Proc. Natl. Sci. Counc. ROC(A)
Vol. 22, No. 5, 1998. pp. 691-702

Vision-based Guidance for Autonomous Land Vehicle Navigation in Outdoor Road Environments with Static and Moving Cars

KUANG-HSIUNG CHEN AND WEN-HSIANG TSAI[†]

Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

(Received January 16, 1998; Accepted March 2, 1998)

ABSTRACT

A new effective approach to vision-based guidance for autonomous land vehicle (ALV) navigation in outdoor road environments with static and moving cars using model matching and color information clustering techniques is proposed. The conventional way of detecting obstacles and cars in the navigation route, which is in general difficult, is avoided; instead, collision-free road area detection, which is usually easier, is adopted. Road boundaries are used to construct the reference model, and road surface intensity is selected as the visual feature. The pixels in a road image near the two lines representing the road boundary shape, which are estimated at the beginning of each navigation cycle, are checked to judge whether the left or right lane width has changed due to occlusion caused by nearby static or moving cars on the road. If both lane widths have not changed, model matching is performed immediately to find the ALV location. If either or both lane widths have changed, corresponding processes are performed to find the width of the occluded road, and a model is recreated if the new road width is different from the old one in the previous navigation cycle. Model matching is then performed to locate the ALV on the occluded road. To save computing time, only partial model matching is performed. A turn angle is then computed to guide the ALV to follow the central path line on the extracted road for safe navigation. Various color information on roads is used to extract road surfaces, and a clustering algorithm is used to solve the problem caused by great changes of intensity in navigation environments. Successful navigation tests show that the proposed approach is effective for ALV guidance on common roads with static and moving cars.

Key Words: autonomous land vehicle, vision-based navigation, guidance, cars, model matching, color information clustering, image processing, computer vision

I. Introduction

The application of computer vision to the guidance of autonomous land vehicles (ALV's) in outdoor road environments is a challenging task because of the great variety of road conditions, such as changes of illumination, the existence of static or moving obstacles, and shadows, degraded surfaces, rain etc. Successful ALV navigation requires integration of the techniques of environment sensing, image processing, ALV location, path planning, wheel control, and so on. This study is mainly concerned with ALV guidance using computer vision techniques in outdoor road environments with static and moving cars.

Many research works have been reported for obstacle detection on outdoor roads. Most systems, such as the Carnegie-Mellon University Navlab (Goto and Stentz, 1987; Thorpe, 1991; Thorpe *et al.*, 1988, 1991a, 1991b; Singh and Keller, 1991; Crisman and Thorpe, 1991, 1993), the vehicle constructed by Martin Marietta Denver Aerospace (Waxman *et al.*, 1987; Turk *et al.*, 1988; Olin and Tseng, 1991), and the navigation system developed at the University of Maryland (Davis, 1991), use laser range sensors to detect obstacles in outdoor roads or cross-country terrain. The system developed by Kuan *et al.* (1988) uses a sonic imaging sensor and an infrared sensor for obstacle avoidance and target detection.

As for vision-based approaches to obstacle detection, they basically can be divided into three classes. The first class extracts obstacles directly from 2-D images. Only one camera is used, and only the image

[†]To whom all correspondence should be addressed.

in the current navigation cycle with *a priori* knowledge and predefined assumptions is considered. The second class of approaches uses motion information obtained from a sequence of images to detect obstacles. The most popular approaches in this class are based on optical flow. The third class of approaches detects obstacles using stereo-vision techniques. Besides, Xie *et al.* (1994) use a range finder coupled with a camera to acquire 3D information about obstacles. Although the second and third classes have fewer assumptions than the first, usually more errors are produced and much computing time is required because image correspondence is involved. On the other hand, the first class in general takes less computing time and has better detection results, though it has more assumptions. Hence, the first class of approaches is widely adopted for real-time navigation.

The major obstacles on outdoor roads are static and moving cars. Several approaches have been proposed for detecting cars. Schwarzinger *et al.* (1992), Thomanek *et al.* (1994), and Regensburger and Graefe (1994) used some symmetry features at the rear of a car to detect and track cars in front. Cappello *et al.* (1994) proposed a car detection method based on the assumptions that cars are approximately solid blocks with parallel or orthogonal sides and that the sides are parallel or orthogonal to road boundaries. Efenberger *et al.* (1992) developed a vision system to recognize vehicles approaching from behind. Schmid (1994) created a generic 3-D shape model of different cars, including passenger cars, trucks, and buses. Model matching is performed to detect cars on the road ahead. Heisele and Ritter (1995) and Smith and Brady (1994) detected cars based on optical flow. Kehtarnavaz *et al.* (1991) developed a visual control system for vehicle following. It tracks an apparent feature, a single point, on the back of the leading vehicle using stereo-vision techniques. Additionally, Brauckmann *et al.* (1994) developed an integrated visual sensing system for car detection with three main tasks, namely, long-distance detection of cars in front, short-distance detection of sudden-intruding cars or other obstacles, and lateral monitoring of overtaking and overtaken cars in neighboring lanes. All of the above car detection approaches identify features on the body of the car directly when nearby cars appear on the road.

In this paper, we propose an effective approach to guiding an ALV on general roads with static and moving cars. The approach allows variations of the road width, which are caused by the existence of static cars on the roadside or moving cars in the road lane. New algorithms are proposed for finding the best-fit road region without having to do exhaustive search. We do not detect cars in the image; instead, we detect road areas using the road surface intensity as a visual feature. The road width is not assumed to be constant; instead, road width changes are allowed and computed as cars appear in the image. Furthermore, we do not plan a complex navigation route but just compute the central line on the extracted road as the route and approach it immediately after the ALV location on the extracted road is found. With neither additional feature extraction from the car body nor additional planning for the navigation path, fast navigation can be achieved.

In this study, the shape of a road is represented by its two road boundaries. To find the road boundaries on a road, edge points are first extracted directly from the road image. The edge points are then used to figure out the road boundaries. When complex obstacles appear on the road, the edges of the road become irregular and cause difficulty in road boundary estimation using the edge points. This motivates in this study the design of a model matching method which uses several candidate boundary lines in the model to match the road pixels nearing the estimated road boundaries in the image according to a reasonable similarity measure.

More specifically, before model matching, we first check in the image the pixels near two reference lines representing the road boundary shape, which are estimated at the beginning of each cycle, to judge whether the left or right lane width has changed due to nearby static or moving cars or other obstacles on the road. If both lane widths are unchanged, model matching is performed immediately to find the ALV location. If either or both lane widths have changed, corresponding processes are performed to find the new width of the occluded road, and the model is recreated in real time if the new road width is different from the old road width in the previous cycle. Model matching is then performed to locate the ALV on the occluded road. To save computing time, only partial model matching is performed. A turn angle is then computed to guide the ALV to follow the navigation path, which is assumed to be the central line on the extracted road in this study. In addition, to achieve steady and flexible navigation, the speed of the ALV is adjusted dynamically if the road width is changed or when the ALV moves on ascending or descending roads.

Furthermore, various color information, which often simplifies object extraction and identification from a scene, is used in this approach to extract road surfaces. An ISODATA algorithm (Duda and Hart, 1973) based on an initial-center-choosing (ICC) technique has been designed to solve the problem caused by great changes of intensity in navigation. This clustering algorithm has been proved effective by nearly one hundred successful tests on real roads with different cars in different weather conditions. The above procedures of road width checking and estimation, model matching, immediate navigation path generation, and dynamic ALV speed adjustment, combined with the color information clustering process, enable the ALV to navigate safely and smoothly on common roads, as demonstrated by our experiments.

Some assumptions under which our approach works are listed in the following.

(1) The road boundaries are straight and parallel to each other, but the road width can vary in the navigation process.

(2) The system does not detect ascending or descending roads during navigation. But when the ALV moves on ascending or descending roads, the speed of the ALV can be adjusted appropriately for steady navigation.

The remainder of this paper is organized as follows. In Section II, the details of the proposed model-based ALV guidance method are described. In Section III, the employed navigation guidance method and the adopted vehicle control schemes are introduced in detail. A description of the employed image processing and feature extraction techniques is included in Section IV. In Section V, the results of some experiments are described. Finally, some conclusions are stated in Section VI.

II. Proposed Model-based ALV Guidance Method

The proposed guidance scheme is performed in a cycle-by-cycle manner. In each navigation cycle, the system identifies the visual feature of the road surface in the road environment to locate the ALV and guides the ALV accordingly from the current position to the navigation path, which is assumed to be the central line of the unoccluded road portion. After the two lines representing the road boundary shape in the image are estimated at the beginning of the cycle, the system checks the pixels near the two lines to judge whether the left or right lane width is changed in the current cycle. Model matching is then performed to find the ALV location if both lane widths are unchanged. If either or both lane widths have changed, appropriate procedures are executed to find the new road width, and the model is recreated if the new road width is different from the old road width computed in the previous cycle. Then, model matching is performed to find the ALV location on the changed road.

In the following, we will state first the steps involved in model creation, which involves several coordinate systems and transformations, and then we will describe the processes for detection of lane width changes as well as estimation of the new road width, followed by the model matching process.

1. Model Creation

Several coordinate systems and coordinate transformations are used in this approach. The image coordinate system (ICS), denoted as *u-w*, is attached to the image plane of the camera mounted on the ALV. The camera coordinate system (CCS), denoted as *u-v-w*, is attached to the camera lens center. The vehicle coordinate system (VCS), denoted as *x-y-z*, is attached to the middle point of the line segment which connects the two contact points of the two front wheels of the ALV with the ground. The *x*-axis and the *y*-axis are on the ground and parallel to the short and the long sides of the vehicle body, respectively. The *z*-axis is vertical to the ground. The transformation between the CCS and the VCS can be written in terms of homogeneous coordinates (Wang *et al.*, 1993; Su and Tsai, 1994) as

$$(u\,v\,w\,1) = (x\,y\,z\,1)\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_d & -y_d & -z_d & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where

$$\begin{aligned}
&r_{11}=\cos\theta\cos\varphi+\sin\theta\sin\phi\sin\varphi,\\
&r_{12}=-\sin\theta\ \cos\phi,\\
&r_{13}=\sin\theta\ \sin\phi\ \cos\varphi-\cos\theta\ \sin\varphi,\\
&r_{21}=\sin\theta\ \cos\varphi-\cos\theta\ \sin\phi\ \sin\varphi,\\
&r_{22}=\cos\theta\ \cos\phi,\\
&r_{23}=-\cos\theta\ \sin\phi\ \cos\varphi-\sin\theta\ \sin\varphi,\\
&r_{31}=\cos\phi\ \sin\varphi,\\
&r_{32}=\sin\phi,\\
&r_{33}=\cos\phi\ \cos\varphi,
\end{aligned} \tag{2}$$

where θ is the pan angle, ϕ is the tilt angle, and φ is the swing angle of the camera with respect to the VCS; and (x_d,y_d,z_d) is the translation vector from the origin

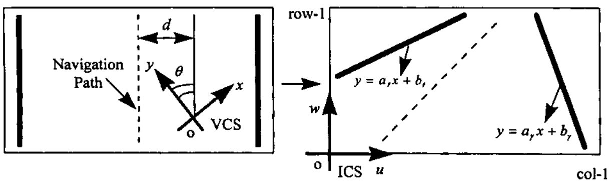

Fig. 1. The road boundary transformation between the VCS and the ICS.

of the CCS to the origin of the VCS.

An ALV location can be represented by two parameters (d,θ), where d is the distance from the ALV to the central path line in the road and θ is the pan angle of the ALV relative to the road direction. The equations of the two road boundaries in the VCS are assumed to be known before navigation. We transform the road boundaries into the ICS, which are displayed on the TV monitor during each navigation session.

For each ALV location (d_i,θ_j), we create a corresponding template $T_{ij}[a_\ell,b_\ell,a_\gamma,b_\gamma]$, where a_ℓ and a_γ are the slopes and b_ℓ and b_γ are the intercepts, of the equations of the left and the right road boundaries in the ICS, respectively. The transformation is shown in Fig. 1. We sample the road width at 41 positions, i.e., $-20 \leq i \leq 20$, with an interval of 1/40 of the road width. At each position, we sample the vehicle direction at 31 angles from −15 degrees to +15 degrees, i.e., $-15 \leq j \leq 15$, with an interval of one degree. Hence, the model contains 41x31=1271 templates, and each template represents a specific ALV location. Because the templates are represented in the ICS, the model matching process described later is also operated in the ICS.

2. Detection of Lane Width Changes

The ALV keeps moving forward after an image is taken at the beginning of each navigation cycle. After the image is processed and corresponding algorithms are performed, the ALV location at the time instant of image taking can be found. At this moment, however, the ALV has travelled a distance. Hence, the ALV never knows its actual current position unless the cycle time is zero. To overcome this difficulty, the system uses the ALV control information to estimate the current ALV location according to a method proposed by Cheng and Tsai (1991). We then define the *reference ALV location* in cycle *i* as the estimated ALV location at the beginning time of cycle *i*.

Let T_{rs} denote the reference ALV location (d_r,θ_s) in the current cycle, and let T_{rs} be the *reference template* in the current cycle. To judge whether the left or right lane width has changed, we first check those templates, called *candidate templates*, which are "close" to the reference template in the current cycle. A *candidate template set W* is shown in Fig. 2, where the left line of each template in *W* lies in the area bounded by the left line of the leftmost candidate template (*LCT*) and the left line of the rightmost candidate template (*RCT*), and the right line of each template in *W* lies in the area bounded by the right line of the *LCT* and the right line of the *RCT*. We also call T_{rs} the *center template* in *W*.

Basically, a road in an image can be divided into three clusters according to their intensity values: (1) *cluster-0*: a dark area, like shadows and trees; (2) *cluster-1*: a gray area, coming from the main body of the road; and (3) *cluster-2*: a bright area, like the sky and the white path lines on the road. Then, the ratio of the number of cluster-1 pixels to the number of all the pixels in the area bounded by the left line of the *LCT* and the left line of the *RCT* is checked to judge whether the left lane width has changed. If the ratio is smaller than some threshold value, say *TH-1*, the left lane is decided to have narrowed. If it is larger than some threshold value, say *TH-2*, the left lane is decided to have widened. If it is between *TH-1* and *TH-2*, the left lane width is decided to be unchanged. Similarly, we check the ratio of the number of cluster-1 pixels to the number of all the pixels in the area bounded by the right line of the *LCT* and the right line of the *RCT* to judge whether the right lane width has changed. If both lane widths are unchanged, model matching (MM) is then performed to find the ALV location, which will be described later. If either or both lane

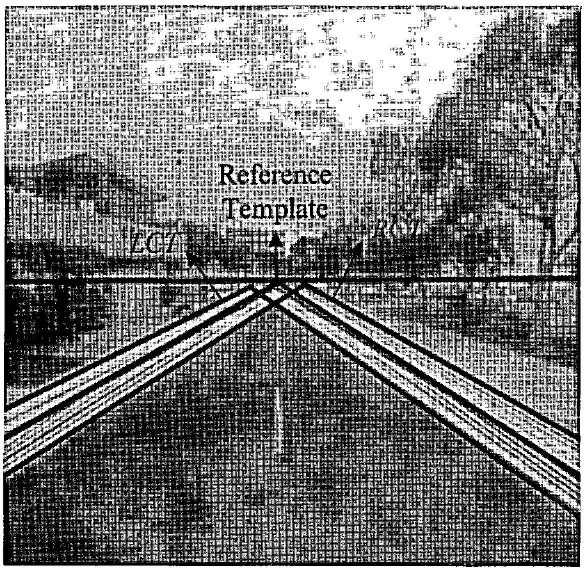

Fig. 2. A candidate template set *W*, where the *LCT* denotes the leftmost candidate template and the *RCT* denotes the rightmost candidate template in *W*.

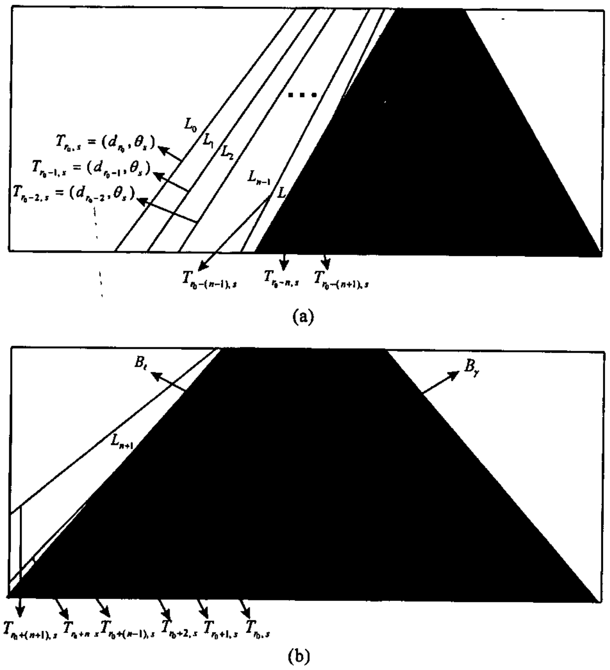

Fig. 3. Illustration of the estimation process for the change in the left lane width. (a) The left lane has narrowed. (b) The left lane has widened.

widths have changed, corresponding processes are executed to calculate the new road width, as stated in the following.

3. Estimation of New Road Width

If the left lane width is checked and found to have changed, then the changed amount W_ℓ of the left lane width is estimated as illustrated in Fig. 3. Figure 3(a) illustrates the estimation process for the case that the left lane has narrowed, where $T_{r_0,s}=(d_{r_0},\theta_s)$ is the reference template, L_i is the left line of template $T_{r_0-i,s}$, and the dark area bounded by B_ℓ and B_γ is composed of the cluster-1 pixels (extracted road pixels). To calculate W_ℓ, we first check the cluster-1 pixels in the area bounded by L_0 and L_1. If they are sufficient in number, then L_0 is decided to be close enough to B_ℓ and is selected as the "approximate left road boundary", and W_ℓ is set to 0. Otherwise, the cluster-1 pixels in the area bounded by L_1 and L_2 are checked. If they are sufficient in number, then L_1 is decided to be close enough to B_ℓ and is selected as the approximate left road boundary, and W_ℓ is set to $d_{r_0-1}-d_{r_0}$. Otherwise, the cluster-1 pixels in the area bounded by L_2 and L_3 are checked in a similar way.

This process is repeated until a certain L_{n+1} is chosen such that the cluster-1 pixels in the area bounded by L_n and L_{n+1} are sufficient in number. Then, L_n is selected as the approximate left road boundary, and W_ℓ is set to $d_{r_0-n}-d_{r_0}$. Note that this method of detecting the approximate road boundary, as proposed above, facilitates the avoidance of direct estimation of road boundary lines from the edge points of broken- or irregular-shaped road boundaries resulting from existing roadside cars or obstacles. On the other hand, if the left lane has widened, a similar criterion is used to find W_ℓ, which is illustrated in Fig. 3(b), where L_n is the approximate left road boundary.

Similarly, if the right lane width has changed, the changed amount W_γ of the right lane width and the "approximate right road boundary" R_n, which is close enough to the real right road boundary B_γ in the image, are estimated in a similar way. After the changed amounts W_ℓ and W_γ of both lane widths are found, the new road width is calculated by

$$\text{New road width } W_{new}$$

$$= \text{old road width } W_{old} + W_\ell + W_\gamma \tag{3}$$

and the *approximate template*, which is defined as composed of the approximate left and right road boundaries, L_n and R_n, is estimated to be

$$T_{ms} = (d_m, \theta_s) = (d_{r_0} + \frac{W_\ell}{2} - \frac{W_\gamma}{2}, \theta_s)\,, \tag{4}$$

where $T_{r_0,s}=(d_{r_0},\theta_s)$ is the reference template. If $W_{new} \neq W_{old}$, then we can recreate the model in the way described in Section II.1 according to the new road width W_{new}. Finally, model matching is performed to find the ALV location, which is described in the following.

4. Model Matching for ALV Location

If both lane widths are detected to be unchanged in the current cycle, we use the candidate template set W described in Section II.2 to perform model matching because the templates in W are close to the reference template, which is the estimated road boundary shape of the unchanged road in the current cycle. If either or both lane widths are detected to have changed due to nearby cars or other obstacles on the road ahead in the current cycle, the reference template could be far away from the real road boundary shape in the image, and it is unsuitable to use the set W described above to perform matching. As described in the previous section, after the changed amounts W_ℓ and/or W_γ are found, we obtain the approximate template T_{ms}, which is close to the real road boundary shape in the image. We thus choose T_{ms} as the center template to reconstruct

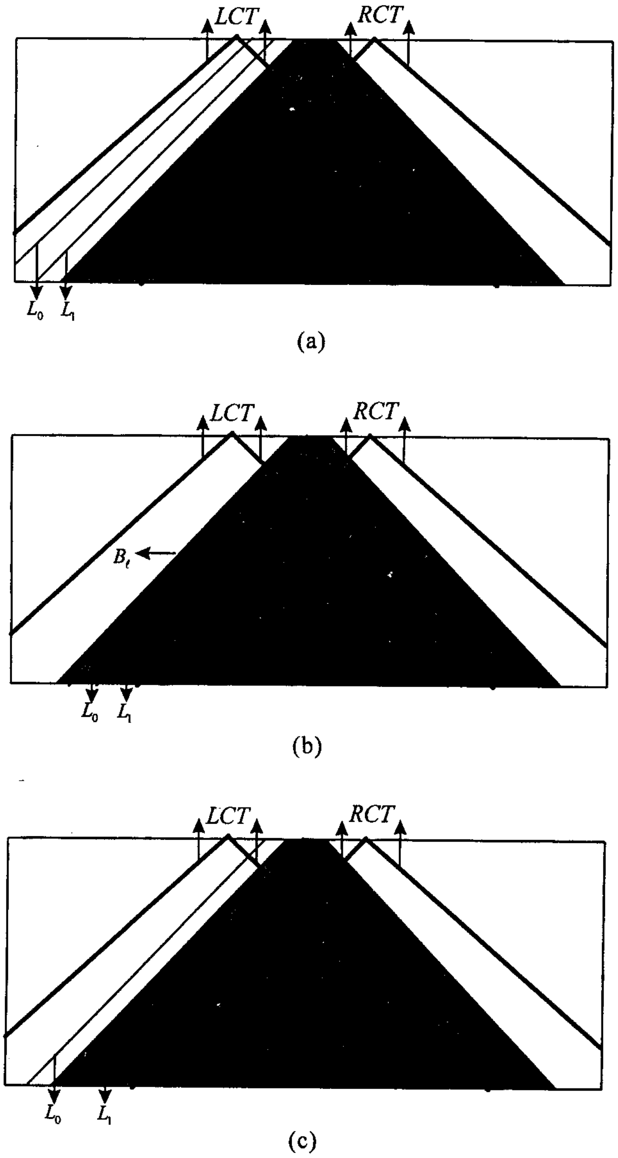

Fig. 4. Illustration of how we check the *LBR* and *LRR* values to decide which line is closer to the real left road boundary B_ℓ in different cases. (a) Two lines are on the left side of B_ℓ. (b) Two lines are on the right side of B_ℓ. (c) One line is on the left side of B_ℓ, and one line is on the right side of B_ℓ.

W such that the templates in the reconstructed set *W* can be made closer to the real road boundary shape.

After *W* is chosen or reconstructed, we use it to perform model matching to find the ALV location. Without loss of generality, we will first state how we match the templates in *W* with the set *LCP* of the left-check-pixels, which are defined as the cluster-1 pixels in the area bounded by the left line of the *LCT* and the left line of the *RCT*, to find the template T_ℓ whose left line is most likely to be the left road boundary. The matching process is based on a criterion which is described as follows. We define:

Left-bounded-ratio LBR_{ij}
=(the number of the cluster-1 pixels in the area bounded by the left line of T_{ij} and the left line of the *RCT*)/(the number of all the pixels in the area bounded by the left line of T_{ij} and the left line of the *RCT*);

and

Left-road-ratio LRR_{ij}
=(the number of the cluster-1 pixels in the area bounded by the left line of T_{ij} and the left line of the *RCT*)/(the number of the *LCP*). **(5)**

Figure 4 illustrates how we check the *LBR* and *LRR* values to decide which line is closer to the real left road boundary B_ℓ in different cases, where L_i is the left line of template T_i. In Fig. 4(a), L_0 and L_1 are on the left side of B_ℓ, and L_1 is closer to B_ℓ. It can be seen from the figure that T_0 and T_1 have the same *LRR* value (=100%). But the *LBR* value of T_1 is greater than that of T_0 because L_1 is closer to B_ℓ. This means that all of the templates in *W* whose left lines are on the left side of B_ℓ have the same *LRR* value, and that the template whose left line is closer to B_ℓ has a larger *LBR* value.

In Fig. 4(b), L_0 and L_1 are on the right side of B_ℓ, and L_0 is closer to B_ℓ. It can be seen from the figure that T_0 and T_1 have the same *LBR* value (=100%). But the *LRR* value of T_0 is greater than that of T_1 because L_0 is closer to B_ℓ. This means that all of the templates in *W* whose left lines are on the right side of B_ℓ have the same *LBR* value, and that the template whose left line is closer to B_ℓ has a larger *LRR* value.

In Fig. 4(c), L_0 and L_1 are on the left and right sides of B_ℓ, respectively. It can seen from the figure that the *LRR* value of T_0 is equal to the *LBR* value of T_1 (=100%). To decide which of L_0 and L_1 is closer to B_ℓ, we can compare the *LBR* value of T_0 with the *LRR* value of T_1. If the *LBR* value of T_0 is larger than the *LRR* value of T_1, then it is decided that L_0 is closer to B_ℓ. Otherwise, it is decided that L_1 is closer to B_ℓ.

From the above observation and analysis, it is reasonable to define the following similarity measure based on the two ratio values, LBR_{ij} and LRR_{ij}, for model matching:

$$LS_{ij}=LBR_{ij}+LRR_{ij}. \qquad \textbf{(6)}$$

Then, the best-matched template $T_\ell = (d_\ell, \theta_\ell)$ whose left line is closest to B_ℓ has the maximum *LS* value. We call this criterion of choosing T_ℓ the maximum-sum-of-double-ratio (MSODR) criterion.

Similarly, we can match the templates in W with the set RCP of the right-check-pixels based on the MSODR criterion to obtain the best-matched template T_γ whose right line is most likely to be the real right road boundary.

After $T_\ell=(d_\ell,\theta_\ell)$ and $T_\gamma=(d_\gamma,\theta_\gamma)$ are found, the ALV location $T=(d,\theta)$ is estimated more accurately by averaging T_ℓ and T_γ, i.e.,

$$T=(d,\theta)=\frac{T_\ell+T_\gamma}{2}=(\frac{d_\ell+d_\gamma}{2},\frac{\theta_\ell+\theta_\gamma}{2})\,. \qquad (7)$$

The proposed MM algorithm based on the MSODR criterion is stated below.

Algorithm MM.

Input. (a) A set W of candidate templates.
    (b) The set LCP of left-check-pixels and the set RCP of right-check-pixels.
Output. The ALV location T.
    Step 1. For each candidate template T_{ij} in W, compute the left and right similarities LS_{ij} and RS_{ij}.
    Step 2. From all of the computed LS_{ij} values, find the best-matched template T_ℓ with the largest LS value.
    Step 3. From all of the computed RS_{ij} values, find the best-matched template T_γ with the largest RS value.
    Step 4. Set $T=(T_\ell+T_\gamma)/2$ and take T as the desired output.

Finally, a complete ALV location algorithm (CALVL) is described below.

Algorithm CALVL.

Input. (a) A set V of cluster-1 pixels.
    (b) A model M of templates created in the previous cycle.
    (c) The reference template $T_{r_0,s}=(d_{r_0},\theta_s)$ in the current cycle.
    (d) A set W of candidate templates in M whose center template is $T_{r_0,s}$.
Output. The ALV location T.
    Step 1. Check of lane widths:
        (a) Use V and W to check whether the left lane width has changed. If it has changed, compute the changed amount W_ℓ of the left lane width.
        (b) Use V and W to check whether the right lane width has changed. If it has changed, compute the changed amount W_γ of the right lane width.
    Step 2. Model recreation and candidate template set reconstruction:
        If either or both lane widths have changed, then perform:
        (a) Calculate the new road width W_{new}=the old road width $W_{old}+W_\ell+W_\gamma$.
        (b) If $W_{new}\neq W_{old}$, then recreate the model M according to W_{new}.
        (c) Estimate the approximate template $T_{ms}=(d_m,\theta_s)=(d_{r_0}+W_\ell/2-W_\gamma/2,\theta_s)$.
        (d) Use T_{ms} as the center template to reconstruct the candidate template set W.
    Step 3. Model matching:
        Use V and W to find the LCP and the RCP, and with W, LCP, and RCP as inputs run the MM algorithm to find the desired ALV location T as the output.

After the desired ALV location is found, the navigation path on the extracted new road can be generated immediately. A turn angle for guiding the ALV from the current ALV location to the immediately generated navigation path is then computed. The navigation path estimation and the turn angle computation are described in the following.

III. Navigation Guidance and Vehicle Control

1. Navigation Guidance

To achieve safe navigation, we choose the central line on the extracted road as the navigation path. Let $T=(d,\theta)$ denote the obtained ALV location in the current cycle, where d is the distance from the ALV to the central path line in the road (positive to the right) and θ is the pan angle of the ALV with respect to the road direction (positive to the left), as described previously. As shown in Fig. 1, the equation of the navigation path in the VCS in the current cycle is just

$$x\cos\theta-y\sin\theta+d=0. \qquad \textbf{(8)}$$

To guide the ALV from the current estimated location to follow the navigation path, we have to compute an appropriate turn angle for the ALV. For this, a closeness distance measure from the ALV to the navigation path proposed by Cheng and Tsai (1991) is used, which is defined as

$$L(\delta)=\frac{1}{1+[D_F(\delta)]^2+[D_R(\delta)]^2}\,, \qquad \textbf{(9)}$$

where D_F and D_R are the corresponding distances from

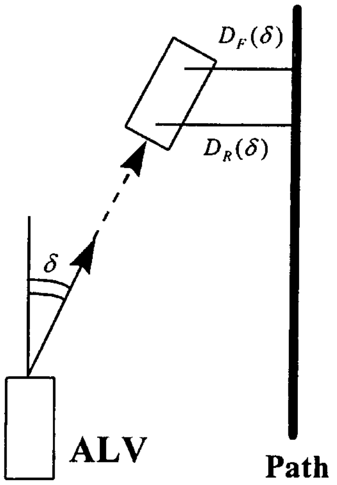

Fig. 5. Illustration of the meaning of the closeness distance measure $L(\delta)=1/\{1+[D_F(\delta)]^2+[D_R(\delta)]^2\}$.

the front and the rear wheels of the ALV to the navigation path after the ALV traverses a distance with the turn angle δ, as illustrated in Fig. 5. A larger value of L means that the ALV is closer to the path. It is easy to verify that $0<L\leq 1$, and that $L=1$ if and only if both the front wheels and the rear wheels of the ALV are located just right on the path.

To find the turn angle of the front wheel to drive the ALV as close to the path as possible, a range of possible turn angles is searched. An angle is hypothesized each time, and the value of L is calculated accordingly. The angle that produces the maximal value of L is then used as the desired turn angle in this study.

It should be mentioned that allowing the ALV a larger angle to turn in a session of turn drive does not mean that better navigation can be achieved. It may cause serious twisting. On the other hand, a smaller range of turn angles may bring the ALV slightly closer to the given path. Hence, the largest angle allowing the ALV to turn is a tradeoff between smoothness of navigation and closeness to the given path. In our experiment, we found through many iterative navigation runs that a turn from -5 degrees to +5 degrees is a good compromise.

2. ALV Speed Adjustment on Varying Road Situations

To achieve steady and flexible navigation, the speed of the ALV is adjusted under varying road situations. The speed is adjusted proportionally to the road width for flexible navigation; i.e., it is increased if the road has widened, decreased if the road has narrowed, and kept unchanged if the road width has not changed. The speed is also adjusted according to varying road surface heights; i.e., it is increased if the ALV moves on an ascending road and decreased if the ALV moves on a descending road for steady navigation. To drive the ALV at an appropriate speed during navigation, we propose the following solution. Let V_i and W_i denote the actual ALV speed and the obtained road width in navigation cycle i. Then, to adjust the speed proportionally to the road width during navigation, the desired speed V_i' of the ALV in cycle i is computed as

$$V_i' = (\frac{W_{i-1}}{W_{i-2}})V_{i-1}\,. \qquad \textbf{(10)}$$

As we guide the ALV in cycle i, we can find its travelled distance S_i, the elapsed time T_i, and its driving power P_i by checking the system encoder, the system clock, and the motor controller provided by the control system. Then, the actual speed of the ALV in cycle i is calculated by

$$V_i = \frac{S_i}{T_i}\,. \qquad \textbf{(11)}$$

In cycle i, if $V_i>V_i'$, it is decided that the ALV is on a descending road, and if $V_i<V_i'$, the ALV is on an ascending road. If a ascending or descending road is detected in cycle i, we multiply the driving power by the ratio V_i'/V_i to balance the speed of the ALV for steady navigation. On the other hand, if the road width changes from cycle i–1 to cycle i, i.e., if $W_i\neq W_{i-1}$, then we multiply the driving power by the ratio W_i/W_{i-1} to change the speed of the ALV for flexible navigation. Hence, when it is necessary to take both situations described above into consideration, we multiply the driving power by (V_i'/V_i) (W_i/W_{i-1}) to obtain a proper speed for steady and flexible navigation. That is, a reasonable driving power for the ALV in cycle i+1 can be taken to be

$$\begin{aligned} P_{i+1} &= (\frac{V_i'}{V_i})(\frac{W_i}{W_{i-1}})P_i \\ &= (\frac{W_{i-1}\bullet V_{i-1}}{W_{i-2}\bullet V_i})(\frac{W_i}{W_{i-1}})P_i \\ &= (\frac{W_i\bullet V_{i-1}}{W_{i-2}\bullet V_i})P_i\,. \qquad \textbf{(12)} \end{aligned}$$

IV. Image Processing and Feature Extraction Techniques

To reduce the image size, the upper portion in the road image is discarded because it does not contain any

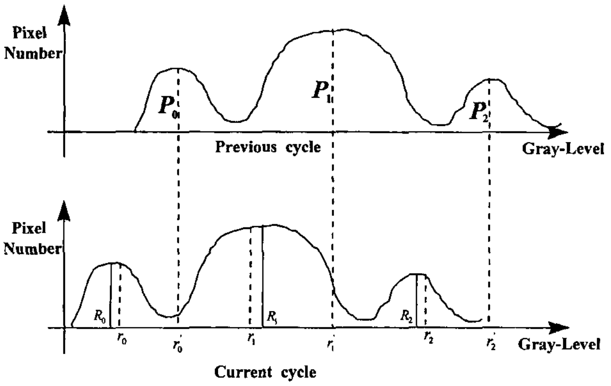

Fig. 6. The histograms of the *R*-planes of two consecutive images, where r_k' is the *R*-component of the resulting center of cluster *k* in the previous cycle, R_k is the *R*-component of the real center of cluster *k* in the current cycle, and r_k is the *R*-component of the candidate center of cluster *k* resulting from applying the ICC technique in the current cycle.

road area. Next, pixels are sampled from the remaining image portion with an interval of six pixels in both the horizontal and vertical directions. We then use an ISODATA algorithm (Duda and Hart, 1973), which includes an initial-center-choosing (ICC) technique that can solve the problem caused by great changes of intensity during navigation, to divide the road image into three clusters.

Intuitively, we can select the resulting centers in the previous navigation cycle as the initial centers in the current cycle to run the clustering algorithm. However, this selection may be unsuitable, thus producing unacceptable clusters, because some difference may exist between two consecutive images. One kind of the difference comes from the change of intensity. If the change of intensity between two consecutive images is great, the candidate initial centers chosen from the resulting centers in the previous cycle may be far away from their real centers in the current cycle. In this situation, many iterations are needed for the ISODATA algorithm to move the candidate initial centers close to their real centers, and this takes too much computing time. Figure 6 shows this situation, where the *R*-component r_k' of the resulting center of cluster *k* in the previous cycle is far away from the *R*-component R_k of the real center of cluster *k* in the current cycle because of the great change of intensity between the two consecutive images.

To choose proper initial centers which are close to their real centers in the current cycle, we propose an ICC technique based on the assumption that the changes of bright, gray, and dark areas between two consecutive input images are small. ·As shown in Fig. 6, the *R*-component r_k of the candidate initial center of cluster *k* resulting from the ICC technique is very close to the *R*-component R_k of the real center of cluster *k* in the current cycle. The ICC technique is described as follows. Let P_0, P_1, and P_2 be the numbers of pixels belonging to cluster-0, cluster-1, and cluster-2 in the previous cycle, respectively. What we want to compute is the initial centers of the three clusters in order to run the clustering algorithm in the current cycle based on the values of P_0, P_1, and P_2. We first observe the histogram of the *R*-plane shown in Fig. 6 and let r_0, r_1, and r_2 be such that the following equalities are satisfied:

$$\sum_{s=0}^{r_0} [\text{pixel no. of g.1.}(s)] = P_0/2,$$

$$\sum_{t=0}^{r_1} [\text{pixel no. of g.1.}(t)] = P_0 + P_1/2,$$

and

$$\sum_{u=0}^{r_2} [\text{pixel no. of g.1.}(u)] = P_0 + P_1 + P_2/2, \qquad \textbf{(13)}$$

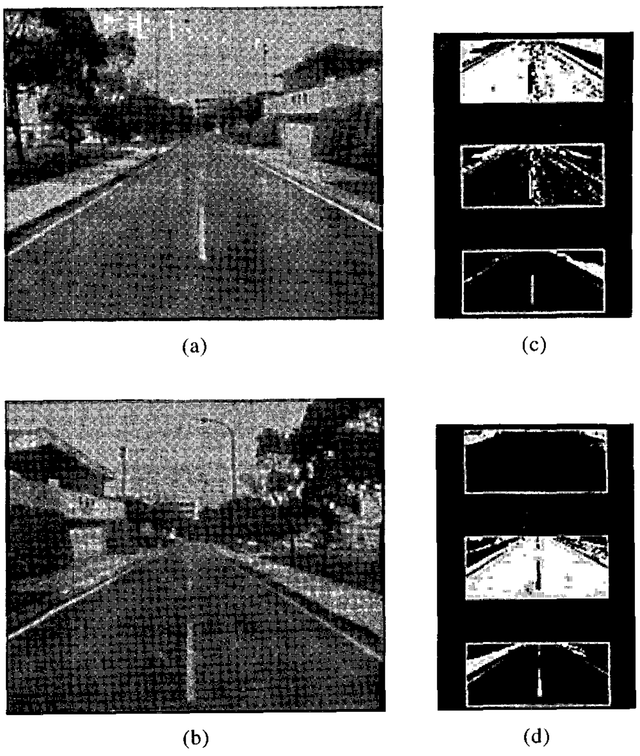

Fig. 7. Obvious improvement obtained using the ICC technique on a real road scene. (a) The input image with high intensity in the previous cycle. (b) The input image with low intensity in the current cycle. (c) Poor clustering result when the resulting centers in the previous cycle are used as the initial centers to run the clustering algorithm for three iterations in the current cycle. (d) Better clustering result produced by the ICC technique.