References

[1] W.-Y. Shieh, W.-H. Lee, S.-L. Tung, and C.-D. Ho, "A novel architecture for multilane-free-flow electronic-toll-collection systems in the millimeter-wave range," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 3, pp. 294–301, Sep. 2005.

[2] G. K. H. Pang and H. H. S. Liu, "LED location beacon system based on processing of digital images," *IEEE Trans. Intell. Transp. Syst.*, vol. 2, no. 3, pp. 135–150, Sep. 2001.

[3] M. Akanegawa, Y. Tanaka, and M. Nakagawa, "Basic study on traffic information system using LED traffic lights," *IEEE Trans. Intell. Transp. Syst.*, vol. 2, no. 4, pp. 197–203, Dec. 2001.

[4] J. S. Kwak and J. H. Lee, "Infrared transmission for intervehicle ranging and vehicle-to-roadside communication systems using spread-spectrum technique," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 1, pp. 12–19, Mar. 2004.

[5] A. Visser, H. H. Yakah, A. J. van der Wees, M. Oud, G. A. van der Spek, and L. O. Herzberger, "A hierarchical view on modeling the reliability of a DSRC link for ETC applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 2, pp. 120–129, Jun. 2002.

[6] B. Edde, *Radar—Principles, Technology, Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1993, ch. 1.

[7] M. I. Skolnik, *Introduction to Radar Systems*, 2nd ed. New York: McGraw-Hill, 1980, ch. 1.

[8] J. M. Kahn and J. R. Barry, "Wireless infrared communications," *Proc. IEEE*, vol. 85, no. 2, pp. 265–298, Feb. 1997.

[9] W.-Y. Shieh, "The analysis of single-lane OBU and RTU optimum setup configuration in automatic electronic-toll-collection systems," Res. Inst. ChungHwa Telecom, Taoyuan, Taiwan, 89-EC-023, Nov. 2000.

[10] ——, "The comparison of the efficacy of two-piece and single-piece RTU LED-module on the down-link data transmission in our ETC system," Res. Inst. ChungHwa Telecom, Taoyuan, Taiwan, 90-EC-001, Jan. 2001.

[11] J. D. Kraus, *Antennas*, 2nd ed. New York: McGraw-Hill, 1988, ch. 3.

[12] *Datasheet of TSHF5400 High Speed IR Emitting Diode in ⊘ 5 mm* $(\mathrm{T}-1(3/4))$ *Package*, Jun. 2003. The Vishay website, 81024.pdf. [Online]. Available: http://www.vishay.com/docs/81024/

[13] *Datasheet of Silicon PIN Photodiode with Daylight Filter*, Mar. 2004. The OSRAM website, bpw34fa.pdf. [Online]. Available: http://www.osram.convergy.de/upload/documents/2004/03/10/15/15/

Lane Detection With Moving Vehicles in the Traffic Scenes

Hsu-Yung Cheng, Bor-Shenn Jeng, Pei-Ting Tseng, and Kuo-Chin Fan

***Abstract*—A lane-detection method aimed at handling moving vehicles in the traffic scenes is proposed in this brief. First, lane marks are extracted based on color information. The extraction of lane-mark colors is designed in a way that is not affected by illumination changes and the proportion of space that vehicles on the road occupy. Next, for vehicles that have the same colors as the lane marks, we utilize size, shape, and motion information to distinguish them from the real lane marks. The mechanism effectively eliminates the influence of passing vehicles when performing lane detection. Finally, pixels in the extracted lane-mark mask are accumulated to find the boundary lines of the lane. The proposed algorithm is able to robustly find the left and right boundary lines of the lane and is not affected by the passing traffic. Experimental results show that the proposed method works well on marked roads in various lighting conditions.**

***Index Terms*—Camera geometry, computer vision, intelligent transportation systems (ITS), intelligent vehicles, lane detection.**

I. Introduction

In intelligent transportation systems, intelligent vehicles cooperate with smart infrastructure to achieve a safer environment and better traffic conditions [1]. Intelligent vehicles are expected to be able to give route directions, sense objects or pedestrians, prevent impending collisions, or warn drivers of lane departure [2], [3]. Therefore, lane detection is a crucial element for developing intelligent vehicles. Lane detection based on machine vision is accomplished by taking images from cameras mounted on the intelligent vehicles. There are many related research works on this issue in recent years [4]–[17]. These works generally used different strategies aimed at certain kinds of surroundings and road conditions. One category of methods used intensity images as the basis of lane detection. Kluge and Lakshmanan [4] used a deformable template model of lane structure to locate lane boundaries without thresholding the intensity gradient information. Wang *et al.* [5], [6] computed a potential edge field and a potential orientation field image and then applied B-snake or Catmull-Rom spline-based lane model to handle curved roads. Yim and Oh [7] developed a three-feature-based automatic lane-detection algorithm using the starting position, direction, and gray-level value of a lane boundary as features to recognize the lane. Another category of works combined edge information with color information or other features to distinguish the road area from the surroundings. Gibbs and Thomas [8] fused the results of edge detection, road segmentation, and white lane follower to obtain the final detection result. Rasmussen [9] utilized texture features to deal with rural roads that do not have obvious lane marks, under the assumption that the color of road surface is homogeneous. Kluge and coauthor [10], [11] and Gonzalez and Ozguner [12] employed histogram information to select thresholds and detect lane marks. Redmill *et al.* [13] utilized a matched filter and a Kalman filter

Manuscript received July 17, 2005; revised December 9, 2005, June 26, 2006, July 31, 2006, and August 2, 2006. The work of H.-Y. Cheng was supported by the MediaTek Fellowship. The Associate Editor for this paper was L. Vlacic.

H.-Y. Cheng, P.-T. Tseng, and K.-C. Fan are with the Department of Computer Science and Information Engineering, National Central University, Chung-Li, Taiwan, R.O.C. (e-mail: chengsy@fox1.csie.ncu.edu.tw; gf1022@yahoo.com.tw; kcfan@csie.ncu.edu.tw).

B.-S. Jeng is with the Department of Communications Engineering, Yuan-Ze University, Chung-Li 320, Taoyuan, Taiwan, R.O.C. (e-mail: bsjip@cht.com.tw).

Digital Object Identifier 10.1109/TITS.2006.883940

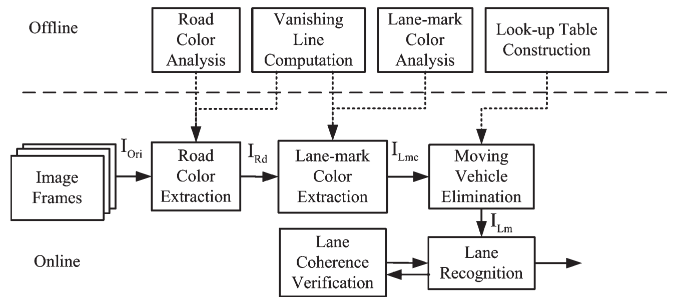

Fig. 1. System block diagram.

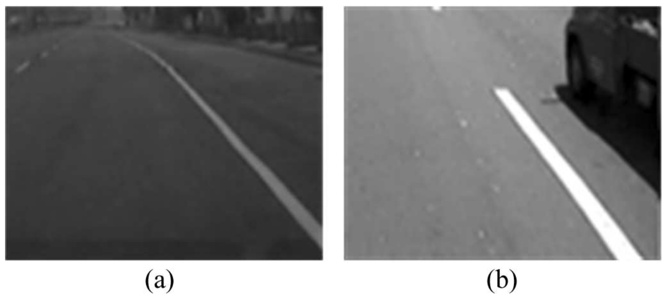

Fig. 2. Lane-mark colors under different lighting conditions.

for lane detection. Many of these works are efficient and exhibited adequate robustness on roads clear of other vehicles. However, few of these papers dealt with the problem that moving vehicles in the traffic scenes bring about. Some of them mentioned the problem in the introduction section but did not give a sufficiently effective solution, and displayed experimental results without vehicles. He *et al.* [14] proposed a color-based road-detection method aiming at urban traffic scenes. In their work, they emphasized the importance of color information by mentioning that the edges of shadows, vehicles, pedestrians, etc., can hopelessly clutter the tru e road boundaries as they will appear as edges in the intensity images. However, their road-area extraction module used every pixel between two chosen road boundary candidates to compute the mean vector and covariance matrix of a multivariable Gaussian distribution. This scheme works only under the assumption that cars, pedestrians, and road markings on the road cover only a very small ratio of the road surface. If some vehicles occupy more than a certain amount of space between the two chosen boundary candidates, the mean vector and covariance matrix would inevitably be biased, and the accuracy of the final lane-detection results would be affected. Other papers integrated camera-specific parameters to provide further information to reconstruct the road. Onoguchi *et al.* [15] proposed a planar projection stereopsis (PPS) method that utilized height information to distinguish the road from the surroundings. This method can resolve roadside obstacles such as parked cars. However, in many environments, the borders between the road area and surroundings consist of only lane marks or very low road shoulders, whose heights above the road surface are too small to ensure the success of PPS.

In this brief, we propose an accurate lane-detection scheme that is not disturbed by the surrounding vehicles. We assume that the roads to be handled are tarred and marked with white, yellow, or red lane marks. This assumption is true to most of the highways and main roads in cities. Traffic tends to be busier on these kinds of roads, making the ability to deal with moving vehicles necessary for intelligent vehicles.

The proposed system block diagram is illustrated in Fig. 1. Because moving vehicles often appear in the camera scene and can fill the picture with unwanted edges, lane detection based on edges is not preferred in this situation. Therefore, we utilize color information as the basis of our lane-detection algorithm. By detecting the colors of the lane marks, which can only be white, yellow, or red, we can eliminate cars with different colors. Road colors and lane-mark colors in different lighting conditions are analyzed in advance to provide necessary information for the road and lane-mark color extraction modules. The vertical position of the vanishing line in an image frame is also computed in advance through camera calibration parameters. The vanishing line is the horizon of an image. Parallel lines receding from an observer converge at the vanishing point on this horizontal line. We do not have to process the area above the vanishing line in an image frame.

The purpose of road-color extraction is not to find the exact area of the road surface but to help us find the colors of the lane marks more accurately. The reason this module is necessary is that, in different lighting conditions, the colors of the lane marks are very different due to illumination changes. By carefully examining images taken in different weather, we can observe that a "white" lane mark in an image taken in a cloudy day may appear "gray" compared to an image taken in a sunny day, as shown in Fig. 2. If we consider only "white" colors within a narrow range, the lane marks in Fig. 2(a) would not be detected. If we consider the "white" colors within a wide illumination range, many pixels on the road surface in Fig. 2(b) would be classified as lane marks. Therefore, we solve this problem by trying to extract the road colors first and then find the reasonable lane-mark colors, in contrast with the average intensity of the road-color areas. The procedure for finding road and lane-mark colors is explained in Section II.

The lane-mark color extraction procedure results in a binary mask I_{Lmc}, highlighting possible regions of the lane marks in an image. When extracting lane marks based on their colors, we would inevitably extract some vehicles such as white, gray, yellow, or red cars at the same time. Vehicles with the same colors as lane marks are the factors that are most likely to cause confusion in color-based lane-detection schemes. To differentiate these nonlane-mark regions from the real lane marks, we perform a moving vehicle elimination procedure. In this procedure, size and shape information is first used to determine whether a region belongs to a vehicle or a lane mark. If we are sure that a region cannot belong to a lane mark, the region is eliminated from the mask. This first elimination process is conservatively designed in order not to eliminate real lane marks. The remaining regions in the mask might still contain nonlane-mark regions, which mostly result from light reflections from car windows or fragments of larger vehicles. Since these regions have different motion patterns compared to the motion pattern of the lane marks in a video, we can further eliminate these regions using interframe motion information and speed information provided by the intelligent vehicle. The moving vehicle elimination procedure is elaborated upon in Section III.

After eliminating moving vehicles and preserving the real lane-mark regions, we can perform the final lane-recognition procedure. An algorithm based on accumulating pixels in the extracted lane-mark mask is designed to mark the boundary lines of the lane on which the intelligent vehicle is driving. The previous lane boundary lines are stored in the lane-coherence-verification module. The lane-coherence-verification module helps the lane-recognition module to check the accuracy of the newly detected lane boundaries by examining the lane width and the orientation of the lane boundaries. If the newly detected boundary lines result in an unrealistic lane width compared to the previous lane boundaries, they would be invalidated, and the previous lane boundaries would be reused. The lane-coherence-verification module can ensure the correctness of the lane-detection results when the lane

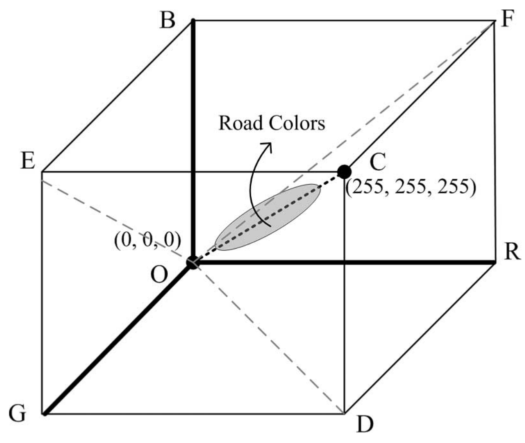

Fig. 3. Distribution of road colors.

marks are temporarily occluded by the passing vehicles. The lane-coherence-verification and lane-recognition modules are explained in Section IV. In Section V, several experimental results are displayed and explained. Finally, a conclusion is made in Section VI.

II. Road and Lane-Mark Color Extraction

By analyzing road colors under different lighting conditions, we can observe that the distribution of road colors is located in the narrow region around line segment $\overline{OC}$, as illustrated in Fig. 3. We change the color components RGB to D_1, D_2, and D_3 by letting

$$\begin{aligned} D_1 &= |R-B| \\ D_2 &= |B-G| \\ D_3 &= |R-G|. \end{aligned} \qquad (1)$$

We can see that the colors on line segments $\overline{OD}$, $\overline{OE}$, and $\overline{OF}$ will be mapped to the zero points on D_1, D_2, and D_3 components, respectively. Therefore, the conversion from RGB to $D_1D_2D_3$ will discard some color information, but the $D_1D_2D_3$ components are more suitable for us to determine if a color belongs to road colors.

To detect the road colors, we first find a wide range of gray colors by specifying some loose constraints on the D_1, D_2, and D_3 components, respectively. These first-step loose constraints are obtained by analyzing the road colors in different lighting conditions in advance. Then, the colors passing these loose constraints are analyzed again, and the mean values and standard deviations of the three color components are computed. Suppose that the mean values and standard deviations are μ_{D_1}, μ_{D_2}, and μ_{D_3}, and σ_{D_1}, σ_{D_2}, and σ_{D_3}, respectively. Road colors are extracted based on examining each pixel (u,ν) with the following rule:

$$I_{\mathrm{Rd}}(u,\nu) = \begin{cases} I_{\mathrm{Ori}}(u,\nu), & \text{if } \mu_{D_1}-f\sigma_{D_1}<D_1(u,\nu)<\mu_{D_1}+f\sigma_{D_1} \\ & \text{AND } \mu_{D_2}-f\sigma_{D_2}<D_2(u,\nu)<\mu_{D_2}+f\sigma_{D_2} \\ & \text{AND } \mu_{D_3}-f\sigma_{D_3}<D_3(u,\nu)<\mu_{D_3}+f\sigma_{D_3} \\ 0, & \text{otherwise} \end{cases} \qquad (2)$$

where f is an adjusting factor, I_{Ori} is the original image frame, and I_{Rd} is the result of the road-color extraction procedure.

Lane-mark colors are detected with the help of both $D_1D_2D_3$ and $YCbCr$ color components for we need to extract these colors more accurately. Similar to the road colors stated above, lane-mark colors are first detected by specifying some loose constraints on the D_1, D_2, and D_3 components, respectively, with the enforcement of an additional constraint $Y > \hat{\mu}_Y + \hat{f}\hat{\sigma}_Y$. $\hat{\mu}_Y$ and $\hat{\sigma}_Y$ are the mean value and standard deviation of the Y color component of the colors that have nonzero values in I_{Rd}, and $\hat{f}$ is another adjusting factor. The additional constraint ensures that the intensities of the detected colors are relatively higher compared to the average intensity of the road surface in the same image frame. Again, the loose constraints are obtained by analyzing the lane-mark colors in different lighting conditions in advance. Note that the constraints for detecting white, yellow, and red colors are all different, so we detect these three main classes of lane-mark colors separately.

Then, we use three multivariable Gaussian distributions to represent the three main classes of lane-mark colors. $P_{\mathrm{white}}(I_{\mathrm{Ori}}(u,\nu))$ represents the probability that a pixel (u,ν) belongs to the class of white lane-mark colors

$$P_{\mathrm{white}}\left(I_{\mathrm{Ori}}(u,\nu)\right) = \frac{1}{(2\pi)^{3/2}|\sum_{\mathrm{white}}|^{1/2}} \times e^{-\frac{1}{2}(I_{\mathrm{Ori}}(u,\nu)-\mu_{\mathrm{white}})^T \sum_{\mathrm{white}}^{-1}(I_{\mathrm{Ori}}(u,\nu)-\mu_{\mathrm{white}})} \qquad (3)$$

where $\hat{\mu}_{\mathrm{white}}$ and σ_{white} are the mean vector and covariance matrix of the $YCbCr$ components of the detected white lane-mark colors. The lane-mark color extraction procedure is completed by setting the mask I_{Lmc} according to the following rule:

$$I_{\mathrm{Lmc}}(u,\nu) = \begin{cases} 1, & \text{if } \; P_{\mathrm{white}}\left(I_{\mathrm{Ori}}(u,\nu)\right) > \mathrm{Thr}_{\mathrm{white}} \\ & \text{OR } \; P_{\mathrm{yellow}}\left(I_{\mathrm{Ori}}(u,\nu)\right) > \mathrm{Thr}_{\mathrm{yellow}} \\ & \text{OR } \; P_{\mathrm{red}}\left(I_{\mathrm{Ori}}(u,\nu)\right) > \mathrm{Thr}_{\mathrm{red}} \\ 0, & \text{otherwise} \end{cases} \qquad (4)$$

where $\mathrm{Thr}_{\mathrm{white}} = P_{\mathrm{white}}(\mu_{\mathrm{white}} + f_{\mathrm{white}} \sum_{\mathrm{white}})$, and f_{white} is an adjusting factor. P_{yellow}, P_{red}, $\mathrm{Thr}_{\mathrm{yellow}}$, and $\mathrm{Thr}_{\mathrm{red}}$ are similarly defined.

III. Moving Vehicle Elimination

The purpose of the moving vehicle elimination module is to distinguish the actual lane marks from vehicles that have the same colors as the lane marks. The principal idea is to distinguish the real lane marks and vehicles by their sizes and shapes first. Afterwards, the remaining undecided objects are further differentiated utilizing motion information. Because lane marks are fixed points on the ground plane and vehicles are not, their motion patterns are more different in the video. To improve system performance, we use size, shape, and motion information to find moving vehicles every k frames instead of every frame. The regions marked as moving vehicles are eliminated from the lane-mark masks I_{Lm} for the following $k-1$ frames. The procedure for moving vehicle elimination is illustrated in Fig. 4.

A. *Initialization*

Initially, there is no lane-mark mask I_{Lm} available for previous frames. Therefore, the initialization stage of the moving vehicle elimination procedure is a little bit different from the rest of the procedure. In the initialization stage, the tracking list of the $(k+1)$th frame is constructed following the same steps described in Section III-B. Afterwards, object matching, as described in Section III-D, is performed with the help of I_{Lmc} of the first frame. At this point, I_{Lm} of frame $k+1$ is obtained. Then, for the tracking points of all the objects in I_{Lm} of frame $k+1$, their new positions in frame $2k+1$ can be estimated according to Section III-E.

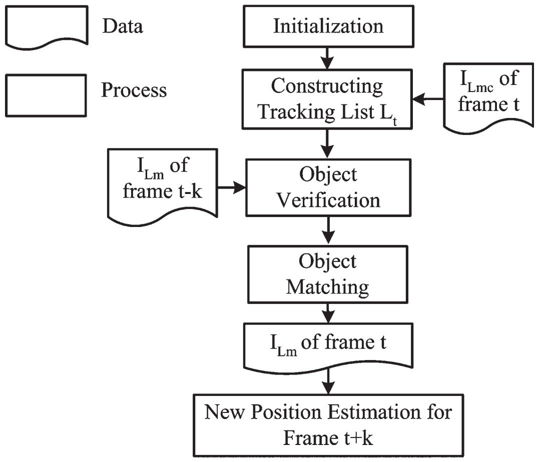

Fig. 4. Moving vehicle elimination procedure.

B. Constructing Tracking List

The steps for constructing the tracking list L_t are the following.

Step 1) Each connected component in I_{Lmc} of frame t is copied to I_{Lm} and the tracking list L_t initially. Note that each connected component is defined as an object.

Step 2) Objects that are smaller than a certain size are preserved in I_{Lm} but removed from L_t. The reason is that these small objects are not easily tracked and would not seriously affect the lane-detection result if they are not real lane marks.

Step 3) Objects with widths smaller than a threshold and objects that pass through a certain range of the entire frame are considered to be real lane marks and need not be tracked. Therefore, these objects are preserved in I_{Lm} and removed from L_t.

Step 4) Objects with widths larger than a threshold and heights smaller than a threshold are marked as vehicles and removed from L_t. These objects are also removed from I_{Lm}.

Step 5) For each object in L_t and I_{Lm}, four corners of the object are found to serve as the tracking points of the object.

After performing steps 2)–4), the tracking list L_t would contain only a few objects that cannot be obviously classified as vehicles or real lane-mark regions by simple size or shape information. These objects need to be tracked and classified using further motion information between frames.

C. Object Verification

For a tracking point (u_{t-k}, ν_{t-k}) in I_{Lm} of frame $t-k$, if we have already predicted its position in frame t following formula (15) presented in Section III-E, we can check whether the tracking points in tracking list L_t are close to the positions we have predicted. If the predicted tracking points match the tracking points of the objects in L_t, we know that these objects obey the motion pattern of real lane marks. Supposing that the estimated position for (u_{t-k}, ν_{t-k}) is $(\tilde{u}_t, \tilde{\nu}_t)$ in frame t, we compute the similarity S_0 between (u_{t-k}, ν_{t-k}) and $(\tilde{u}_t, \tilde{\nu}_t)$ with the following formula:

$$S_0 = \sum_{i=-1}^{1} \sum_{j=-1}^{1} (u_{t-k}+i, \nu_{t-k}+j) \text{ XNOR } (\tilde{u}_t+i, \tilde{\nu}_t+j). \quad (5)$$

Similarly, the similarities S_1–S_8 are computed between (u_{t-k}, ν_{t-k}) and the eight neighbors N_1–N_8 of $(\tilde{u}_t, \tilde{\nu}_t)$, respectively, as illustrated in Fig. 5. A tracking point (u_t, ν_t) in L_t is classified as a fixed point on the ground plane if (u_t, ν_t) belongs to one of the eight neighbors N_i and $S_i > \mathrm{Tr}, i = 1, \ldots, 8$. Tr is a threshold. An object is verified as a real lane-mark region if more than two tracking points belonging to this object are classified as fixed points on the ground plane. The verified objects are preserved in I_{Lm} of frame t and removed from tracking list L_t. In addition, the displacements between the verified tracking points (u_t, ν_t) and the corresponding points (u_{t-k}, ν_{t-k}) are recorded as the velocities of the tracking points.

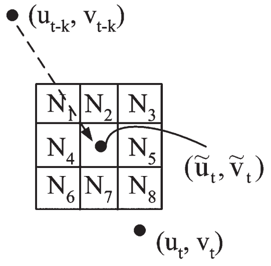

Fig. 5. Object verification.

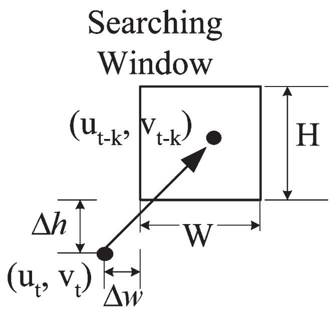

Fig. 6. Searching window for object matching.

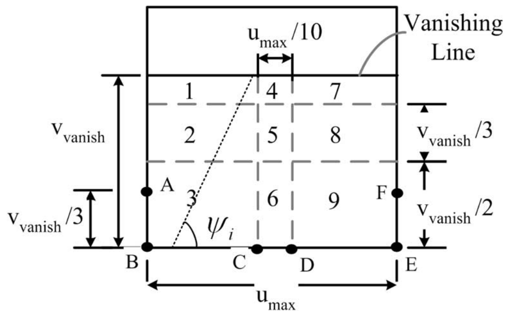

Fig. 7. Nine areas defined for different sizes and relative positions of searching windows.

D. Object Matching

There are two possibilities that objects in L_t remain unverified. One possibility is that these objects do not obey the motion pattern of real lane marks, and thus, their tracking points do not match the predicted tracking points. Another possibility is that the estimated positions for the tracking points of these objects are not yet available. For the latter case, we perform object matching to find the initial velocity of each tracking point so that its new position can be estimated. For a tracking point (u_t, ν_t) in L_t, we search for its previous matching tracking point (u_{t-k}, ν_{t-k}) in a search window, as illustrated in Fig. 6. The size and the relative position of the search window are adjusted according to camera and road geometry. We divide the entire area below the vanishing line in an image frame into nine areas, as illustrated in Fig. 7. The width and height as well as the relative position Δw and Δh of the searching window are set according to Table I, depending on which

TABLE I
SIZES AND RELATIVE POSITIONS OF THE SEARCHING WINDOW

Area	W	H	Δw	Δh
1	$\alpha/4$	$\beta/4$	0	0
2	$\alpha/2$	$\beta/2$	1	1
3	α	β	3	3
4	$\alpha/4$	$\beta/4$	-W/2	0
5	$\alpha/2$	$\beta/2$	-W/2	1
6	α	β	-W/2	3
7	$\alpha/4$	$\beta/4$	-W	0
8	$\alpha/2$	$\beta/2$	-W-1	1
9	α	β	-W-3	3

area to which (u_t, ν_t) belongs. Objects are matched if more than two of its tracking points are matched.

In Table I, $\alpha \propto$ velocity $\cdot k \cdot u_{\max}$/frame rate, and $\beta \propto$ velocity $\cdot k \cdot \nu_{\text{vanish}}$/frame rate. We can use a lookup table constructed in advance to obtain α and β for different velocities. The lookup table construction module illustrated in Fig. 1 is responsible for constructing the lookup table for α and β offline. Objects in L_t that cannot be matched are deleted from L_t and I_{Lm}. Note that, when performing object verification and matching procedures, we always take into account the neighboring pixels of a tracking point. Also, the threshold Tr is selected so that slight mismatching between tracking points can be tolerated. Therefore, the algorithm can cope with minor vibration. To deal with significant camera vibration, a video stabilization system described in [18] could be applied first.

E. New Position Estimation

To estimate the new positions of the tracking points in frame $t+k$, we first analyze the motion pattern of the lane marks. As the vehicle carrying the camera moves forward along the lane, the relative velocity between the vehicle and the road makes a person on the vehicle feel that the lane marks on the road are moving toward him. Therefore, we can view the situation this way: The camera is fixed, and the points on the ground plane are moving backward toward the camera. As shown in Fig. 8, supposing that a point is located at (X_{t-k}, Y_{t-k}) at instant $t-k$ and moves to (X_t, Y_t) at instant t, the corresponding positions on the image plane would move from (u_{t-k}, ν_{t-k}) to (u_t, ν_t). Fig. 8 also defines the ground plane coordinate system XYZ, the camera coordinate system $X_cY_cZ_c$, the image plane coordinate system $u\nu$, and the UVW coordinate system. The UVW system is obtained by rotating an angle ϕ around the X axis [19]. Therefore

$$\begin{aligned} X_c &= U = X \\ Y_c &= W = Y\sin\phi + Z\cos\phi \\ Z_c &= -V - F = -(Y\cos\phi - Z\sin\phi) - F. \end{aligned} \tag{6}$$

According to pinhole camera model [20], the relations between the image plane coordinate system $u\nu$ and ground plane coordinate system XYZ are

$$\begin{aligned} u &= -f\frac{X_c}{Z_c} = -f\frac{X}{-(Y\cos\phi - Z\sin\phi) - F} \\ \nu &= -f\frac{Y_c}{Z_c} = -f\frac{Y\sin\phi + Z\cos\phi}{-(Y\cos\phi - Z\sin\phi) - F} \end{aligned} \tag{7}$$

where f is the focal length. If we consider only the points on the ground plane where $Z = 0$, (7) becomes

$$\begin{aligned} u &= f\frac{X}{Y\cos\phi + F} \\ \nu &= f\frac{Y\sin\phi}{Y\cos\phi + F}. \end{aligned} \tag{8}$$

Suppose (u_t, ν_t) is a pixel in the image frame t and that its corresponding ground plane coordinate is (X_t, Y_t). According to (8), we can express (X_t, Y_t) in terms of (u_t, ν_t)

$$\begin{aligned} X_t &= \frac{u_t \sin\phi F}{f\sin\phi - \nu_t\cos\phi} \\ Y_t &= \frac{\nu_t F}{f\sin\phi - \nu_t\cos\phi}. \end{aligned} \tag{9}$$

By taking the partial derivative of X_t and Y_t with respect to t, we can get the velocities of X direction and Y direction at instant t, which are denoted $\dot{X}_t$ and $\dot{Y}_t$, respectively

$$\begin{aligned} \dot{X}_t &= \frac{\partial X_t}{\partial t} = \frac{\partial}{\partial t}\left[u_t\sin\phi F(f\sin\phi - \nu_t\cos\phi)^{-1}\right] \\ &= \frac{\dot{u}_t F f\sin^2\phi - \dot{u}_t\nu_t\sin\phi\cos\phi F + u_t\dot{\nu}_t\sin\phi\cos\phi F}{(f\sin\phi - \nu_t\cos\phi)^2} \end{aligned} \tag{10}$$

$$\begin{aligned} \dot{Y}_t &= \frac{\partial Y_t}{\partial t} = \frac{\partial}{\partial t}\left[\nu_t F(f\sin\phi - \nu_t\cos\phi)^{-1}\right] \\ &= \frac{\dot{\nu}_t F f\sin\phi}{(f\sin\phi - \nu_t\cos\phi)^2}. \end{aligned} \tag{11}$$

Assuming that the velocity of the intelligent vehicle remains unchanged within the short period from instant $t-k$ to instant t, we can have

$$\begin{aligned} \dot{X}_t &= \dot{X}_{t-k} \\ \dot{Y}_t &= \dot{Y}_{t-k}. \end{aligned} \tag{12}$$

Furthermore, the relations between (u_t, ν_t) and (u_{t-k}, ν_{t-k}) can be written as

$$\begin{aligned} u_t &= u_{t-k} + \dot{u}_{t-k} \\ \nu_t &= \nu_{t-k} + \dot{\nu}_{t-k} \end{aligned} \tag{13}$$

where $\dot{u}_{t-k}$ and $\dot{\nu}_{t-k}$ are the velocities of u and ν directions, respectively. By using (10)–(13), we can obtain the following equation:

$$\begin{aligned} \dot{u}_t &= \left(1 - \frac{\dot{\nu}_{t-k}}{\nu_{\text{vanish}} - \nu_{t-k}}\right)^2 \dot{u}_{t-k} \\ \dot{\nu}_t &= \left(1 - \frac{\dot{\nu}_{t-k}}{\nu_{\text{vanish}} - \nu_{t-k}}\right)^2 \dot{\nu}_{t-k} \end{aligned} \tag{14}$$

where ν_{vanish} is the ν coordinate of the vanishing line. Thus, we can obtain the velocity of a pixel in frame t based on the position and the velocity of the pixel in frame $t-k$. That is, for a tracking point (u_t, ν_t), we can estimate its new position (u_{t+k}, ν_{t+k}) by the following equations:

$$\begin{aligned} u_{t+k} &= u_t + \dot{u}_t = u_t + \left(1 - \frac{\dot{\nu}_{t-k}}{\nu_{\text{vanish}} - \nu_{t-k}}\right)^2 \dot{u}_{t-k} \\ \nu_{t+k} &= \nu_t + \dot{\nu}_t = \nu_t + \left(1 - \frac{\dot{\nu}_{t-k}}{\nu_{\text{vanish}} - \nu_{t-k}}\right)^2 \dot{\nu}_{t-k}. \end{aligned} \tag{15}$$

The details of how (14) is acquired are elaborated upon in the Appendix.

IV. Lane Recognition and Lane Coherence Verification

Boundary lines of the lane are specified using the algorithm described below. First, we find the initial angles of inclination and starting points of the lane boundaries. Then, based on the initial angles of inclination and starting points of the lane boundaries, the turning

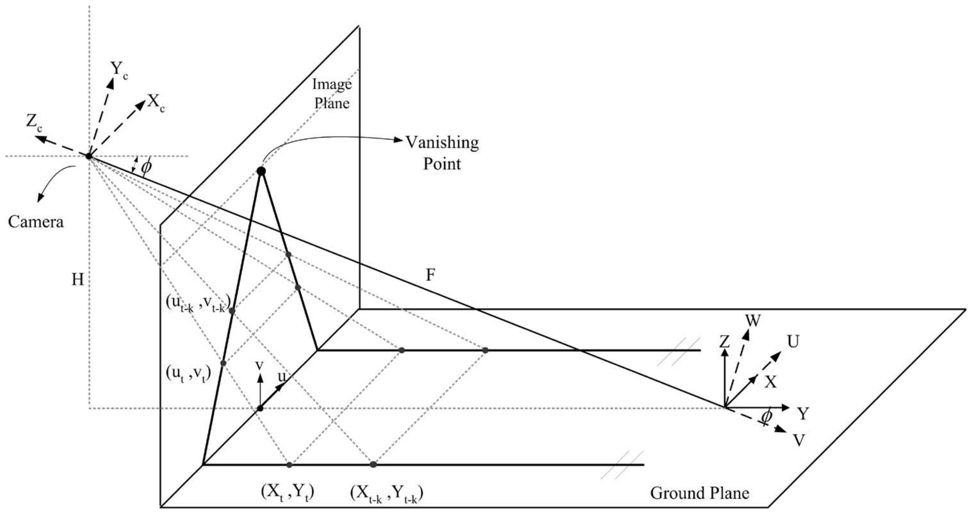

Fig. 8. Camera geometry and coordinate systems.

points on the lane boundaries are determined, and the entire lane boundaries are specified. To find out the initial angle of inclination and starting point for the left boundary, we accumulate the pixels in binary image I_{Lm} along the lines that have angle ψ_i and starting points ranging from line segments $\overline{AB}$ to $\overline{BC}$, as illustrated in Fig. 7. As for the right boundary, we accumulate the pixels in I_{Lm} along the lines that have angle $180^\circ - \psi_i$ and starting points ranging from line segments $\overline{DE}$ to $\overline{EF}$, as illustrated in Fig. 7. We define ψ_i as

$$\psi_i = \tan^{-1}\left(\frac{\nu_{\mathrm{vanish}}}{u_{\max}/2}\right) + 2^\circ \times i, \qquad i = 0, \pm 1, \pm 2, \ldots, \pm 15. \tag{16}$$

For each angle ψ_i, we find two candidates with the largest accumulated values $C^1_{\psi_i}$ and $C^2_{\psi_i}$. The initial angle of inclination and starting point of the left boundary are selected with the angle and starting point such that $\psi_i = \arg\max_{i=-15}^{15}(C^1_{\psi_i} + C^2_{\psi_i})$. The initial angle of inclination and starting point for the right boundary are selected similarly. With the starting points S_L and S_R and the initial angles ψ_L and ψ_R, we can draw a straight line L, which will intersect with the vanishing line at the ending point E_L, and a straight line R, which will intersect with the vanishing line at the ending point E_R, as illustrated in Fig. 9. From the ending points E_L and E_R, calculate along lines L and R the number of pixels that are not in the lane-mark mask I_{Lm}. If the number of accumulated pixels that are not in I_{Lm} along these two lines reaches a threshold at vertical position $\varepsilon\nu_{\mathrm{vanish}}$, then we decide that the curve of the lane is going through a major change at this point. The point where line L intersects with the horizontal line at vertical position $\varepsilon\nu_{\mathrm{vanish}}$ is denoted as d_1, as illustrated in Fig. 9(a). d_1 is defined as the first turning point of the left boundary. From the first turning point d_1, we divide the rest of the vertical space $(1-\varepsilon)\nu_{\mathrm{vanish}}$ into n segments and search for the turning point of each segment on the curved lane boundary. In other words, it is desired to determine $d_2, \ldots, d_{n+1}$. To search for turning points $d_{j+1}, j = 1, \ldots, n$ for the left boundary, start from the pixels that have the same ν coordinate as d_j and have u coordinates within the range $u(d_j) \pm \Delta s$. Then, find the maximum accumulated scan line of length $\Delta\ell_j$ with angles in the range of $\theta_{j-1} \pm \theta$ in the lane-mark mask I_{Lm}, as illustrated in Fig. 9(b). Suppose that the maximum accumulated scan line has angle θ_j; the n , the u and ν coordinates of turning point d_{j+1} can be obtained by $u(d_{j+1}) = u(d_j) + \Delta\ell_j \cos\theta_j$ and $\nu(d_{j+1}) = \nu(d_j) + \Delta\ell_j \sin\theta_j$, respectively. Note that$\theta_0 = \psi_L$. In this way, we can determine the entire left boundary by linking all the points S_L, d_1, d_2, up to d_{n+1}. The right boundary is determined in a similar way. Note that Δs is set to be four pixels, which is about half of the average width of the lane marks in the videos being processed. $\Delta\ell_j$ is dynamically determined according to (17). An advantage of this algorithm is that we can find out each angle of inclination θ_j for the curved lane boundary at different positions d_j, which is a useful information for lane keeping

$$\Delta\ell_j = \begin{cases} \frac{(1-\varepsilon)\nu_{\mathrm{vanish}}}{n-1}, & j = 1 \\ \frac{(1-\varepsilon)\nu_{\mathrm{vanish}} - \Delta\ell_{j-1}\sin\theta_{j-1}}{n-j+1} \csc\theta_{j-1}, & j = 2, \ldots, n. \end{cases} \tag{17}$$

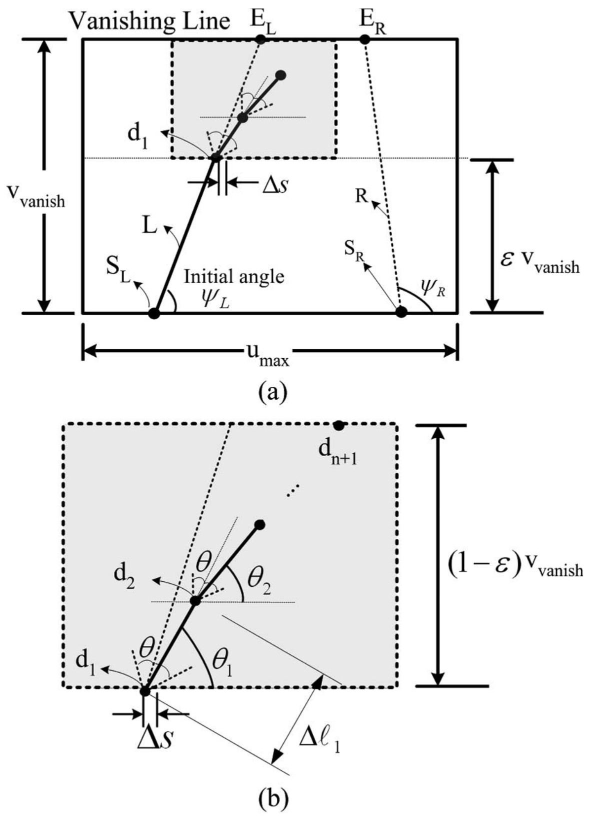

Fig. 9. Lane recognition.

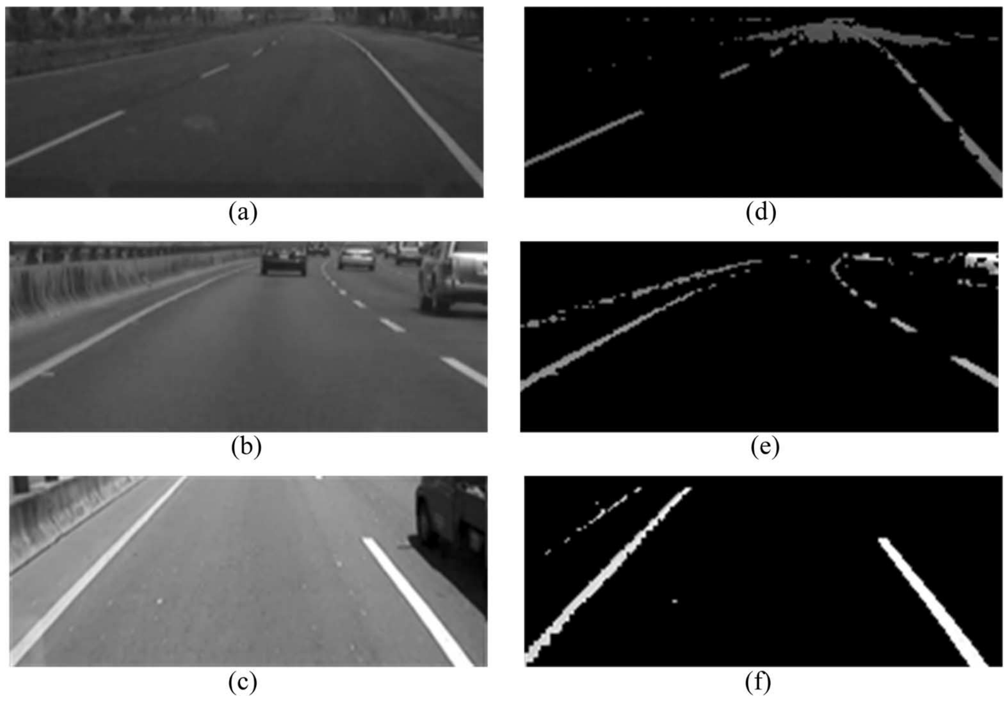

Fig. 10. Lane-mark color extraction in different weathers.

The detected lane boundaries are stored in the lane-coherence-verification module. Suppose that the left and right lane boundaries in frame t have initial angles ψ_L^t and ψ_R^t, respectively. We only need to consider initial angles within a certain range of ψ_L^t and ψ_R^t for frame $t+1$. Also, the width of the lane is computed for each image frame when the lane boundaries are detected. The newly detected lane width is compared with the lane width of the previous frame. If the difference between the two width values is larger than a threshold, the newly detected lane boundaries would be invalidated, and the previous lane boundaries would be reused. The lane-coherence-verification module can ensure the correctness of the lane-detection results when the lane marks are temporarily occluded by the passing vehicles.

When there are no lane marks visible, the lane-coherence-verification module will use the previous detected lane boundaries. If the lane boundaries have not been detected successfully for a certain amount of frames, a warning message will pop up so that the driver can be informed that there is a potential discontinuity in lane boundaries. At an intersection, if the driver wishes to turn left or right, there should be some notification to the system. Then, the system can respond according to the driver's instructions. Otherwise, we assume that the vehicle is going straightforward. At a T intersection or a dead-end street where there is no way in the front, obstacle detection using a radar sensor can easily confirm the fact that the unsuccessful detection of lane boundaries is due to such a road condition.

V. Experimental Results

Several experimental results are displayed and explained in this section. We implemented the proposed algorithm in C code incorporated with MATLAB code and conducted the experiments on a PC with 1.7-GHz CPU. The frame rate of all the experimental videos is 25 fps. Each image frame is down-sampled to size 320 × 240 pixels. Note that, when producing a 25-frames-per-second video, the digital camcorder records 50 pictures (fields) per second and intermixes every two consecutive pictures (fields) with half the height into one frame. To deinterlace, we can discard one field and preserve the other field to obtain a fully deinterlaced image frame. The size of image frames in the original video is 640 × 480 pixels. If we simply discard one field, we will get image frames of size 640 × 240. Instead of simply discarding one field, we down-sample the image frames. By down-sampling each image frame to 320 × 240 pixels, we also discard one field and preserve the other field and obtain a fully deinterlaced image frame.

The first experiment is designed to test the robustness of our lane-mark color extraction module. Fig. 10(a)–(c) is road images taken in different weather. The weather is cloudy in Fig. 10(a) and (b), and the colors of both road surface and lane-mark areas are darker compared to those in Fig. 10(c). Fig. 10(d)–(f) is the results of the lane-mark color extraction for Fig. 10(a)–(c), respectively. We can observe that the lane-mark colors are successfully extracted, regardless of the weather.

Fig. 11 illustrates an example of the proposed lane-detection process. Fig. 11(a) is the original image frame I_{Ori}. Fig. 11(b) shows the result of the road-color extraction module I_{Rd}. Fig. 11(c) shows the binary mask of the extracted lane-mark color area I_{Lmc}. Fig. 11(d) shows the binary mask of the actual lane-mark area I_{Lm} after moving vehicle elimination procedure. Fig. 11(e) is the final lane-detection result. Note that the detected results are superimposed onto the original image frames.

Fig. 12 shows some lane-detection results selected from different image sequences. Fig. 12(a)–(d) is taken in clear weather. The left lane marks are double solid lines, with rows of oblique lines between them in Fig. 12(a), a solid line in Fig. 12(b) and (c), and double solid lines in Fig. 12(d). The right lane marks are dotted lines in Fig. 12(a)–(c) and a solid line in Fig. 12(d). We can see from the detection results that the boundary lines of the lanes are all correctly detected. Fig. 12(e)–(h) is captured in cloudy weather. Note that the camera position for Fig. 12(e)–(h) is different from that for Fig. 12(a)–(d). Therefore, the vanishing lines are at the same position in Fig. 12(a)–(d) but are different from the position of the vanishing lines in Fig. 12(e)–(h). The road under test is dotted and dotted in Fig. 12(e) and (f), dotted and solid in Fig. 12(g), and solid and dotted in Fig. 12(h). Again, the displayed results show that the proposed algorithm also works well in cloudy weather.

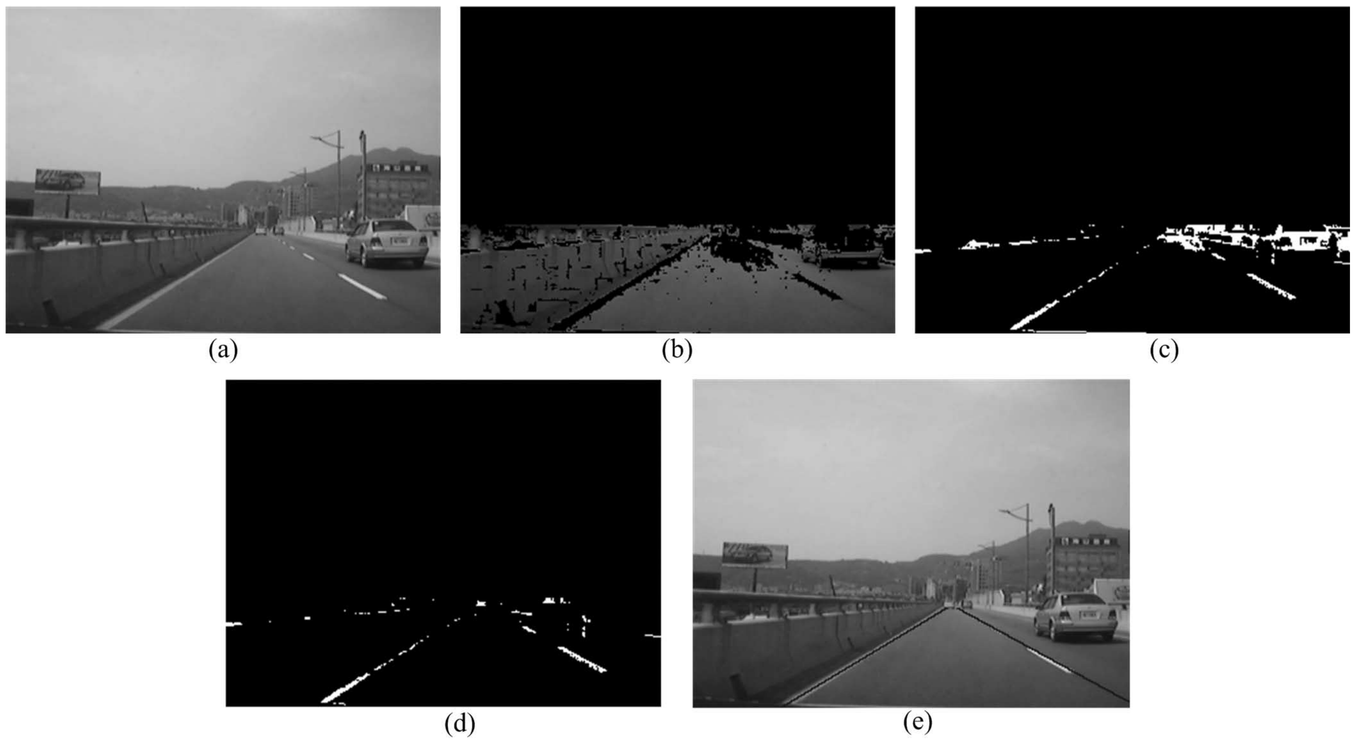

Fig. 11. Example of the proposed lane-detection process. (a)I_{Ori}. (b) I_{Rd}. (c) I_{Lmc}. (d) I_{Lm}. (e) Final lane-detection result.

Figs. 13 and 14 exhibit lane-detection results on curved roads and night scenes, respectively. Since it is more difficult to see the original lane marks after the detection results are superimposed onto the original image frames in night scenes, we display the original image frames in Fig. 14(a)–(c). If the lighting condition along the road is reasonably good and we include the road and lane-mark colors at night when performing color analysis, then the algorithm can also work under night conditions. However, for night scenes that are too dark, the proposed algorithm is not applicable. The computational time for each image frame depends on the number of objects to be tracked in that frame. The average computational times for one image frame in the sequences shown in Figs. 12–14 are 0.3981, 0.2975, and 0.3472 s, respectively. Note that the lane-mark color extraction module and the moving vehicle elimination module are implemented using MATLAB, which are the most time-consuming parts in this system. According to [14], the boundary module and area module in their method takes 0.7310 and 0.7110 s per image frame, respectively, on a 1.5-GHz PC, with the algorithm implemented in MATLAB and image size 192 $\times$ 144 pixels. Therefore, the computational time for our algorithm is satisfactory.

In multilane highways, the algorithm gives the lane boundaries of the lane on which the vehicle is driving. There will not be a false result due to multilane, as shown in the experimental results. For lane change maneuver in automated-highway-systems applications, there are two schemes: One is infrastructure-guided lane changes; the other is lane changes with dead reckoning [21]. If there is some infrastructure enhancement or installation of extra magnets to mark a path between adjacent lanes, the lateral position information can be made continuously available to the intelligent vehicles. Otherwise, due to the nature of discontinuous availability of valid preview data that sensing systems (either vision or radar systems) can acquire during lane-to-lane transitions, intelligent vehicles will inevitably encounter a sensory "dead-zone" period during the transition period. Hatipoglu *et al.* [21] designed a virtual yaw reference trajectory during lane-to-lane transition and converted the open-loop lane change problem into an equivalent virtual reference trajectory tracking problem. For lane following and lane departure warning applications, the system will warn the driver once the intelligent vehicle is deviating away from the lane centerline. If the driver wishes to change from one lane to another, he should notify the system (for example, flash the left or right signal) before steering the wheel so that it will not be treated by the system as a dangerous condition. Then, according to the notification, the system can redetect the lane boundaries of the new lane when the transition is completed.

This brief presents a method for robust road lane detection for highways and major city roads with lane markings. The main contributions of this brief include 1) extracting lane-mark colors in a way that is not affected by illumination changes and the proportion of space that vehicles on the road occupy; 2) deriving equations that utilize only the vertical position of the vanishing line, the previous position of a point, and the velocity of the point to estimate the new position of the same point on the ground plane in an image sequence taken from a driving vehicle; and 3) designing a mechanism to effectively eliminate the influence of passing vehicles when performing lane detection. The comparisons between this brief and other papers in the literature are made as follows.

1) Compared with edge-based methods, the proposed algorithm will not be disturbed by unnecessary edges, whereas the strong edges in the middle of the road, as shown in Figs. 12(d) and 13(c), will be mistaken as possible road boundaries in [5].
2) Compared with the intensity-based method described in [7], a car with bright colors, as illustrated in Fig. 12(d), will make the intensity feature $I(C_i)$ very strong on the car body in the regions of interest and thus bias the distance metric in [7]. However, our algorithm can eliminate those vehicles in the moving vehicle elimination module and still work well in this situation.
3) Compared with the color-based method described in [14], the percentage of road surface that is occupied by objects will not

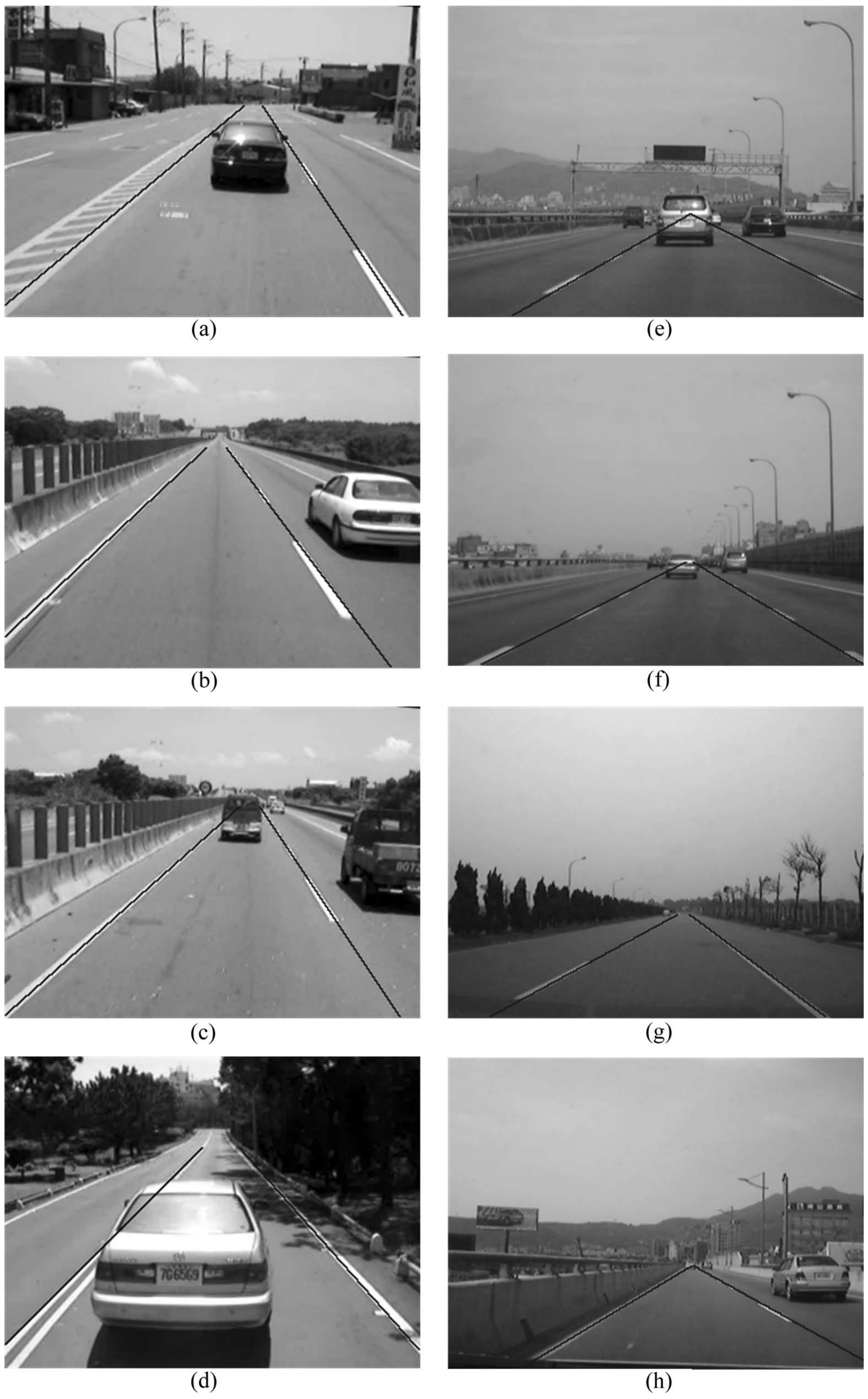

Fig. 12. Lane-detection results on straight roads.

affect our detection result, whereas the road-color model in [14] will be influenced by vehicles occupying a certain amount of space between the two chosen boundary candidates, such as the examples in Fig. 12(a)–(d).

Also, note that different camera positions in the experimental results illustrated in Figs. 12–14 do not require significant system recalibration since the only parameter required by the system that depends on camera positions is the vertical position of the vanishing line, which

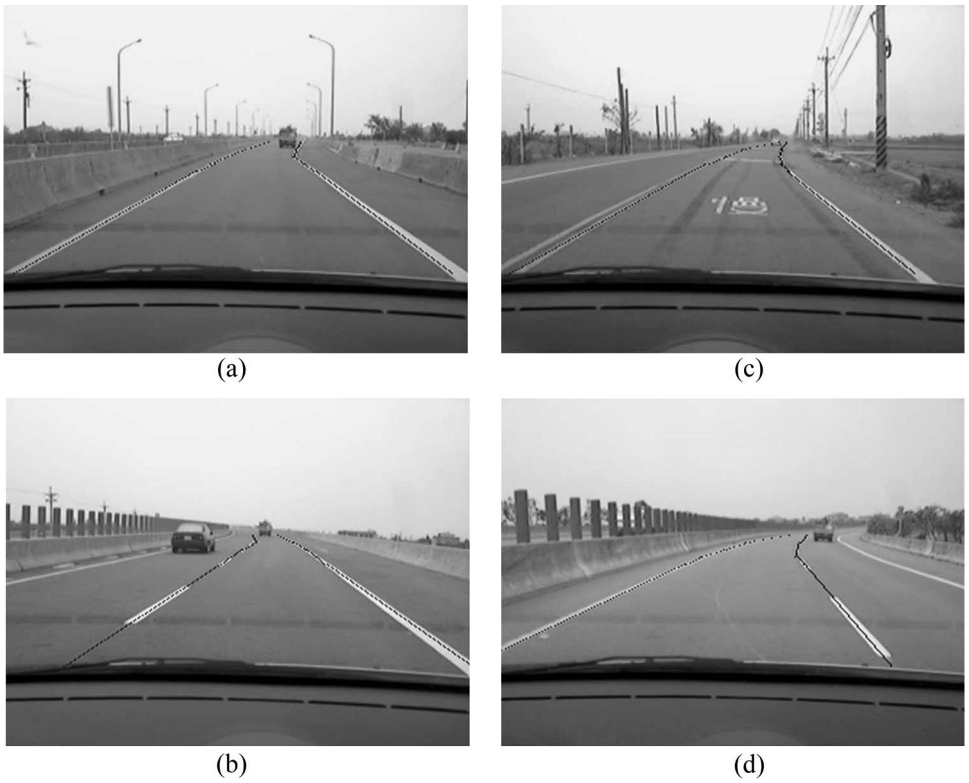

Fig. 13. Lane-detection results on curved roads.

can be computed offline. To determine the vertical position of the vanishing line, we only need the focal length f and the tilting angle ϕ because $\nu_{\text{vanish}} = f \tan \phi$ (see Appendix). Since the parameters for new position estimation are simplified, no complicated system recalibration needs to be involved. When the same camera is moved from a position, as illustrated in Fig. 12(a)–(d), to a position, as illustrated in Fig. 12(e)–(h), the only camera parameter changed is the angle ϕ.

VI. Conclusion

The lane-detection method proposed in this brief can robustly find the left and right boundary lines of the lane and is not affected by the passing traffic. By extracting lane marks based on color information, we can eliminate many articles on the road that might interfere with lane-detection process. For vehicles that have the same colors as the lane marks, we utilize size, shape, and motion information to distinguish them from the real lane marks. In the final process, we specify the boundary lines of the lane by accumulating pixels in the lane-mark mask. The initial angles of inclination and starting points of the lane boundaries are determined first. Scanning process is performed afterward in search of the turning points for the entire curved lane boundaries. The main contributions of this brief include 1) extracting lane-mark colors in a way that is not affected by illumination changes and the proportion of space that vehicles on the road occupy; 2) deriving equations that utilize only the vertical position of the vanishing line, the previous position of a point, and the velocity of the point to estimate the new position of the same point on the ground plane in an image sequence taken from a driving vehicle; and 3) designing a mechanism to effectively eliminate the influence of passing vehicles when performing lane detection. In conclusion, the proposed algorithm works well on marked roads under various kinds of lighting conditions. The experimental results confirm the effectiveness of the proposed method.

Appendix

From $\dot{Y}_t = \dot{Y}_{t-k}$ in (12) and $\dot{Y}_t = [(\dot{\nu}_t F f \sin\phi)/(f \sin\phi - \nu_t \cos\phi)^2]$ in (11), we have

$$\frac{\dot{\nu}_t F f \sin\phi}{(f\sin\phi - \nu_t \cos\phi)^2} = \frac{\dot{\nu}_{t-k} F f \sin\phi}{(f\sin\phi - \nu_{t-k}\cos\phi)^2}.$$

By substituting ν_t with $(\nu_{t-k} + \dot{\nu}_{t-k})$, we can get

$$\begin{aligned}\dot{\nu}_t &= \frac{[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2}{(f\sin\phi - \nu_{t-k}\cos\phi)^2}\dot{\nu}_{t-k} \\ &= \frac{(f\tan\phi - \nu_{t-k} - \dot{\nu}_{t-k})^2}{(f\tan\phi - \nu_{t-k})^2}\dot{\nu}_{t-k} \\ &= \left(1 - \frac{\dot{\nu}_{t-k}}{f\tan\phi - \nu_{t-k}}\right)^2 \dot{\nu}_{t-k}.\end{aligned}$$

Furthermore, we know that

$$\begin{aligned}\nu_{\text{vanish}} &= \lim_{Y\to\infty} \frac{fY\sin\phi}{Y\cos\phi + F} \\ &= \lim_{Y\to\infty} \frac{f\sin\phi}{\cos\phi + F/Y} = f\tan\phi.\end{aligned}$$

Therefore, we have $\dot{\nu}_t = \{1 - [\dot{\nu}_{t-k}/(\nu_{\text{vanish}} - \nu_{t-k})]\}^2 \dot{\nu}_{t-k}$. Similarly, from $\dot{X}_t = \dot{X}_{t-k}$ in (12) and

$$\dot{X}_t = \frac{\dot{u}_t F f \sin^2\phi - \dot{u}_t \nu_t \sin\phi\cos\phi F + u_t \dot{\nu}_t \sin\phi\cos\phi F}{(f\sin\phi - \nu_t\cos\phi)^2}$$

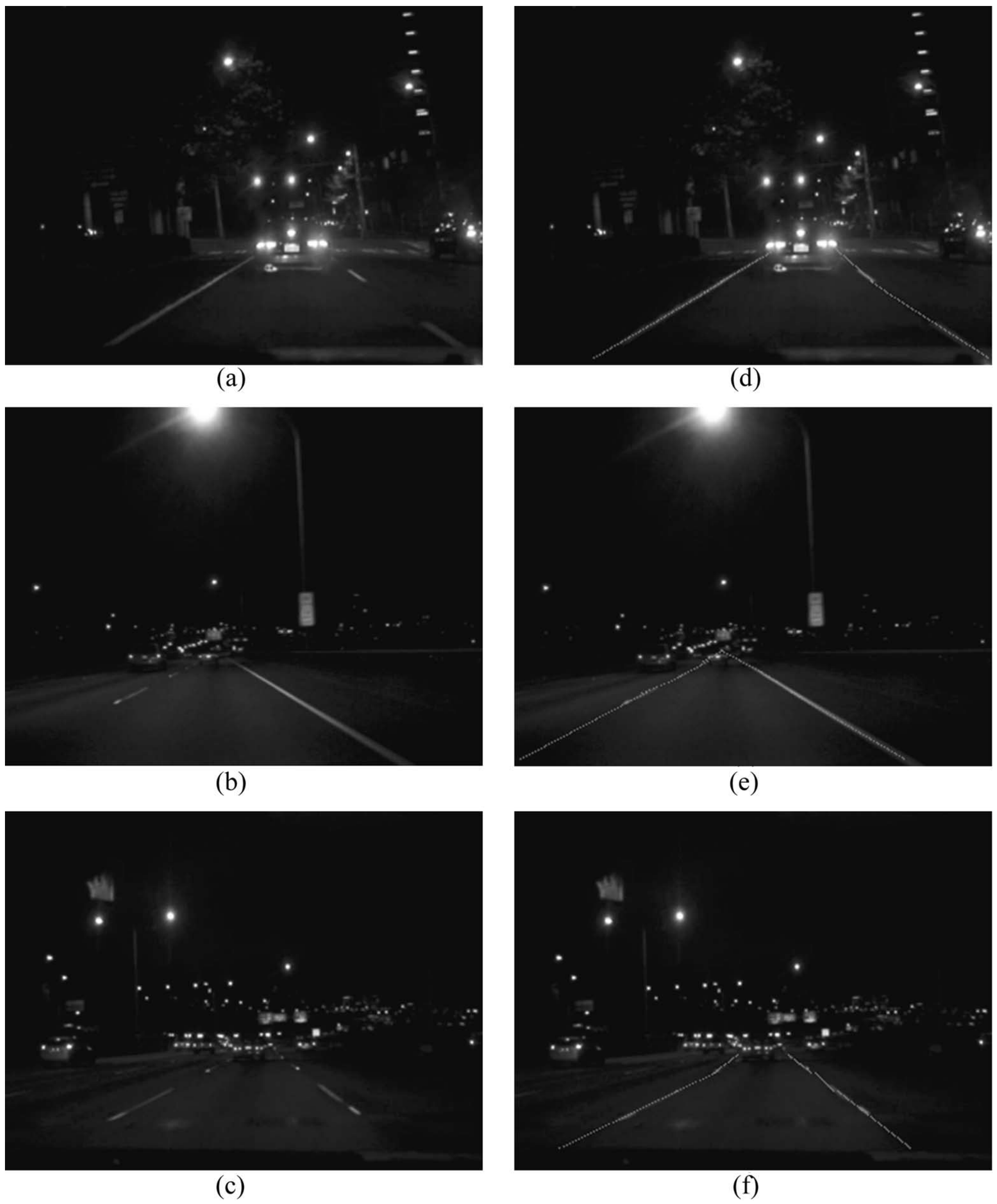

Fig. 14. Lane-detection results under night conditions. (a)–(c) Original image frames. (d)–(f) Detected results.

in (10), we have the expression

$$\frac{\dot{u}_t F f \sin^2\phi - \dot{u}_t \nu_t \sin\phi\cos\phi F + u_t \dot{\nu}_t \sin\phi\cos\phi F}{(f\sin\phi - \nu_t\cos\phi)^2}$$
$$= \frac{\dot{u}_{t-k} F f \sin^2\phi - \dot{u}_{t-k}\nu_{t-k}\sin\phi\cos\phi F + u_{t-k}\dot{\nu}_{t-k}\sin\phi\cos\phi F}{(f\sin\phi - \nu_{t-k}\cos\phi)^2}.$$

Eliminating $F\sin\phi$ on both sides and substituting ν_t and u_t with $(\nu_{t-k} + \dot{\nu}_{t-k})$ and $(u_{t-k} + \dot{u}_{t-k})$, respectively, we get

$$\frac{\dot{u}_t f\sin\phi - \dot{u}_t(\nu_{t-1} + \dot{\nu}_{t-k})\cos\phi + (u_t + \dot{u}_{t-k})\dot{\nu}_t\cos\phi}{[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2}$$
$$= \frac{\dot{u}_{t-k} f\sin\phi - \dot{u}_{t-k}\nu_{t-k}\cos\phi + u_{t-k}\dot{\nu}_{t-k}\cos\phi}{(f\sin\phi - \nu_{t-k}\cos\phi)^2}.$$

Replacing $\dot{\nu}_t$ by $\{[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2/(f\sin\phi - \nu_{t-k}\cos\phi)^2\}\dot{\nu}_{t-k}$ and rearranging the above equation, it becomes

$$\begin{aligned}&\dot{u}_t\,[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]\,(f\sin\phi - \nu_{t-k}\cos\phi)^2\\ &\quad + (u_t + \dot{u}_{t-k})\,[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2\,\dot{\nu}_{t-k}\cos\phi\\ &\quad = [f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2\\ &\qquad \times (\dot{u}_{t-k} f\sin\phi - \dot{u}_{t-k}\nu_{t-k}\cos\phi + u_{t-k}\dot{\nu}_{t-k}\cos\phi).\end{aligned}$$

Rearranging the above equation again, we have

$$\begin{aligned}&\dot{u}_t\,[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]\,(f\sin\phi - \nu_{t-k}\cos\phi)^2\\ &\quad = [f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2\\ &\qquad \times [\dot{u}_{t-k}(f\sin\phi - \nu_{t-k}\cos\phi - \dot{\nu}_{t-k}\cos\phi)\\ &\qquad\quad + u_{t-k}\dot{\nu}_{t-k}\cos\phi - u_t\dot{\nu}_{t-k}\cos\phi]\,.\end{aligned}$$

Finally, the equation for $\dot{u}_t$ can be obtained as follows:

$$\begin{aligned}\dot{u}_t &= \frac{[f\sin\phi - (\nu_{t-k} + \dot{\nu}_{t-k})\cos\phi]^2}{(f\sin\phi - \nu_{t-k}\cos\phi)^2}\dot{u}_{t-k} \\ &= \frac{[f\tan\phi - (\nu_{t-k} + \dot{\nu}_{t-k})]^2}{(f\tan\phi - \nu_{t-k})^2}\dot{u}_{t-k} \\ &= \left(1 - \frac{\dot{\nu}_{t-k}}{f\tan\phi - \nu_{t-k}}\right)^2 \dot{u}_{t-k} \\ &= \left(1 - \frac{\dot{\nu}_{t-k}}{\nu_{\text{vanish}} - \nu_{t-k}}\right)^2 \dot{u}_{t-k}.\end{aligned}$$

References

[1] M. Bertozzi *et al.*, “Artificial vision in road vehicles,” *Proc. IEEE*, vol. 90, no. 7, pp. 1258–1271, Jul. 2002.
[2] M. Bertozzi, A. Broggi, A. Fascioli, T. Graf, and M.-M. Meinecke, “Pedestrian detection for driver assistance using multiresolution infrared vision,” *IEEE Trans. Veh. Technol.*, vol. 53, no. 6, pp. 1666–1678, Nov. 2004.
[3] C. Little, “The intelligent vehicle initiative: Advancing ‘human-centered’ smart vehicles,” *Public Roads Mag.*, vol. 61, no. 2, pp. 18–25, Sep./Oct. 1997.
[4] K. Kluge and S. Lakshmanan, “A deformable-template approach to lane detection,” in *Proc. IEEE Intell. Vehicle Symp.*, Sep. 1995, pp. 54–59.
[5] Y. Wang, E. K. Teoh, and D. Shen, “Lane detection using B-snake,” in *Proc. Int. Conf. Inf. Intell. and Syst.*, 1999, pp. 438–443.
[6] Y. Wang, D. Shen, and E. K. Teoh, “Lane detection using spline model,” *Pattern Recognit. Lett.*, vol. 21, no. 8, pp. 677–689, Jul. 2000.
[7] Y. U. Yim and S. Y. Oh, “Three-feature based automatic lane detection algorithm (TFALDA) for autonomous driving,” *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 4, pp. 219–225, Dec. 2003.
[8] F. W. J. Gibbs and B. T. Thomas, “The fusion of multiple image analysis algorithms for robot road following,” in *Proc. IEEE 5th Int. Conf. Image Process. and Appl.*, Jul. 1995, pp. 394–398.
[9] C. Rasmussen, “Grouping dominant orientations for ill-structured road following,” in *Proc. IEEE Comput. Soc. Conf., Comput. Vis. and Pattern Recognit.*, Jul. 2004, vol. 1, pp. 470–477.
[10] K. Kluge, “Extracting road curvature and orientation from image edge points without perceptual grouping into features,” in *Proc. IEEE Intell. Vehicle Symp.*, Oct. 1994, pp. 109–114.
[11] K. Kluge and G. Johnson, “Statistical characterization of the visual characteristics of painted lane marking,” in *Proc. IEEE Intell. Vehicle Symp.*, Sep. 1995, pp. 488–493.
[12] J. P. Gonzalez and U. Ozguner, “Lane detection using histogram-based segmentation and decision trees,” in *Proc. IEEE Intell. Transp. Syst.*, Oct. 2000, pp. 346–351.
[13] K. A. Redmill, S. Upadhya, A. Krishnamurthy, and Ü. Özgüner, “A lane tracking system for intelligent vehicle application,” in *Proc. IEEE Intell. Transp. Syst.*, Aug. 2001, pp. 273–279.
[14] Y. He, H. Wang, and B. Zhang, “Color-based road detection in urban traffic scenes,” *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 309–318, Dec. 2004.
[15] K. Onoguchi, N. Takeda, and M. Watanabe, “Planar projection stereopsis method for road extraction,” *IEICE Trans. Inf. Syst.*, vol. E81-D, no. 9, pp. 1006–1018, 1998.
[16] A. Takahashi, Y. Ninomiya, M. Ohta, M. Nishida, and M. Takayama, “Rear view lane detection by wide angle camera,” in *Proc. IEEE Intell. Vehicle Symp.*, Jun. 2002, vol. 1, pp. 148–153.
[17] S. G. Jeong *et al.*, “Real time lane detection for autonomous navigation,” in *Proc. IEEE Intell. Transp. Syst.*, Aug. 2001, pp. 508–513.
[18] J. S. Jin, Z. Zhu, and G. Xu, “A stable vision system for moving vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 1, pp. 32–39, Mar. 2000.
[19] T. N. Schoepflin and D. J. Dailey, “Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation,” *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 90–98, Jun. 2003.
[20] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis and Machine Vision*, 2nd ed. Glenrothes, U.K.: Thomson-Engineering, 1998.
[21] C. Hatipoglu, U. Ozguner, and K. A. Redmill, “Automated lane change controller design,” *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 1, pp. 13–22, Mar. 2003.

Application of Neural Networks to Disturbances Encountered Landing Control

Jih-Gau Juang, *Member, IEEE*, and Kai-Chung Cheng

***Abstract*—Neural network applications to aircraft automatic landing control are presented. Conventional automatic landing systems (ALSs) can provide a smooth landing, which is essential to the comfort of passengers. However, these systems work only within a specified operational safety envelope. When the conditions are beyond the envelope, such as turbulence or wind shear, they often cannot be used. The objective of this paper is to investigate the use of neural networks in ALSs and to make these systems more intelligent. Current flight control law is adopted in the intelligent controller design. Tracking performance and robustness are demonstrated through software simulations. This paper presents five different neural network controllers to improve the performance of conventional ALSs based on a modified learning-through-time process. Simulation results show that the neural network controllers can successfully expand the safety envelope to include more hostile environments such as severe turbulence.**

***Index Terms*—Automatic landing system (ALS), flight control, neural networks, turbulence.**

I. Introduction

On August 22, 1999, China Airlines Flight 642 had a hard landing at Hong Kong International Airport. The lifting wing of the airplane was broken upon landing. The accident killed three passengers and injured 211 people. After a 15-month investigation, a crash report was released on November 30, 2000. It showed that the accident was mainly due to the improper crosswind correction by Software 907 on the Boeing MD-11. Boeing also confirmed this software problem later and replaced nearly 190 MD-11 crosswind-correction software with the 908 software. Although this accident was attributed to software problem, the automatic landing system (ALS) has been the focus of the safety issue. The first ALS was made in England in 1965. Since then, most aircraft have been installed with this system. The ALS relies on the Instrument Landing System to guide the aircraft into the proper altitude, position, and approach angle during the landing phase.

Most conventional control laws generated by the ALS are based on the gain scheduling method [1]. Control parameters are preset for different flight conditions within a specified safety envelope, which is relatively defined by Federal Aviation Administration (FAA) regulations. According to FAA regulations, environmental conditions considered in the determination of dispersion limits are: headwinds up to 25 knots; tailwinds up to 10 knots; crosswinds up to 15 knots; moderate turbulence, wind shear of 8 knots per 100 ft from 200 ft to touchdown [2]. If the flight conditions are beyond the preset envelope, the ALS is disabled and the pilot takes over. An inexperienced pilot may not be able to guide the aircraft to a safe landing. According to National Transportation Safety Board (NTSB) report [3], between 1989 and 1999, 22.6% of the flight safety events are due to weather factors. Among them, 62% are attributed to wind disturbances. It is therefore

Manuscript received March 7, 2004; revised August 29, 2004, November 4, 2005, May 14, 2006, September 4, 2006, and September 14, 2006. This work was supported in part by the National Science Council, Taiwan, under Grant NSC92-2213-E-019-005. The Associate Editor for this paper was S. Tang.

J.-G. Juang is with the Communications and Guidance Engineering Department, National Taiwan Ocean University, Keelung 20224, Taiwan, R.O.C. (e-mail: jgjuang@mail.ntou.edu.tw).

K.-C. Cheng is with the LITE-ON Technology Corporation, Taipei 114, Taiwan, R.O.C. (e-mail: kennyj.cheng@liteon.com).

Color versions of Figs. 4 and 9–11 are available at http://ieeexplore.ieee.org.
Digital Object Identifier 10.1109/TITS.2006.884885