Vision-Based Autonomous Land Vehicle Guidance in Outdoor Road Environments Using Combined Line and Road Following Techniques

Kuang-Hsiung Chen
Wen-Hsiang Tsai*
Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 300, R.O.C.

Received January 18, 1996; accepted May 13, 1997

An intelligent approach to autonomous land vehicle (ALV) guidance in outdoor road environments using combined line and road following and color information clustering techniques is proposed. Path lines and road boundaries are selected as reference models, called the line-model and the road-model, respectively. They are used to perform line-model matching (LMM) and road-model matching (RMM) to locate the ALV for line and road following, respectively. If there are path lines in the road, the LMM process is used to locate the ALV because it is faster than the RMM process. On the other hand, if no line can be found in the road, the RMM process is used. To detect path lines in a road image, the Hough transform is employed, which does not take much computing time because bright pixels in the road are very few. Various color information on roads is used for extracting path lines and road surfaces. And the ISODATA clustering algorithm based on an initial-center-choosing technique is employed to solve the problem caused by great changes of intensity in navigations. The double model matching procedure combined with the color information clustering process enables the ALV to navigate smoothly in roads even if there are shadows, cars, people, or degraded regions on roadsides. Some intelligent methods to speed up the model matching processes and the Hough transform based on the feedback of the previous image information are also presented. Successful navigations show that the proposed approach is effective for ALV guidance in common roads. *© 1997 John Wiley & Sons, Inc.*

*To whom all correspondence should be sent.

Journal of Robotic Systems 14(10), 711–728 (1997)

CCC 0741-2223/97/0100711-18

線路と道路の組み合わせの追従および色情報のクラスタ化の技術を使って、屋外の道路環境における自動陸上車両（ALV）の誘導を行う、インテリジェントな手法を提案する。経路の線路と道路の境界は参照モデルとして選択され、それぞれ、ラインモデルおよびロードモデルと呼ばれる。これらは、線路と道路に追従するALVの位置を決めるための、ラインモデル・マッチング（LMM）とロードモデル・マッチング（RMM）に使われる。経路の線路が道路の中にある場合は、LMM処理によってALVの位置が決定される。これは、LMM処理の方がRMMよりも高速なためである。一方、道路の中に線路が見つからない場合は、RMM処理が使われる。道路のイメージから経路の線路を検出するために、Hough変換を採用している。これは、道路の中の明瞭なピクセルは極わずかなので、それほど多くの計算時間はかからない。道路上の様々な色情報は、経路の線路と道路表面を抽出するために使われる。そして、初期中心選択技法に基づいたISODATAクラスタ化アルゴリズムを使って、航行における輝度の大幅な変化によって起こる問題を解決する。そして、2つのモデル・マッチング手法と色情報のクラスタ化処理を組み合わせることで、影、他の車両、人、段差などが路肩にあっても、道路上のALVを円滑に運行できる。さらに、以前のイメージ情報をフィードバックすることで、モデル・マッチング処理とHough変換を高速化するインテリジェントな方法も紹介する。最後に、提案した方法で、一般道路上にあるALVを効果的に誘導できることを示す。

1. INTRODUCTION

Autonomous land vehicles (ALVs) are useful for many automation applications. Successful ALV navigation requires integration of techniques of environment sensing, ALV location, path planning, wheel control, etc. This study is mainly concerned with ALV guidance in outdoor road environments using computer vision techniques. Many techniques have been proposed for ALV guidance in outdoor roads[1–15] and indoor environments.[16–19] In outdoor environments, because of the great variety of road conditions like shadows, degraded regions, moving objects, changes of illumination, and even rain, we need to combine different problem-solving algorithms and perhaps equip multiple sensors to solve the complex problem of ALV guidance in roads.

The CMU Navlab[1–8] is a system with several function-oriented categories such as SCARF, YARF, ALVINN, UNSCARF, and EDDIE. SCARF[6,8] handled unstructured roads. It recognized roads having degraded surfaces and edges with no lane markings in shadow conditions. It also recognized intersections automatically with no supervised information. YARF[6] dealt with structured roads. It guided individual trackers using an explicit model of features. It could drive the vehicle on urban roads at speeds up to 96 kph, which is much higher than the speed of SCARF. ALVINN[4,6] preprocessed input images, put them to a neural net, and instantly obtained the wheel angles from the network. UNSCARF[5] separated unstructured roads into homogeneous regions using an unsupervised clustering technique.[20] A shrink and expand algorithm[21] was used to remove small regions. A minimum distance criterion based on a Grassfire transformation[22] is used to obtain the most reasonable interpretation. EDDIE[7] was a software architecture that combined separate functional parts like sensing, planning, and control into an intelligible system. Additionally, Kluge and Thorpe[3] split the assumptions made in road modeling into three loose classes: subconscious, implicit, and explicit models. Moreover, a new road tracking system called FERMI was designed to explain explicit models and their purposes.

VITS[9] and Navlab[2] both used color information in RGB planes for road following. Turk et al.[9] showed that the G-component is redundant in their road analysis. Navlab[2] deduced both color and texture segmentation results using a pattern classification method involving the mean and the covariance matrix. Lin and Chen[10] divided roads into three clusters: sunny road, shadow road, and nonroad. Their classification is based on the Karhunen-Loève transform in the HSI color space. The Germanic vision system[11–13] used a high-speed vision algorithm to detect road border lines. The system has carried out both road following and vehicle following in real time. Kuan, Phipps, and Hsueh[14] proposed a technique with transformation and classifier parameters being updated continuously and cyclically with respect to the slow change of color and intensity. Hughes Research Laboratories[15] designed a planning method using digital maps to route a desired path. When the vehicle went through the path, sensors were used to provide environmental descriptions.

In this article, we propose an intelligent approach to ALV guidance in outdoor road environments using combined line and road following and color information clustering techniques. Although a road scene is often uncertain and noisy in an outdoor environment, there are still stable features that can be extracted from a road image for use as reference models. Hence, model matching is a quite reasonable approach for ALV guidance in outdoor environments. We use path lines and road boundaries, which are apparent in common roads, to construct reference models, called the line-model and the road-model. Line-model matching (LMM) and road-model matching (RMM) are then used to locate the ALV for line and road following, respectively. Because the LMM process is much faster than the RMM process, the system always uses the LMM process to locate the ALV if any path line is detected. Only when no path line exists does the system use the RMM to guide the ALV. By combining the two matching processes in this way, faster and more flexible navigations in general roads can be achieved.

Furthermore, various color information on roads is used in this study to extract path lines and road surfaces. For this, the ISODATA algorithm[20] based on an initial-center-choosing (ICC) technique is employed, which can solve the problem caused by great changes of intensity in navigations. Proper initial centers are chosen as the clustering algorithm begins to run. To detect path lines in a road image, the Hough transform[23] is used, which can find the path lines in the image even though they are dotted or noisy. The transform does not take much computing time because bright pixels in the road are very few.

The above double model matching procedure combined with the color information clustering process enables the ALV to navigate on the road smoothly even though there are shadows, cars, people, or degraded regions on the roadsides. Besides, some intelligent methods to speed up the model matching processes and the Hough transform, based on the feedback of the previous image information, are also presented. The proposed approach is proved effective after many practical navigation tests.

The main difference of our approach from existing methodologies is that the proposed system can switch visual features dynamically at the appropriate time during navigation for improving the system's efficiency. And different visual features can be extracted directly from the clustering result. Also, new and effective model matching methods are used to locate the ALV accurately.

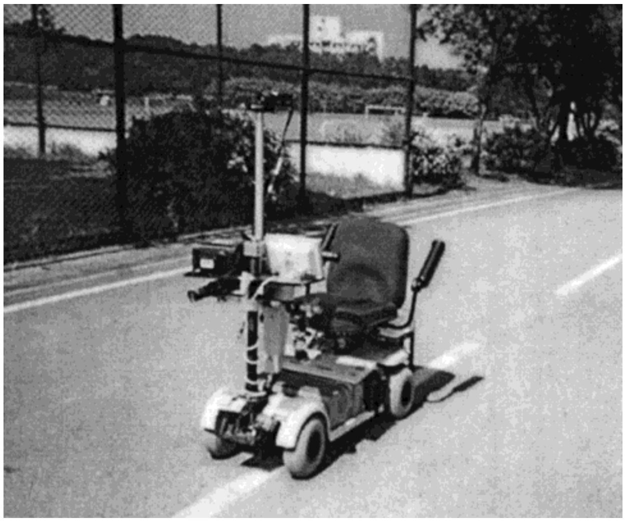


Figure 1. The prototype ALV used in this study.

A new prototype ALV (whose dimensions are 118.5 by 58.5 cm) with smart, compact, and ridable characteristics, as shown in Figure 1, is constructed as a testbed for this study. It has four wheels in which the front two are the turning wheels and the rear two the driving wheels. Above the front wheels is a cross-shaped rack on which some CCD cameras are mounted, and above the rack is a platform on which two monitors, one being the computer monitor and the other the image display, are placed. Above the platform is a vertical bar on which another camera used for line and road following in this study is mounted. The central processor is an IBM PC/AT compatible personal computer (PC486) with a color image frame grabber that takes 512 × 486 RGB images, with eight bits of intensity per image pixel.

The ALV is computer-controlled with a modular architecture, as shown in Figure 2, including four components, namely, a vision system, a central processor PC486, a motor control system, and a DC power system. The vision system consists of a camera, a TV monitor, and a Targa Plus color image frame grabber. The central processor PC486 has an RAM with two mega bytes, one floppy disk, a 120-mega-byte hard disk, a 300-mega-byte hard disk, and an EL slim display. The motor control system includes a main control board with an Intel 8085 controller, a motor driver, and two motors. The power of the system is supplied by a battery set including two 12-volt power sources, each being divided into various voltages using a DC-to-DC converter set to provide power to the ALV components.

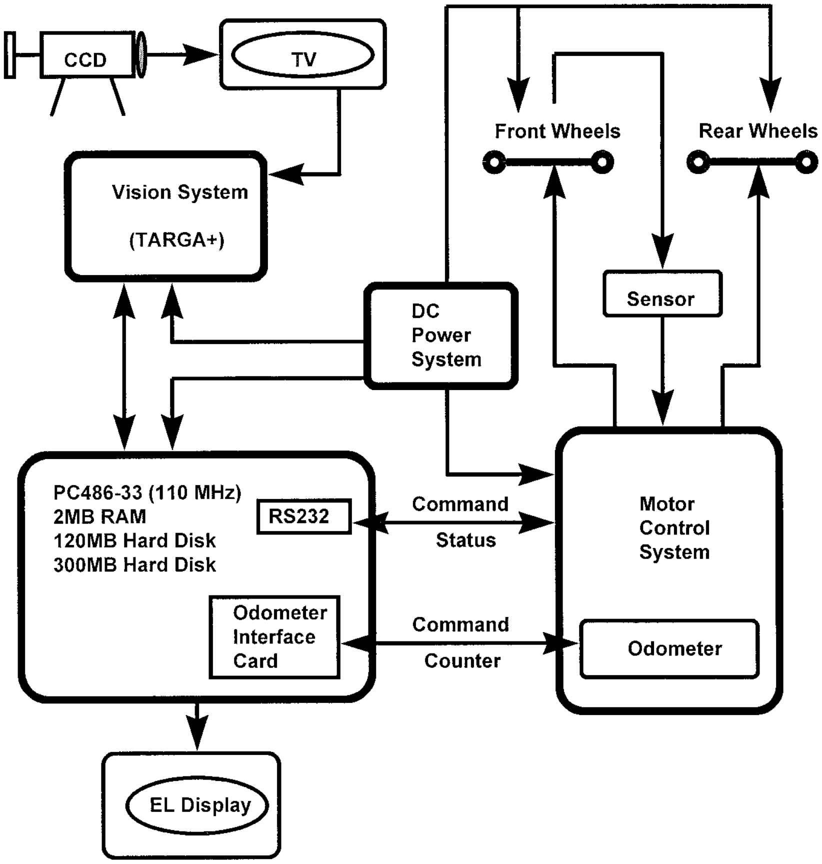


Figure 2. System structure of prototype ALV.

The remainder of this article is organized as follows. In section 2, the details of the proposed model-based ALV navigation method is described. In section 3, the details of the proposed ALV location method is introduced. The description of the used image processing techniques is included in section 4. In section 5, the results of some experiments are described. Finally, some conclusions are made in section 6.

2. PROPOSED MODEL-BASED ALV NAVIGATION METHOD

The proposed navigation scheme is performed in a cycle by cycle manner. An overview of the proposed approach is shown in Figure 3. To finish each navigation cycle, the system identifies some stable features in the road environment to locate the ALV, and plan a smooth path from the current position of the ALV to the goal (or to a given navigation path). After the two reference models are created, the system performs individual matchings with these models to determine the ALV location. If any path line is detected in the road, the LMM process is used to guide the ALV because it is faster than the RMM process. On the other hand, if no line can be found in the road, the RMM is used.

In the following, we first describe the steps of model creation, which involve several coordinate systems and coordinate transformations. Then, the line-model matching process for line following and the road-model matching process for road following are introduced, followed by a description of the combined line and road following technique.

2.1. Model Creation

Several coordinate systems and coordinate transformations are used in this approach. The coordinate systems are shown in Figure 4. The image coordinate system (ICS), denoted as u-w, is attached to the image plane. 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 with the ground. The x-axis and y-axis are on the ground and parallel to the short side and the long side 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[18–19, 26] as

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

where

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

where θ is the pan angle, ϕ the tilt angle, and φ 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 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 of the ALV to the central path line in the road and θ is the pan angle of the ALV relative to the road direction (positive to the left). The equations of the three path lines on the road in the VCS are assumed to be known. Then, we transform them into the ICS, which is displayed on the TV monitor. For each ALV location (d_i, θ_j), we can create the corresponding line-templates $LINE_{ij}[al, bl, am, bm, ar, br]$, where al, am, and ar are the slopes, and bl, bm, and br are the intercepts of the equations of the left, middle, and right path lines in the ICS, respectively. The transformation is shown in Figure 5(a). We sample the road width at 23 positions with the interval of 25 cm. At each position, we sample the vehicle direction from −16° to +16° at 17 angles with the interval of 2°. So, the line-model contains $23 \times 17 = 391$ line-templates, and each line-template represents one specific ALV location.

We use the same method to create the road-model. The road-model also contains $23 \times 17 = 391$ road-templates, and each road-template $ROAD_{ij}[al, bl, ar, br]$ also represents an ALV location (d_i, θ_j). The transformation is shown in Fig. 5(b). Thus, each ALV location $T = (d_i, \theta_j)$ can be represented by a line-template $LINE_{ij}[al, bl, am, bm, ar, br]$ or a road-template $ROAD_{ij}[al, bl, ar, br]$, and vice versa. We say that line-template (or road-template) $T_i = (d_i, \theta_i)$ is similar to line-template (or road-template) $T_j = (d_j, \theta_j)$ (denoted by $T_i \cong T_j$) if and only if

$$|d_i - d_j| \le 25\ (\text{cm}) \text{ and } |\theta_i - \theta_j| \le 2\ (\text{degree}). \quad (3)$$

Because the line-templates and the road-templates are represented in the ICS, the model matching processes described in the following are also operated in the ICS.

2.2. Line-Model Matching for Line Following

When path lines exist in the road image, they are extracted and matched with a few candidate line-templates in the line-model, i.e., the LMM is performed. Without loss of generality, we first take the extracted left path line to perform the matching if the path line exists. We want to find the line-template T_L whose left path line is the most similar to the extracted left path line. We define the following similarity measure

$$S = 1/[\alpha \cdot |a_t - a_e| + \beta \cdot |b_t - b_e|] \quad (4)$$

where α and β are two weighting coefficiencies, and a_e and a_t are the slopes, and b_e and b_t are the intercepts, of the extracted left path line and the left path line of the candidate line-template, respectively. Then, the desired line-template $T_L = (d_L, \theta_L)$ is the one that has the maximum S value. If the extracted path lines include the middle or the right path line, we use the same criterion to obtain their corresponding most-similar line-templates $T_M = (d_M, \theta_M)$ and $T_R = (d_R, \theta_R)$, respectively. After T_L, T_M and T_R are computed, the most meaningful ALV location T is derived by the following majority-vote rule.

Case I (T_L, T_M, and T_R are all similar):

if $T_L \cong T_M \cong T_R$,

then set $T = (T_L + T_M + T_R)/3$

$= ((d_L + d_M + d_R)/3, (\theta_L + \theta_M + \theta_R)/3);$

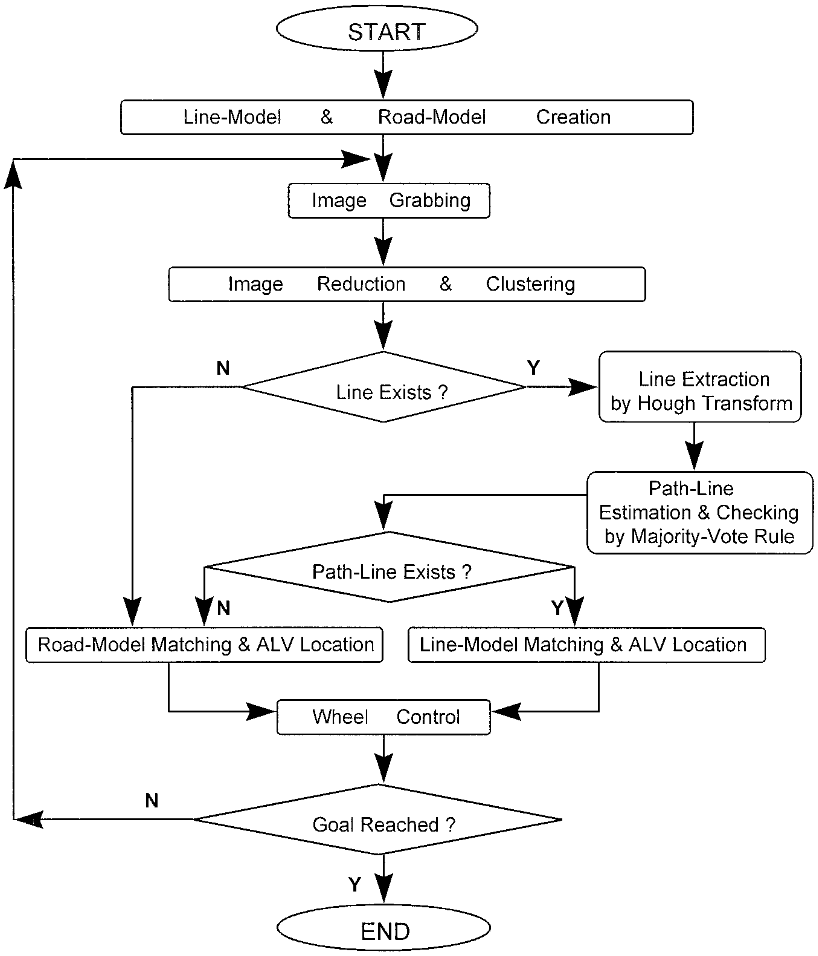

START
Line-Model & Road-Model Creation
Image Grabbing
Image Reduction & Clustering
N
Line Exists ?
Y
Line Extraction by Hough Transform
Path-Line Estimation & Checking by Majority-Vote Rule
Path-Line Exists ?
N
Y
Road-Model Matching & ALV Location
Line-Model Matching & ALV Location
Wheel Control
N
Goal Reached ?
Y
END

Figure 3. An overview of the proposed approach.

Case II (only two of T_L, T_M, and T_R are similar):

if only $T_i \cong T_j\ (i \neq j, i, j = L, M, R)$,

then set $T = (T_i + T_j)/2 = ((d_i + d_j)/2, (\theta_i + \theta_j)/2)$;

Case III (T_L, T_M, and T_R are not similar to each other):

if T_K is the most similar to the reference line-template (i.e., with the maximum S value) where $K = L$, M, or R, then set $T = T_K$.

The reference line-template mentioned above is the one at the reference ALV location estimated by the system at the beginning of each navigation cycle. The estimation of the reference ALV location will be described later. If no noise exists on the road, we can derive a very accurate ALV location because three lines can be used. If noise appears on the right roadside, we can use the left and the middle lines to locate the ALV. Similarly, we can locate the ALV using the right and the middle lines if noise appears on the left roadside. Moreover, if noise exists on both sides, the middle line, which is seldom affected by shadows or other noise, can also be used to locate the ALV. Finally, when no line can be found on the road, we can perform the road-model matching process (described later) to locate the ALV. Hence, we can locate the ALV even when there are shadows, people, cars, or degraded regions on the roadsides. This flexible guidance process makes the ALV navigation steady. The LMM algorithm is described below.

LMM Algorithm

Input: A set E of extracted path lines and a set U of candidate line-templates.

Output: The most meaningful ALV location T.

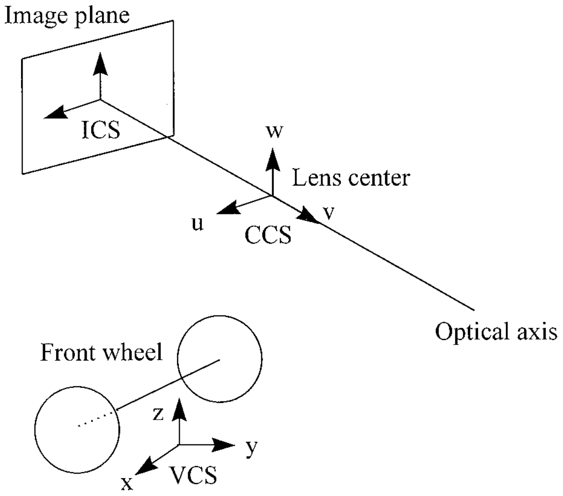


Figure 4. The three coordinate systems ICS, CCS, and VCS.

```
Step 1. For each extracted path line $y = a_E x + b_E$ in $E$:
    (a) for each candidate line-template $LINE_{ij}[a_L, b_L, a_M, b_M, a_R, b_R]$ in $U$:
        compute $S_{ij} = 1/[\alpha \cdot |a_E - a_K| + \beta \cdot |b_E - b_K|]$, where $K = L, M$, or $R$;
    (b) for all of the computed $S_{ij}$ values:
        use the sequential search algorithm to find the best-matched line-template $T_K$ having the largest $S_{ij}$ value, where $K = L, M$, or $R$.
Step 2. From all of the obtained best-matched line-templates (with the number at most three):
    derive the most meaningful ALV location T according to the majority-vote rule described above.
```

It is not necessary to use the entire line-model to perform the matching with the extracted path lines. We only use the neighboring ones in the model around the reference line-template to save computing time. To analyze the time requirement of the LMM algorithm, let

1. the number of the candidate line-templates to perform the matching be k_1;
2. the time to compute the similarity value be k_2;
3. the time to run the algorithm of the majority-vote rule be k_3,

where k_1 is constant, and k_2 and k_3 can be easily shown to be constants by observing Eq. (2) and the majority-vote rule, respectively. The time to process Step 1(b) in the LMM algorithm using the sequential search algorithm can easily be shown to be k_1.[27] Then, for each extracted path line, the time to calculate the similarities of all the candidate line-templates plus the time to find the best line-template having the maximum S_{ij} value can be calculated by

$$T_{L1} = k_1 \times k_2 + k_1. \tag{5}$$

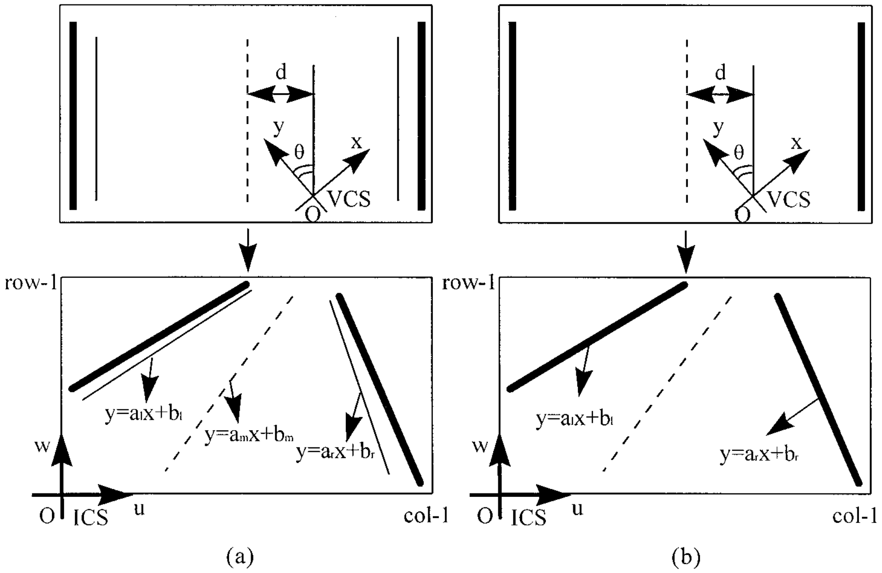


Figure 5. The transformation between the VCS and the ICS. (a) Path line transformation. (b) Road boundary transformation.

Because there are at most three path lines that can be extracted and Step 2 takes k_3 time, the total time used by the LMM algorithm can be calculated by

$$\begin{aligned} T_L &= 3 \times T_{L1} + k_3 \\ &= 3 \times k_1 \times k_2 + 3 \times k_1 + k_3 \end{aligned} \tag{6}$$

which is a constant. Consequently, the time complexity of the LMM algorithm is of the order $O(1)$, which does not depend on the image size.

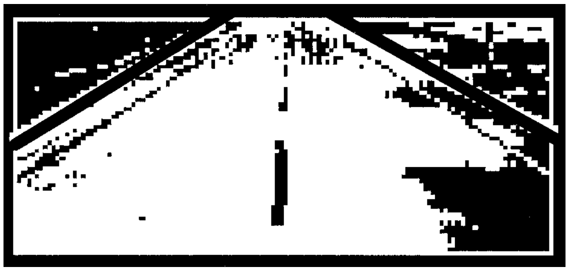


Figure 6. The matched road-template represented by the two black lines based on the MBPNM criterion.

2.3. Road-Model Matching for Road Following

If no high peak is found in the $\rho - \theta$ Hough counting space, or if the extracted line-templates are not similar to each other and none of them is similar to the reference line-template, we conclude that no path line exists on the road. We then perform the more time-consuming road-model matching (RMM) based on a criterion proposed in this study, called the maximum-bounded-pixel-number matching (MBPNM) described as follows. Basically, a road can be divided into three clusters: (1) cluster-0: dark area, like shadows and trees; (2) cluster-1: gray area, coming from the main body of road; (3) cluster-2: bright area, like the sky and the white or yellow lines on the road. We define the bounded-area for each navigation cycle as that bounded by the two lines of the road-template, and the bounded-pixels as those belonging to cluster-1 (most likely to be the road area) in the bounded-area. If a road-template includes within its bounded-area the largest number of bounded-pixels, it can be regarded as the one most likely to be the road boundary shape, i.e., as the best-matched road-template. The RMM algorithm based on MBPNM is described below.

RMM Algorithm

```
Input: A set $V$ of pixels belonging to cluster-1 and a
    set $W$ of candidate road-templates.
Output: The most meaningful ALV location $T$.
Step 1. For each candidate road-template $T_i$ in $W$:
    (a) initialize the number of bounded-pixels
        $N_i = 0$;
    (b) for each pixel $p_j$ in the bounded-area of $T_i$:
        if $p_j$ is also in $V$, then set $N_i = N_i + 1$.
Step 2. For all of the computed $N_i$ values:
    use the sequential search algorithm to find the
    best-matched road-template $T$ having the largest
    $N_i$ value and take $T$ as the desired output.
```

If the best-matched road-template is not similar to the reference road-template at the reference ALV location, we assume that the visual features are missing in the present navigation cycle and drive the ALV blindly, i.e., using only the vehicle control information. If such a case happens for several continuous navigation cycles, we assume that the vision system has lost its function and the vehicle is stopped. As an example, a binary image showing the cluster-1 pixels and the best-matched road-template represented by the two black lines obtained with the MBPNM criterion is shown in Figure 6. It can be seen that the shadow area is included in the bounded-area, i.e., it is regarded as part of the road. So, we can say that the adopted MBPNM criterion is not sensitive to shadows. Also, we do not use the entire road-model to do the matching because it wastes too much time. We only use the neighboring ones in the model around the reference road-template (forming the set W in the above RMM algorithm) to perform the matching. To analyze the time requirement of the RMM algorithm, let

1. the image size be $I = m \times n$ pixels;
2. the number of the candidate road-templates be k_1;
3. the number of the pixels in the bounded-area of the road-template T_i be $B_i = c_i \times I$ with $c_i < 1$;
4. the number of the bounded-pixels of the road-template T_i be $N_i = d_i \times B_i$ with $d_i < 1$;
5. the time to check whether one pixel belongs to cluster-1 or not be k_2; and
6. the time to increment a variable be k_3,

where k_1, k_2, k_3, c_i, and d_i are constants. Then, the time to calculate all the numbers of bounded-pixels of all the candidate road-templates (Step 1 in the RMM algorithm) can be expressed as

$$\begin{aligned} T_{R1} &= \sum_{i=1}^{k_1} \sum_{j=1}^{B_i} [k_2 + (N_i/B_i) \times k_3] \\ &= \sum_{i=1}^{k_1} \sum_{j=1}^{c_i \times m \times n} [k_2 + d_i \times k_3] \\ &= \sum_{i=1}^{k_1} ((k_2 + d_i \times k_3) \times c_i \times m \times n) \end{aligned} \tag{7}$$

which is of the order $O(mn)$. Also, the time to perform Step 2 can be expressed as

$$T_{R2} = k_1 \tag{8}$$

which is of the order $O(1)$. Hence, the total time of the RMM algorithm can be calculated by

$$T_R = T_{R1} + T_{R2} \tag{9}$$

which is of the order $O(mn) + O(1)$ or simply $O(mn)$. Consequently, the time complexity of the RMM algorithm depends on the image size.

2.4. Combination of Line and Road Following

If only path lines or only road surfaces are selected as visual features, navigation may still be accomplished. But because we have shown that the LMM algorithm is much faster than the RMM algorithm, we prefer line following to road following if there are path lines on the road, to accomplish faster navigation. On the other hand, if only path lines are used, the vehicle cannot navigate on roads without path lines. Hence, we employ a combined technique to achieve faster and more flexible navigations. The details are described by the following combined line and road model matching (CLRMM) algorithm.

CLRMM Algorithm

```
Input:
    (a) A road image $I$.
    (b) The reference line-template and the reference road-template.
    (c) Two sets $V_1$ and $V_2$ of pixels belonging to cluster-1 (road area) and cluster-2 (path lines) in I, and two sets $W$ and $U$ of candidate road-templates and candidate line-templates, respectively.
Output: The ALV location $T$.
Step 1. Initialization:
    (a) Initialize Error-flag = 0.
    (b) Check in I the surrounding area $A$ of the three lines of the reference line-template.
    (c) If pixels belonging to cluster-2 in the surrounding area $A$ are insufficient (i.e., if area $A \cap V_2$ is too small), then go to Step 3.
Step 2. LMM:
    (a) Extract a set $L$ of path lines from $V_2$ by the Hough transform.
    (b) Use $L$ and $U$ as the inputs to run the LMM algorithm to find the candidate ALV location $T_c$.
    (c) If $T_c$ is similar to the reference line-template, then go to Step 4.
Step 3. RMM:
    (a) Use $V_1$ and $W$ as the inputs to run the RMM algorithm to find the candidate ALV location $T_c$.
    (b) If $T_c$ is not similar to the reference road-template, then set Error-flag = 1.
Step 4. End of cycle:
    If Error-flag = 0, then set $T = T_c$; otherwise set $T$ = reference ALV location and print "all visual features are missing."
```

The purpose of including Steps 1(b) and 1(c) is to save computing time. If pixels belonging to cluster-2 in the surrounding area are insufficient, we conclude that no path line exists on the road and ignore line extraction, and perform the RMM algorithm immediately. On the other hand, if there are enough pixels in the area, line extraction and the LMM algorithm are performed to derive a candidate ALV location T_c. If T_c is not similar to the reference line-template, the LMM algorithm becomes unreliable and the RMM algorithm is then used. Finally, if the candidate ALV location resulting from the RMM algorithm is not similar to the reference road-template, the RMM algorithm becomes unreliable, too. At this moment, all of the desired visual features are missed and the ALV is guided blindly.

3. PROPOSED ALV LOCATION METHOD

3.1. Reference ALV Location Estimation

The ALV keeps moving forward after an image is taken at the beginning of each navigation cycle. When the image has been processed and the LMM or RMM algorithm has been performed, the actual ALV location at the time instant of image taking can be found. At this moment, however, the ALV has already travelled a distance S. Hence, the ALV never knows its actual current position unless the cycle time is zero. To overcome this difficulty, we use the following information to estimate the current ALV location according to a method proposed by Cheng and Tsai:[16]

1. The obtained actual location (d_i, θ_j) at the time instant of image taking.
2. The travelled distance S.
3. The pan angle δ of the front wheels relative to the y-axis of the VCS.
4. The distance d between the front wheels and the rear wheels.

Figure 7(a) illustrates the navigation process, where B_i and E_i are the beginning and the ending times of cycle i, respectively. At time B_i, an image is taken and the current ALV location is assumed to be P_i. At time E_i, P_i is found but the current ALV location P_{i+1} is unknown. At this moment, we use the control information given above to estimate the current ALV location called P_{i+1}. As shown in Figure 7(b), the vehicle is located at P_i. What we desire to know is the relative location of P_{i+1} with respect to P_i, denoted by a vector T. By the basic kinematics of the ALV, the rotation radius R can be found to be

$$R = d/\sin\delta. \tag{10}$$

And angle γ can be determined by

$$\gamma = S/R, \tag{11}$$

where S can be obtained from the counter of the system odometer. So, the length of vector T can be solved to be

$$l_T = R\sqrt{2(1-\cos\gamma)}, \tag{12}$$

and the direction of T is determined by the angle

$$\mu_T = \frac{\pi}{2} - \delta - \frac{\gamma}{2}. \tag{13}$$

Using the vector T, the VCS coordinates of location P'_{i+1} with respect to P_i can thus be computed by

$$x = l_T \cos\mu_T,$$
$$y = l_T \sin\mu_T. \tag{14}$$

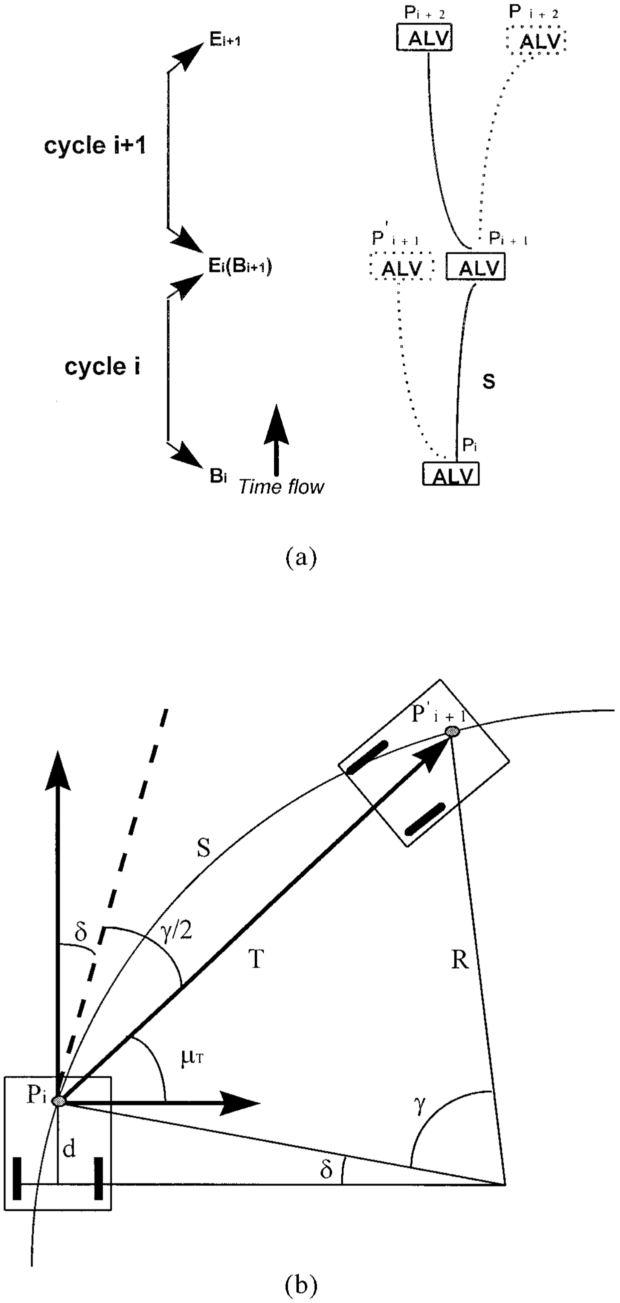


Figure 7. Illustration of reference ALV location estimation. (a) The navigation process in a cycle by cycle manner. (b) The vehicle location before and after the ALV moves a distance S forward.

So, in cycle i, P_i and P'_{i+1} can be determined at time E_i. And the estimated ALV locations at time B_{i+1} can be set to P'_{i+1}. We then define

$$\text{Reference ALV location of cycle } i = \text{the estimated ALV location at time } B_i. \tag{15}$$

Hence, P'_{i+1} is used as the reference ALV location of cycle $i+1$. The reference ALV location can be used for the following purposes:

1. Speeding up the algorithms of line checking and extraction.
2. Speeding up the LMM and the RMM algorithms.
3. Driving the ALV to the given path.

3.2. Path Planning

To guide the ALV to following a given path, we have to plan a smooth trajectory for the ALV from the current estimated ALV location to the given path in each navigation cycle. For this, a closeness distance measure from the ALV to the given path proposed by Cheng and Tsai[16] is employed, which is defined as

$$L_p(\delta) = \frac{1}{1 + [D_P^F(\delta)]^2 + [D_P^B(\delta)]^2} \tag{16}$$

where D_P^F and D_P^B are the corresponding distances from the front and the rear wheels of the ALV to the given path after the ALV traverses a distance S with the turn angle δ as shown in Figure 8. A larger value of L_p means that the ALV is closer to the path. It is easy to verify that $0 < L_P \le 1$, and that $L_P = 1$ if and only if both of 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 are searched. An angle is hypothesized each time, and the value of L_P is calculated accordingly. The angle that produces the maximal value of L_P is then used as the turn angle for safe navigation.

3.3. ALV Control Issues

There is always unavoidable mechanical inaccuracy within the ALV control system. If we only use the control information to drive the ALV, the ALV may be far away from the goal when the navigation ends. Hence, we employ computer vision techniques to extract stable visual features as auxiliary tools to avoid gradual accumulation of control error. Of course, control error still exists in each navigation cycle.

Next, 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 twist. On the other hand, a smaller range of turn angles may cause only a slight closeness 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 navigations that a turn from −5° to +5° is a good compromise.

Finally, we discuss the problem of ALV speed adjustment on varying road situations. When the ALV meets ascending or descending roads, its speed will become slower or faster, respectively. To keep the speed steady, we propose a solution as follows. As we guide the ALV in navigation 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, respectively. Then, the speed of the ALV in cycle i can be calculated by

$$V_i = S_i / T_i. \tag{17}$$

Assume that we want to drive the ALV at a constant speed V. Then, if $V_i > V$, it is decided that the ALV is on a descending road. One way to derive a new driving power P_{i+1} for the ALV is:

$$P_{i+1} = \left(\frac{V}{V_i}\right) P_i. \tag{18}$$

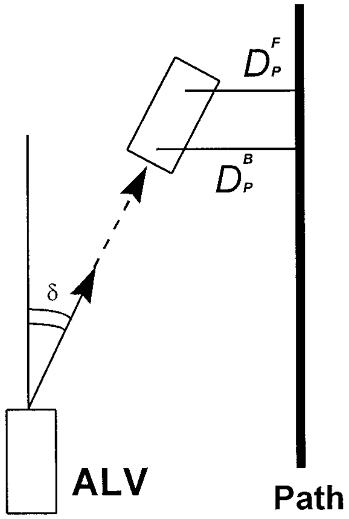


Figure 8. Illustration of closeness distance measure $L_P(\delta) = 1/\{1 + [D_P^F(\delta)]^2 + [D_P^B(\delta)]^2\}$.

Similarly, if $V_i < V$, the ALV is on an ascending road. And P_{i+1} can be calculated by

$$P_{i+1} = \left(\frac{V_i}{V}\right) P_i. \tag{19}$$

We then use P_{i+1} as the new driving power for cycle $i + 1$.

4. IMAGE PROCESSING TECHNIQUES

The general purpose of image processing is to extract useful features from input images. Some forms like region-based or edge-based descriptions can be selected for use as representations of such features. We use a clustering algorithm and a line extraction method as the main image processing techniques to find such descriptions in our study. Moreover, color is a powerful descriptor that often simplifies object extraction and identification from a scene, so we use color features in the clustering algorithm. To solve the problem caused by great changes of intensity, we also propose a technique to acquire better clustering results by choosing suitable initial cluster centers for the clustering algorithm. The proposed color information clustering algorithm and the line extraction technique are described as follows.

4.1. Color Information Clustering

The original image size is 512 × 486. To speed up our algorithms, we have to reduce the image size. The upper portion in the road image is discarded because it does not contain any road area, and pixels are sampled from the remaining image with the interval of five pixels in both the horizontal and vertical directions to form a reduced 103 × 45 image. We then use the ISODATA algorithm,[20] based on an initial-center-choosing (ICC) technique that can solve the problem caused by great changes of intensity in navigations, to divide the road image into three clusters. Because the clustering result is sensitive to the choice of initial centers, it is necessary to choose suitable initial centers as the algorithm begins to run. The ICC technique is described below.

Before ALV navigation, we run an unsupervised clustering algorithm that is guaranteed to converge after several iterations by choosing proper initial centers. The choice of appropriate initial centers is described as follows. We first observe the histogram of the R-plane, and divide all the pixels into six pieces of the same size as shown in Figure 9, i.e., for all $k = 0$, 1, and 2, we have

$$\sum_{s=r_k}^{r'_k} [\text{pixel no. of } g.l.(s)] = \sum_{t=r'_k}^{r_{k+1}} [\text{pixel no. of } g.l.(t)]. \tag{20}$$

where $g.l.$ means gray level. Then, r'_k is taken to be the R-component of the candidate center of cluster k. Using the same criterion on the G-plane and B-plane, we can find the G-component g'_k and B-component b'_k. We then use $[r'_k, g'_k, b'_k]$ as the initial center of cluster k to run the unsupervised clustering algorithm, where $k = 0, 1, 2$.

When navigation begins, it is not a good policy to run the unsupervised clustering algorithm again because it needs about 10 to 20 iterations, which take too much computing time, to finish its convergence and is unsuitable for real-time ALV navigation. Hence, we simply run the clustering algorithm for only three iterations, which is enough to gain ideal clusters if we can choose proper initial centers. Intuitively, we can select the resulting centers in the previous navigation cycle as the initial centers in the current navigation cycle to run the supervised clustering algorithm.

The selection may be unsuitable, producing unacceptable clusters, because some difference may exist between two continuous images. One kind of the difference comes from the change of intensity. If the change of intensity between two continuous images is great (resulting from clouds covering the sun unexpectedly, for example), the candidate initial centers chosen from the resulting centers in the previous cycle may be far away from their real centers. In this situation, three iterations are not enough for the ISODATA algorithm to move the candidate initial centers close to their real centers, and erroneous clusters may be produced.

Furthermore, the number of clusters may decrease and an even more serious problem happens. Figure 10 illustrates this situation. Figure 10(a) shows the distribution of the three clusters and their resulting centers (represented by the big black dots) in the previous cycle. Figure 10(b) shows the greatly-translated distribution of the pixels caused by great changes of intensity in the present cycle, where black big dots represent the candidate initial centers inherited from the resulting centers in the previous cycle. After we run the supervised clustering algorithm for three iterations in the present cycle, we obtain the clustering result shown in Figure 10(c), which is erro-

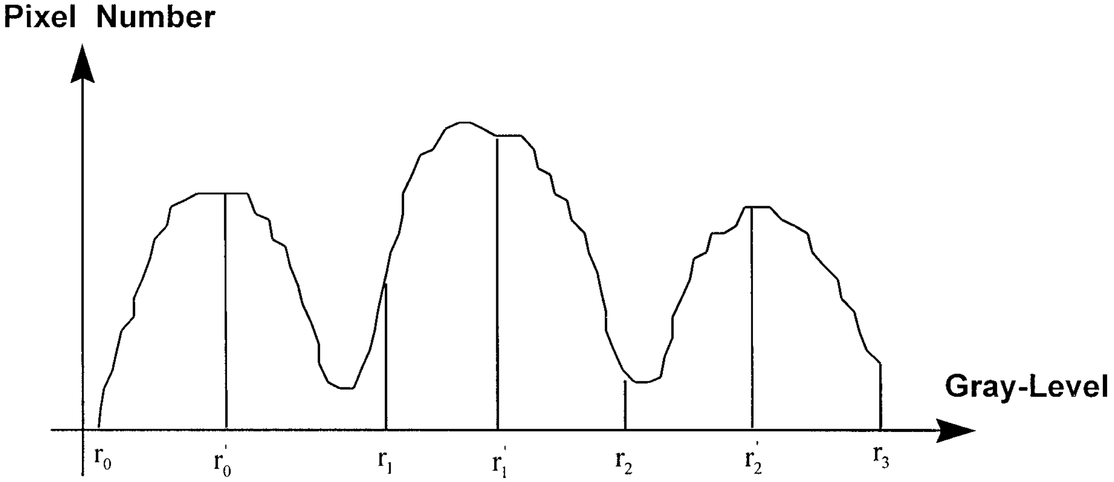


Figure 9. The histogram of the R-plane where r_k' is the R-component of the center of cluster k.

neous because one cluster becomes null and only two clusters contain samples.

To solve this problem, we perform a translation on the resulting centers in the previous cycle to form more representative points that are then selected for use as the actual candidate initial centers in the present cycle. The translation is described as follows. Assume that R'_{ave}, G'_{ave}, and B'_{ave} are the averages of the R-components, the G-components, and the B-components of all the pixels in the previous input image, respectively, and R_{ave}, G_{ave}, and B_{ave} are the averages of the R-components, the G-components, and the B-components of all the pixels in the present input image, respectively. A translation vector is defined by

$$[D_r, D_g, D_b] = [R_{ave} - R'_{ave}, G_{ave} - G'_{ave}, B_{ave} - B'_{ave}]. \quad (21)$$

If the resulting center of cluster k in the previous cycle is $[R'_{ki}, G'_{ki}, B'_{ki}]$, $k = 0, 1, 2$, then, the translated candidate initial center of cluster k is taken to be

$$[R_{ki}, G_{ki}, B_{ki}] = [R'_{ki}, G'_{ki}, B'_{ki}] + [D_r, D_g, D_b]. \quad (22)$$

Figure 10(d) shows the translation process, where the black big dots represent the new translated initial centers. We then use the new translated centers as the new candidate initial centers to run the supervised clustering algorithm, and a much better clustering result is shown in Figure 10(e).

An example of obvious improvement obtained from using the above initial cluster center translation technique in a real road scene is shown in Figure 11. Figure 11(a) shows the input image with high intensity in the previous cycle and Figure 11(b) shows the input image with low intensity in the present cycle. Figure 11(c) shows the poor clustering result when the above translation process is not employed. One cluster becomes null in the result. A nice clustering result obtained from the translation process is shown in Figure 11(d), in which we see that cluster-1 is the road area and cluster-2 the path lines. This observation again justifies our use of a combined line and road following technique.

4.2. Path Line Extraction Using Hough Transform

Generally, two effective methods can be used for line extraction. One is the Least-Square (LS) line approximation[24,25] and the other is the Hough transform.[23] The Hough transform can extract lines in any direction automatically even though they are dotted, whereas the LS line approximation has to select proper pixels before approximating lines and hence complicate coding and debugging. But if many candidate pixels in the XY plane are sent to the Hough counting space, it will take too much computing time. Hence, time is the vulnerability of the Hough transform. From an observation of Figure 11, we find in the binary image that cluster-2 pixels are few and discontinuity exists between pixels, so the Hough transform is a proper line extraction method.

Before line extraction, the surrounding area of the three lines of the reference line-template is checked. As described previously, if pixels belonging

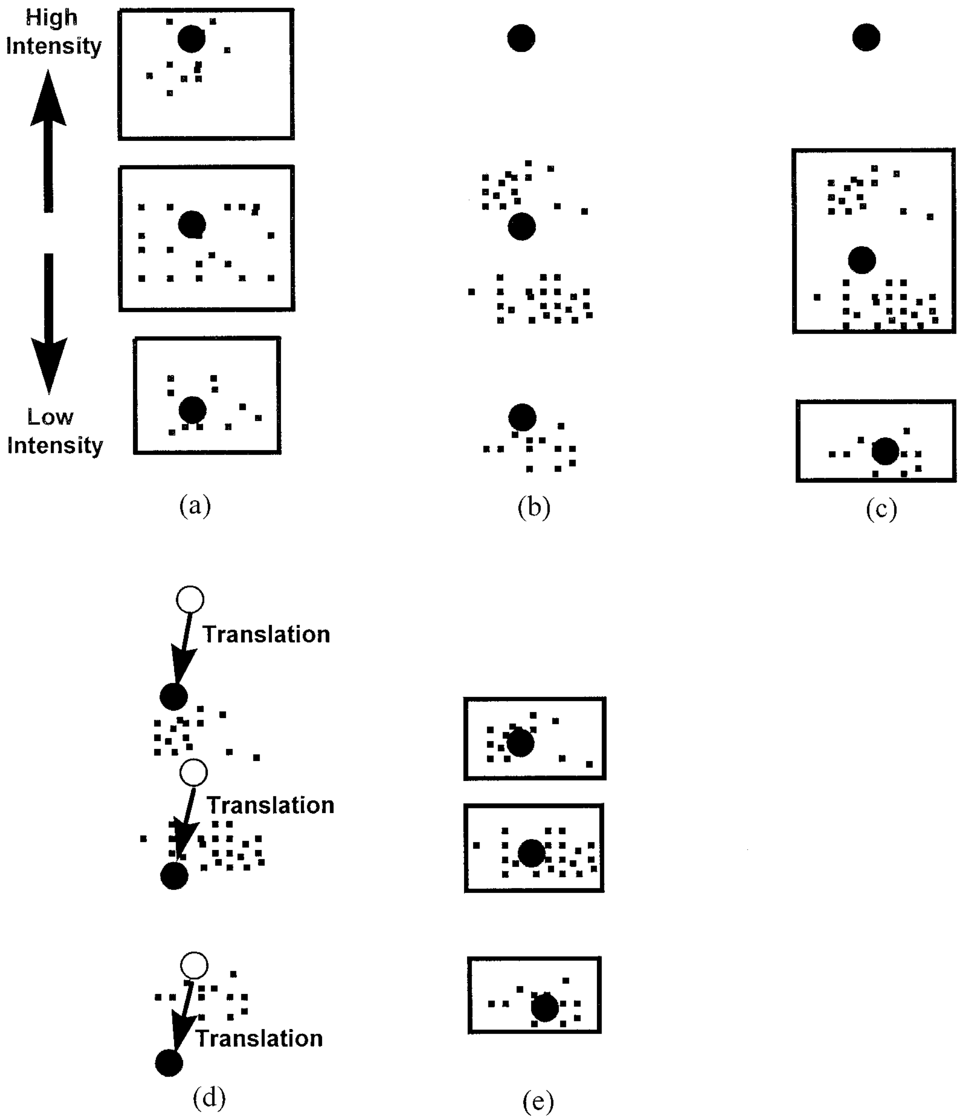


Figure 10. The influence of great changes of intensity and its solution. (a) The distribution of the three clusters and their converged centers in the previous cycle. (b) A greatly-translated distribution of the pixels caused by great change of intensity in the present cycle, where black big dots represent the candidate initial centers inherited from the resulting centers in the previous cycle. (c) The distribution of the three clusters after running the supervised clustering algorithm for three iterations in the present cycle, with one cluster becoming null, causing a serious problem. (d) The translation process, where new translated initial centers are selected according to Eqs. (21) and (22). (e) A much better clustering result with three ideal clusters is obtained after using the new translated initial centers to run the clustering algorithm.

to cluster-2 in the surrounding area are insufficient, we conclude that no path line exists on the road and line extraction is ignored. If there are enough pixels in the area, we decide that there are path lines on the road and send the candidate pixels to the Hough counting space to extract path lines. Normally, three high peaks are found in the counting space because there are three path lines on the road. But because of some unsteady road conditions such as shadows, degraded regions, or cars on the roadsides, we may only find one or two high peaks corresponding to one or two lines with which we still can locate the ALV by using the majority-vote rule described previously.

5. EXPERIMENTAL RESULTS

The prototype ALV constructed in this study was used for testing the proposed approach, and could navigate smoothly along part of the campus road in National Chiao Tung University for about 150 m. Besides, it is not sensitive to sudden changes of intensity because of the effective ICC technique used in the clustering algorithm. The width of the road is 6.8 m, and there are three yellow path lines on the road with the central line dotted. The average cycle time is about 2.1 s, and the average speed is 90 cm/s or 3.3 km/h, which is reasonable for the small-scaled and PC-based ALV. The ALV can navigate steadily in spite of the fact that there were shades, vehicles, people, or degraded regions on the roadsides.

Figure 12 shows some images and their clustering results in several complex road environments. Figure 12(a) shows a road image that includes some small shadows caused by trees on the left roadside and some degraded regions on the left lane. All of the three path lines (represented by the three white lines) in the image were successfully extracted, which were then used as the visual features. Figure 12(b) shows an image including some large shadows caused by trees on the right roadside and one degraded region on the left lane. Only the left and the middle path lines were extracted (represented by the two white lines). A moving car and its shadow on the left lane and one strip of degraded region on the right lane are seen in Figure 12(c), from which no path lines were extracted and the road surface was used as the visual feature, and the best-matched road-template resulting from the RMM process is represented by the two white lines. In all of the three cases, our ALV moved forward successfully.

6. CONCLUSIONS

A model-based approach to ALV guidance in outdoor road environments by computer vision has been proposed. Several techniques have been integrated in this

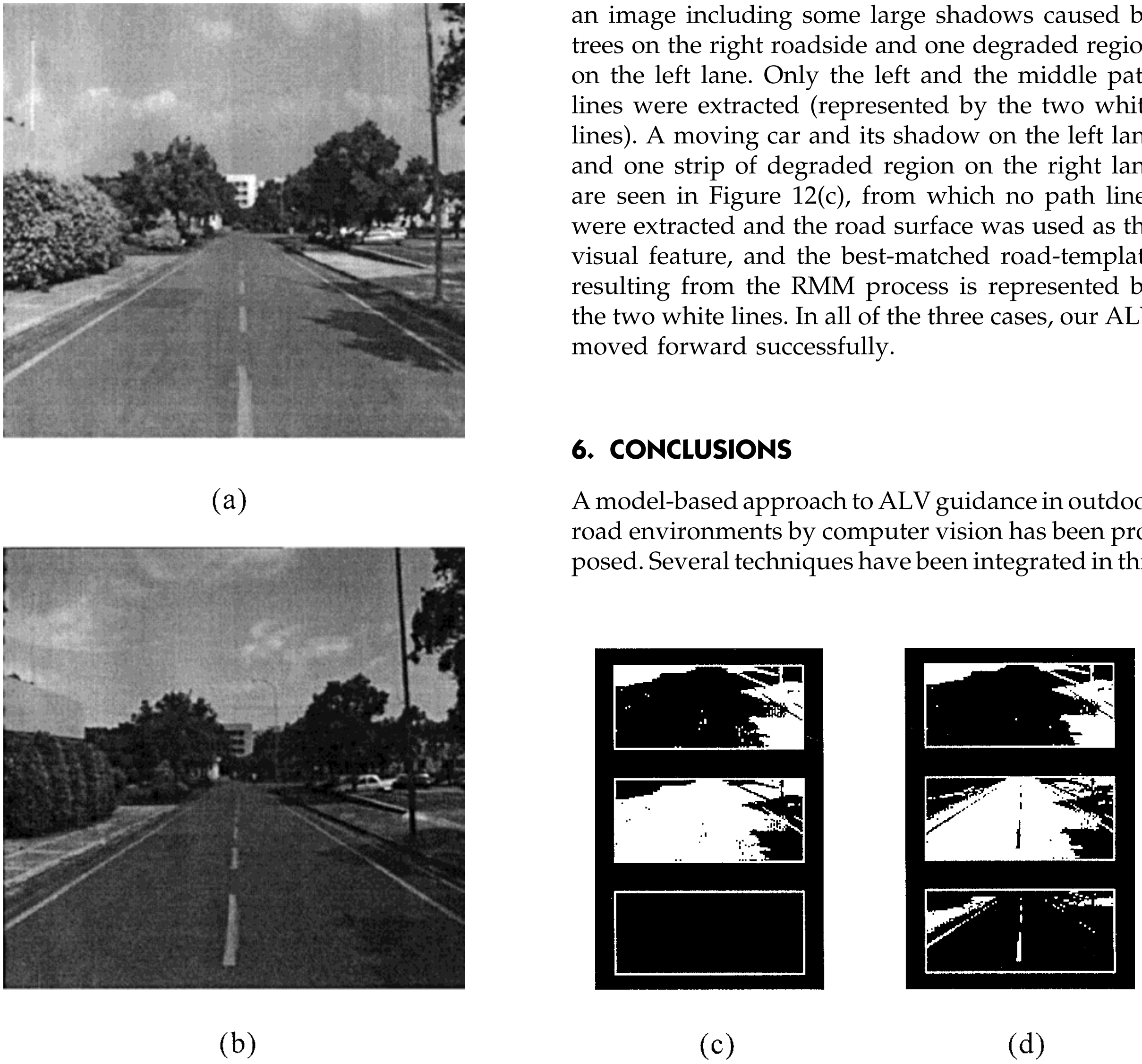
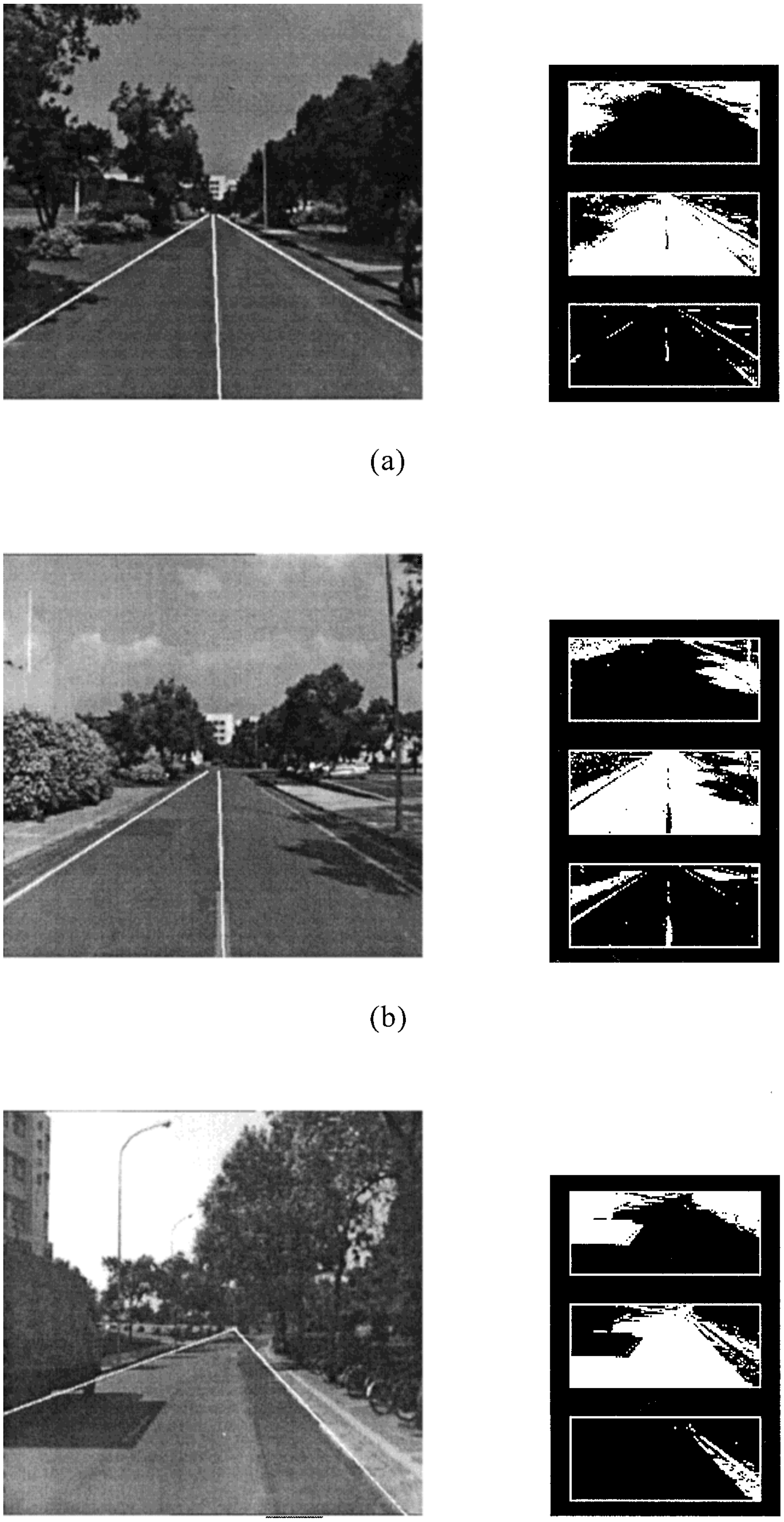


Figure 11. An obvious improvement to use the translation process in a real road scene. (a) The input image with high intensity in the previous cycle. (b) The input image with low intensity in the present cycle. (c) Poor clustering result including one null cluster when we ignore the translation process. (d) Nice clustering result when we adopt the translation process.



(a)

(b)

(c)

Figure 12. Several representative images, their clustering results, and the used visual features in complex road environments.

study to provide a combined line and road following scheme for ALV navigation. The CLRMM algorithm has been used to achieve faster and more flexible navigations in real time. The ISODATA clustering algorithm based on the ICC technique has been employed to solve the problem caused by great changes of intensity in navigations. The process of the reference ALV location estimation by basic vehicle kinematics has been implemented to save computing time when the CLRMM algorithm is run. Smooth, safe, and steady-speed navigation has been achieved by the use of a reasonable path planning method and some realistic vehicle control schemes. Successful navigation tests in general roads confirm the effectiveness of the proposed approach.

One factor influencing our model-based approach is the unflatness of some road segments, which causes larger twists in our tests. Hence, how to overcome road unflatness is a good topic for future study. In addition, the line-model and road-model are created based on the assumption that the path lines and the road boundaries are straight and parallel to each other. When the ALV meets curved roads, it cannot navigate regularly because it violates the assumption. Guidance on the sharp-curved road based on the model matching technique is another good topic for future study. Additionally, problems caused by changes of sunshine directions, selections of different features on the road, and seeking effective algorithms to solve the encountered problems are also interesting directions of future studies.

This work was supported by National Science Council, Republic of China under Grant NSC-83-0404-E009-036.

REFERENCES

1. Y. Goto and A. Stentz, ''The CMU system for mobile robot navigation,'' *Proc. IEEE Int. Conf. Robotics Automat.*, Raleigh, NC, 1987, pp. 99–105.
2. C. Thorpe, M. H. Hebert, T. Kanade, and S. A. Shafer, ''Vision and navigation for Carnegie-Mellon NAVLAB,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **10**(3), 362–373, 1988.
3. K. Kluge and C. Thorpe, ''Explicit Models for Robot Road Following,'' *Proc. IEEE Int. Conf. Robotics Automat.*, Scottsdale, AZ, 1989, pp. 1148–1154.
4. D. Pomerleau, ''Neural network based autonomous navigation,'' *Vision and Navigation: The Carnegie Mellon Navlab*, C. Thorpe, Ed. Kluwer, Norwell, MA, 83–92, 1990.
5. J. D. Crisman and C. E. Thorpe, ''UNSCARF, A Color Vision System for the Detection of Unstructured Roads,'' *Proc. IEEE Int. Conf. Robotics Automat.*, Sacramento, CA, 1991, pp. 2496–2501.
6. C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, ''Toward Autonomous Driving: The CMU Navlab, Part I—Perception,'' *IEEE Expert*, **6**(3), 31–42, 1991.
7. C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, ''Toward Autonomous Driving: The CMU Navlab, Part II—Architecture and Systems,'' *IEEE Expert*, **6**(3), 44–52, 1991.
8. J. D. Crisman and C. E. Thorpe, ''SCARF: A Color Vision System that Tracks Roads and Intersections,'' *IEEE Trans. Robotics Automat.*, **9**(1), 49–58, 1993.
9. M. A. Turk, D. G. Morgenthaler, K. D. Germban, and M. Marra, ''VITS—a vision system for autonomous land vehicle navigation,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **10**(3), 342–361, 1988.
10. X. Lin and S. Chen, ''Color Image Segmentation Using Modified HSI System for Road Following,'' *Proc. IEEE Int. Conf. Robotics Automat.*, Sacramento, CA 1991, pp. 1998–2003.
11. E. D. Dickmans and A. Zapp, ''A curvature-based scheme for improving road vehicle guidance by computer vision,'' *Proc. SPIE Mobile Robot Conf.*, Cambridge, MA, 1986, pp. 161–168.
12. K. Kuhnert, ''A vision system for real time road and object recognition for vehicle guidance,'' *Proc. SPIE Mobile Robot Conf.*, Cambridge, MA, 1986, pp. 267–272.
13. B. Mysliwetz and E. D. Dickmanns, ''A vision system with active gaze control for real-time interpretation of well structures dynamic scenes,'' *Proc. Intelligent Autonomous Systems*, Amsterdam, The Netherlands, 1986.
14. D. Kuan, G. Phipps, and A. Hsueh, ''Autonomous Robotic Vehicle Roading Following,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **10**(4), 648–658, 1988.
15. K. E. Olin and D. Y. Tseng, ''Autonomous Cross-Country Navigation: An Integrated Perception and Planning System,'' *IEEE Expert*, **6**(4), 16–30, 1991.
16. S. D. Cheng and W. H. Tasi, ''Model-based guidance of autonomous land vehicle in indoor environments by structured light using vertical line information,'' *J. Electrical Engineering*, **34**(6), 441–452, 1991.
17. P. S. Lee, Y. E. Shen, and L. L. Wang, ''Model-based location of automated guided vehicles in the navigation sessions by 3D computer vision,'' *J. Robotic Systems*, **11**(3), 181–195, 1994.
18. L. L. Wang, P. Y. Ku, and W. H. Tsai, ''Model-based guidance by the longest common subsequence algorithm for indoor autonomous vehicle navigation using computer vision,'' *Automation in Construction*, **2,** 123–137, 1993.
19. Y. M. Su and W. H. Tsai, ''Autonomous land vehicle guidance for navigation in buildings by computer vision, radio, and photoelectric sensing techniques,'' *J. Chinese Institute of Engineers*, **17**(1), 63–73, 1994.
20. R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, Inc., New York, 1973.
21. D. Ballard and C. Brown, *Computer Vision*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982.
22. K. Castleman, *Digit Image Processing*, A. Oppenheim, Ed., Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
23. R. C. Gonzalez and Richard E. Wood, *Digit Image Pro-*

cessing, Addison-Wesley Publishing Company, Inc., Reading, MA, 1992.

24. J. H. Mathews, *Numerical Methods for Mathematics, Science, and Engineering,* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1992.
25. L. L. Wang and W. H. Tsai, ''Safe highway driving aided by 3-D image analysis techniques,'' *Proc. of 1986 Microelectronics and Information Science and Technology (MIST) Workshop,* Hsinchu, Taiwan, R. O. C., 1986, pp. 671–686.
26. J. D. Foley and A. V. Dam, *Fundamentals of Interactive Computer Graphics,* Addison-Wesley Publishing Company, Inc., Reading, MA, 1982.
27. E. Horowitz and S. Sahni, *Fundamentals of Data structures in Pascal,* Computer Science Press, An Imprint of W. H. Freeman and Company, New York, 1990.