turkey, 2007, pp. 1107 –1111.

[6] H. E. Lu, P. S. P. Wang, "A Comment on A Fast Parallel Algorithm for Thinning Digital Patterns", Communication of the ACM, Vol.29, 1986, pp. 239 –242.

[7] Erin Hasting. "A survey of thinning methodologies", College of Engineering and Computer Science, University of Central Florida4000.

[8] N. Chihab, A. Zergahoh, J-P. Astruc, "Generalized non-uniform B-spline functions for discrete signal interpolation", IEEE Journal, 2003, pp. 129 –132.

[9] Gershon Elber, Craig Gotsman, "Multi-resolution control for B-spline curve editing", CRC Israel Science Center, Technion city, Haifa 32000, Israel, 1995.