Extracting characters from real vehicle licence plates out-of-doors

B.-F. Wu, S.-P. Lin and C.-C. Chiu

Abstract: Most applications of automatic vehicle identification systems are outdoors and are consumer-orientated. The recognition rate, the system reliability and the processing speed are very important. A system for extracting characters from licence plates has been presented. The system can successfully extract characters from licence plates in any part of a captured image without the need to consider the luminance of the surroundings or the size and inclination of the licence plate or the colour of the vehicle. Moreover, the processing time of this system is satisfactory for several applications. The proposed system depends only on the completeness of a character and so is tolerant of skewed plates and complex backgrounds. In a real parking lot, an extraction rate of 95.6% was obtained by applying the system to 228 vehicles.

1 Introduction

Automatic vehicle identification (AVI) systems have many applications, such as traffic control [1], traffic analysis [2–4], parking automation [5] and electrical tollgate management. The common aim of these applications is to reduce manpower and facilitate to the automatic management. An AVI system must thus exhibit a high recognition rate and processing speed. For example, drivers normally have little patience when waiting for their vehicle to be recognised by a car parking system. In some outdoor applications, changes in the weather directly influence the recognition rate of the AVI system. Accordingly, reliability is another requirement of an AVI system. A higher recognition rate or reliability generally corresponds to slow processing by the AVI system. Accordingly, the system must trade-off the processing speed for recognition rate or reliability.

In earlier investigations, an AVI system typically had four stages – the car image capturing stage, the character extraction stage, the character recognition stage and the control dispatching stage.

First, the image of an incoming car is captured by infrared sensors, which can easily detect the passing of a car. In some investigations [6, 7], the change in intensity between two or more consecutive frames when a car is in sight of a camera is used as a substitute for the data obtained using an infrared sensor. After the car is detected, a camera captures an image of it and an analogue-to-digital (A/D) converter is applied to change the analogue image signal into a digital one. Then, in the character extraction stage, the vehicle licence plate is located and the region that contains the characters is extracted.

The locations of a vehicle licence plate and the characters in the plate are identified using edge detection [8–10], feature projection [11–13], neural networks (NN) [14–16], fuzzy logic [17, 18], genetic algorithms [19] or morphology-based [20] technologies. After the characters have been extracted, the character recognition stage matches the features of the extracted character components to template patterns in a database and determines the template pattern in the database that is most similar to the input pattern. A topologically invariant recognition method [21] and a syntactic recognition method [22] are based on different features. Finally, the control dispatching stage uses the result from the preceding stage and responds accordingly, by allowing or prohibiting entrance.

Recently, more studies have emphasised outdoor application [23], robustness and real-time operation. Recognition rate, reliability and processing speed are three important issues associated with AVI systems. The character extraction stage is the most time-consuming and performance-critical stage in an AVI system. Accordingly, this work proposes a character extraction system, which includes three phases – pre-processing, labelling modified connected components and determining components of characters.

2 System overview

For reasons of practicality, the character extraction system was implemented with three units – a software unit, a firmware unit and a hardware unit. Fig. 1*a* depicts the components of each unit and Fig. 1*b* shows the practical installment of the AVI system. The most important part of the software unit in an AVI system is the kernel component, which incorporates the character extraction stage and the character recognition stage. The following section details each process in the character extraction stage, including pre-processing, labelling modified connected components and determining the components of characters.

3 Character extraction system

3.1 Pre-processing

Tasks that must be performed in pre-processing include eliminating noise, deleting interlacing, transferring the

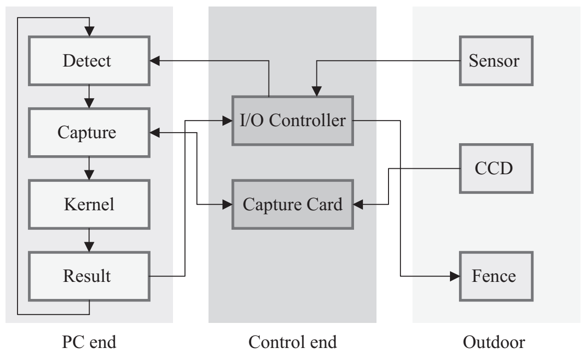
doi:10.1049/iet-cvi:20050132

Paper first received 10th May 2005 and in final revised form 14th November 2006

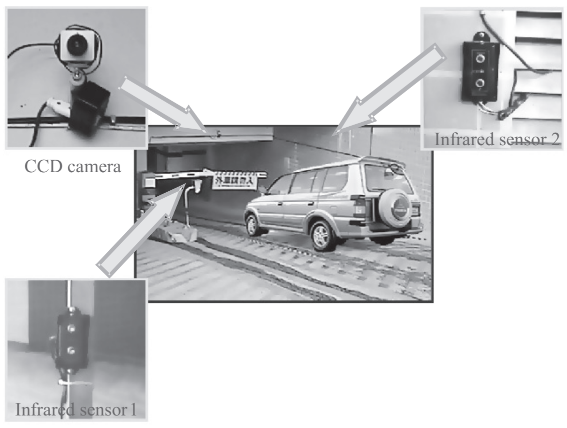
B.-F. Wu and S.-P. Lin are with the Department of Electrical and Control Engineering, National Chiao Tung University, Hsinchu 300, Taiwan, Republic of China

C.-C. Chiu is with the Department of Electrical Engineering, Chung Cheng Institute of Technology, National Defense University, Taoyuan 335, Taiwan, Republic of China

E-mail: chiu@ccit.edu.itw



a



b

Fig. 1 *Implementation of an AVI system*

a Components of each unit

b Practical installation of proposed system

colour space, masking unnecessary background and segmenting the foreground. The factors that influence the images captured by different types of camera are, for example, the frame resolution, the scanning method, the frame rate and the depth of colour. In this work, a low-pass filter eliminates the noise. De-interlacing and colour space transformation use the line-interpolation method and RGB-to-YUV transformation, respectively. The proposed system uses only the Y-component (greyscale image) to extract the characters.

3.1.1 Masking unnecessary background: In order to tolerant of complex background, the system masks unnecessary background for the captured image. Deleting the background of the captured image is more difficult than calculating the difference between the captured image and a stored background image. In fact, the task is not trivial for two reasons. First, changing weather causes the luminance of the background to vary in time. Second, the automatic exposure control of the camera dynamically changes the exposure sensitivity if the surroundings are too light (e.g. when the captured car is white) or too dark (e.g. when it is black). Hence, the change of the exposure sensitivity varies the luminance of the background of the captured image. This work applies a local histogram equalisation method with horizontal–vertical refinement to solve these problems.

Firstly, the captured image and the background image are partitioned into block-images. The size of the image is assumed to be $W \times H$ and the size of the block-image is assumed to be $M \times N$. In addition, the value W and H are divisible by M and N, respectively. Then, the (i, j)th block-image, in the ith row and the jth column of all partitioned block-images, is represented as $f_{i,j}(x, y)$, where $x \in [0, M-1]$, $y \in [0, N-1]$, $j \in [0, \lfloor W/M \rfloor - 1]$ and $j \in [0, \lfloor H/N \rfloor - 1]$, and f denotes the captured image $f^{\mathrm{C}}_{i,j}(x, y)$ or the background image $f^{\mathrm{B}}_{i,j}(x, y)$.

Secondly, the histogram $h_{i,j}(r)$ of $f_{i,j}(x, y)$ is given by the following equality.

$$h_{i,j}(r) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\begin{cases}1 & \text{if } f_{i,j}(x,y) = r\\ 0 & \text{otherwise}\end{cases} \tag{1}$$

where r 0–255.

Then, the difference $D_{i,j}$ between $f^{\mathrm{C}}_{i,j}(x, y)$ and $f^{\mathrm{B}}_{i,j}(x, y)$ is defined as

$$D_{i,j} = \frac{1}{MN} \times \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left|E_{\mathrm{h}}(f^{\mathrm{C}}_{i,j}(x,y)) - E_{\mathrm{h}}(f^{\mathrm{B}}_{i,j}(x,y))\right| \tag{2}$$

where E_{h} is the histogram equalisation function, which is

$$E_{\mathrm{h}}(f_{i,j}(x,y)) = 255 \times \sum_{r=0}^{f_{i,j}(x,y)} h_{i,j}(r) \tag{3}$$

Next, the pixel values of block-images with $D_{i,j} < T$ are set to 255. Thus, a new block-image $\hat{f}_{i,j}(x, y)$ is obtained that may be either of P-type (preserved block) or of C-type (cleared block), according to the following formula

$$\hat{f}_{i,j}(x,y) = \begin{cases}255 & \text{if } D_{i,j} < T \text{ (C-type block)}\\ f^{\mathrm{C}}_{i,j}(x,y) & \text{otherwise (P-type block)}\end{cases} \tag{4}$$

Finally, horizontal–vertical refinement is used to delete horizontal and vertical isolating block-images. The vertical or horizontal sequences, $\mathrm{VS}(n_{\mathrm{v}})$ or $\mathrm{HS}(n_{\mathrm{h}})$, respectively, are defined as the vertical or horizontal contiguous P-type blocks of $\hat{f}_{i,j}(x, y)$, which are bounded by C-type blocks, respectively. The parameters n_{v} and n_{h} depend on the numbers of vertical and horizontal sequences, respectively.

Notably, all blocks in $\mathrm{VS}(n_{\mathrm{v}})$ and $\mathrm{HS}(n_{\mathrm{h}})$ include only P-type blocks and the length of each $\mathrm{VS}(n_{\mathrm{v}})$ or $\mathrm{HS}(n_{\mathrm{h}})$ is $\mathrm{lv}(n_{\mathrm{v}})$ or $\mathrm{lh}(n_{\mathrm{h}})$. Then thresholds T_{v} and T_{h} can be applied to detect the unnecessary background in the following manner

$$\mathrm{VS}(n_{\mathrm{v}}) = \begin{cases}\text{Delete the } \mathrm{VS}(n_{\mathrm{v}}) & \text{if } \mathrm{lv}(n_{\mathrm{v}}) < T_{\mathrm{v}}\\ \text{Reserve the } \mathrm{VS}(n_{\mathrm{v}}) & \text{otherwise}\end{cases} \tag{5}$$

$$\mathrm{HS}(n_{\mathrm{h}}) = \begin{cases}\text{Delete the } \mathrm{HS}(n_{\mathrm{h}}) & \text{if } \mathrm{lh}(n_{\mathrm{h}}) < T_{\mathrm{h}}\\ \text{Reserve the } \mathrm{HS}(n_{\mathrm{h}}) & \text{otherwise}\end{cases} \tag{6}$$

3.1.2 Segmentation: A thresholding method, the modified adaptive thresholding (MAT), is proposed for the outdoor environment. In MAT, a threshold selector based on boundary characteristics is used dynamically to select one of the two well-known thresholding methods – basic adaptive thresholding (BAT) [24] and the c-means algorithm [25]. Although some thresholding methods already use boundary characteristics extracted by gradient or Laplacian operators, they were not very robust when applied to noisy, complex or low-resolution images. Accordingly, the boundary characteristics are not directly used herein for segmentation. Boundary characteristics are rather used to choose between either BAT or c-means to segment the block-images.

Optimal adaptive thresholding (OAT) is equivalent to single-variable multiple-prototype Bayes classification. The histogram $h(r)$ with grey-level r can be modelled as a combination of two single-variable Gaussian distributions with a prior P_0 and P_1 ($P_0 + P_1 = 1$), means r_0 and r_1 ($r_0 < r_1$) and variances σ_0 and σ_1

$$h(r) = \frac{P_0}{\sqrt{2\pi}\sigma_0}\exp\left\{-\frac{(r-r_0)^2}{2\sigma_0^2}\right\} + \frac{P_1}{\sqrt{2\pi}\sigma_1}\exp\left\{-\frac{(r-r_1)^2}{2\sigma_1^2}\right\} \quad (7)$$

On the basis of the above model, an optimal threshold can be calculated by minimising an error function $E_{0,1}$, given by

$$E_{0,1} = \int_{T_{0,1}}^{\infty} \frac{P_0}{\sqrt{2\pi}\sigma_0}\exp\left\{-\frac{(r-r_0)^2}{2\sigma_0^2}\right\} \mathrm{d}r + \int_{-\infty}^{T_{0,1}} \frac{P_1}{\sqrt{2\pi}\sigma_1}\exp\left\{-\frac{(r-r_1)^2}{2\sigma_1^2}\right\} \mathrm{d}r \quad (8)$$

The decision function, which separates the two Gaussian distributions in (8), is quadratic. However, if the variances of the two distributions are equal, such that $\sigma_0 = \sigma_1 = \sigma$, then the decision-making threshold will be $T_{0,1} = ((r_0 + r_1)/2) + \sigma^2/(r_0 - r_1)\ \ln(P_1/P_0)$. Furthermore, if the probabilities are also the same, such that $P_0 = P_1$, then the threshold will become $T_{0,1} = ((r_0 + r_1)/2)$, which is the mean value of $h(r)$. The result is the same as that obtained using BAT.

The BAT is appropriate for segmenting two clusters whose variances and probabilities are equal and the computational complexity of BAT is lower than that of OAT. The OAT is time-consuming, so the c-means algorithm is used here. The proposed thresholding approach, MAT, combines the BAT and c-means algorithm to segment images. The number of iterations of the c-means algorithm is set to 2 to shorten the processing time. The algorithm is described in the following paragraph.

Firstly, an image $f(x, y)$ is separated into block-images. The grey-level of each pixel in the (i, j)th block-image is denoted as $f_{i,j}(x, y)$. Each block-image is of size $M \times N$. $A_{i,j}$ is defined as the set of boundary pixels in $f_{i,j}(x, y)$

$$A_{i,j} = \{(x, y) \mid |f_{i,j}(x, y) - f_{i,j}(x-1, y)| \geq T_\mathrm{d}\} \quad (9)$$

The T_d is a pre-defined threshold value. After the set of boundary pixels is obtained, the updated threshold T_u is defined as

$$T_\mathrm{u} = \frac{\sum_{f_{i,j}(x,y)\in A_{i,j}} f_{i,j}(x, y)}{\sum_{f_{i,j}(x,y)\in A_{i,j}} 1} \quad (10)$$

and the updated probability, corresponding to the (i, j)th updated threshold, is

$$P_{u,0} = \frac{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}\begin{cases}1 & \text{if } f_{i,j}(x,y) < T_\mathrm{u}\\ 0 & \text{otherwise}\end{cases}}{MN} \quad (11)$$

where $P_{\mathrm{u},1} = 1 - P_{\mathrm{u},0}$.

Finally, the operator processed by MAT is given by $M_{i,j}$ in the following equation

$$M_{i,j} = \begin{cases}\text{BAT} & \text{if } |P_{\mathrm{u},1} - P_{\mathrm{u},0}| < T_\mathrm{p}\\ \text{c-means algorithm} & \text{otherwise}\end{cases} \quad (12)$$

Fig. 2*a* shows a licence plate captured by the proposed system. Figs. 2*b–d* present the resulting images after processing by BAT, OAT and MAT, respectively. The image obtained using MAT is better than those obtained using BAT and OAT. Because the histograms of some blocks do not satisfy the assumption of the distribution model shown in (7), the OAT is not suitable for these blocks. This study used $T_\mathrm{d} = 8$ and $T_\mathrm{p} = 0.1$.

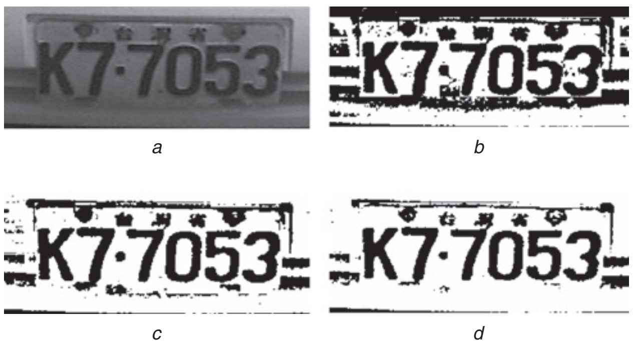


Fig. 2 *Licence plate processed by BAT, OAT and MAT*
a Original image (300 × 160 pixels)
b Processed by BAT
c Processed by OAT
d Processed by MAT

3.2 Labelling modified connected components

Several algorithms have been presented for labelling connected components. They can be grouped as sequential ones and parallel ones. In this work, only sequential algorithms are considered because the proposed AVI system architecture is based on a PC, which can only perform operations sequentially. Two popular sequential algorithms are the quadtree algorithm [26] and the raster scanning algorithm [27–29]

This work presents a modified connected component-labelling algorithm. The data structure of the algorithm is called equivalent linkage. The advantage of the data structure is that it requires less or no more memory for storage than is required in the previous works. The algorithm explores the region adjacent to each pixel only once. The merge and lookup operations are more efficient than others. The following paragraph details the proposed algorithms.

The modified connected component-labelling method involves two passes. In the first, $f(x, y)$ is assumed to represent the image of size $W \times H$ segmented after preprocessing, where $g(x, y) \in \boldsymbol{Z}$ is a label image of $f(x, y)$, and $L \in \boldsymbol{N}$ is a label counter, which stores the current minimum illegal label value obtained by the raster scanning from top to bottom and then from left to right; $E_\mathrm{q}(l) \in \boldsymbol{Z}$ represents the equivalence linkage of a label $l < L$; (x_0, y_0) are the coordinates of the pixel encountered during the raster scanning and $N(x_0, y_0)$ denotes the coordinates of the four previously scanned pixels, based on the coordinates of the central pixel (x_0, y_0)

$$N(x_0, y_0) = \{(x_0 - 1, y_0), (x_0 - 1, y_0 - 1), (x_0, y_0 - 1), (x_0 + 1, y_0 - 1)\} \quad (13)$$

Initially, L is set to 1 to indicate that no label has yet been created. During raster scanning, whenever the encountered pixel is in the background ($f(x_0, y_0) = 255$), the corresponding coordinates of the label of the image are set to 0 ($g(x_0, y_0) = 0$). However, if the encountered pixel is in the foreground and disjoint the previous foreground pixels, a new component is generated by assigning the value of the coordinates of the label image to the label counter ($g(x_0, y_0) = L$) and then increasing the label counter by one ($L = L + 1$). The equivalent linkage of the label, thus, generated is initialized to 0 ($E_\mathrm{q}(g(x_0, y_0)) = 0$; where 0 denotes unlabelled). Otherwise, if the encountered pixel is in the foreground and is joined to previously scanned foreground pixels, then the coordinates of the corresponding label image are set to the minimum equivalent linkage roots of the image's neighbours ($g(x_0, y_0) = \min_{(x,y)\in N(x_0,y_0)} E_Q(g(x, y))$; $E_Q(l)$ finds the root of the equivalent linkage for a specified label l.

$$g(x_0,y_0)=\begin{cases}0 & \text{if } f(x_0,y_0)=255\\ L,(L=L+1) & \\ \quad\text{else if } \forall\{(x,y)\in N(x_0,y_0)\}g(x,y)=0 & \\ \min_{(x,y)\in N(x_0,y_0)} E_Q(g(x,y)) & \text{otherwise}\end{cases} \tag{14}$$

$$E_Q(l)=\begin{cases}E_Q(E_q(l)) & \text{if } E_q(l)\neq 0\\ l & \text{otherwise}\end{cases} \tag{15}$$

Additionally, the equivalent linkages of the neighburs of encountered pixel in the label image must be updated, using the following equation

$$\begin{cases}\text{no operation} & \text{if } f(x_0,y_0)=255\\ E_q(g(x_0,y_0))=0 & \\ \quad\text{else if } \forall\{(x,y)\in N(x_0,y_0)\}\ g(x,y)=0 & \\ \text{update equivalent linkages} & \text{otherwise}\end{cases} \tag{16}$$

where the pseudo code of update equivalence linkages is shown below:

```
for each (x, y) in $N(x_0, y_0)$
{
    $E = E_q(g(x, y))$;
    if ($E_Q(E) \neq g(x_0, y_0)$)
    {
        do
        {   $E' = E_q(E)$;
            $E_q(E) = g(x_0, y_0)$;
            $E = E'$;
        } while ($E$ is unequal to 0);
    }
}
```

The first pass resolves all equivalent linkages but not the label image. Consequently, in the second pass, labels must be reassigned to resolve the label values stored in the label image. Equation (17) describes the label reassignment for each pixel with coordinates (x_0, y_0)

$$g(x_0,y_0)=\begin{cases}\text{no operation} & \text{if } g(x_0,y_0)=0\\ & \text{or } E_q(g(x_0,y_0))=0\\ E_Q(g(x_0,y_0)) & \text{otherwise}\end{cases} \tag{17}$$

After the second pass, all connected components are labelled. The primary attributes of a connected component are the coordinates and area. The extended attributes are the aspect ratio and the density of a component. For example, Fig. 3 shows a character 'A', labelled L_A by connected component labelling. The primary attributes, coordinates and area of the connected component, are assumed to be (X_A, Y_A) and A_A, respectively; accordingly, the corresponding attributes are

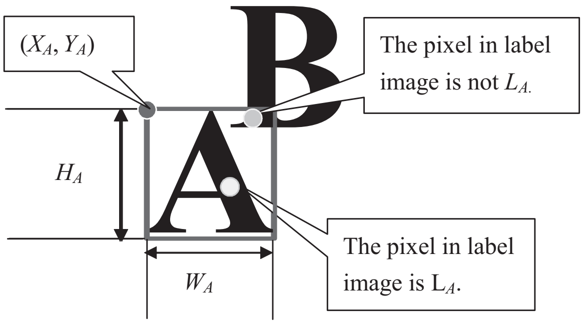
$$\text{Aspect ratio:}\qquad R_A=\frac{H_A}{W_A} \tag{18}$$

$$\text{Area:}\qquad A_A=W_A\times H_A \tag{19}$$

Density:

$$D_A=\sum_{r=0}^{H_A-1}\sum_{c=0}^{W_A-1}\frac{\begin{cases}1 & \text{if } g(X_A+r,Y_A+c)=L_A\\ 0 & \text{otherwise}\end{cases}}{A_A} \tag{20}$$

where $g(\ )$ is the label of image.



(X_A, Y_A)
The pixel in label image is not L_A.
H_A
W_A
The pixel in label image is L_A.

Fig. 3 *Primary attributes of a connected component, 'A'*

3.3 Determining components of characters

After the attributes of connected components have been determined, the candidates of character components can be determined according to the following processes.

3.3.1 Delete the noise components: In this process, some connected components can be deleted by examining whether their heights, widths or sizes are too large. The size of an input image is assumed to be $W \times H$, and the thresholding operators are then as listed below

$$\text{Thresholding operators}:\begin{cases}W_A > \text{width\_threshold}\\ H_A > \text{height\_threshold}\\ A_A > \text{area\_threshold}_{\max}\end{cases} \tag{21}$$

where the parameters width_threshold, height_threshold and area_threshold$_{\max}$ are application adaptable. This work applies width_threshold $= \lfloor W/10\rfloor$, height_threshold $= \lfloor H/4\rfloor$ and area_threshold$_{\max} = \lfloor W\times H/10.5\rfloor$.

3.3.2 Adjust the horizontal positions and widths of components and eliminate inappropriate components: The aspect ratios of some special characters, such as '1', 'I' and 'J', are larger than those of other characters. If a fixed aspect ratio is used to delete inappropriate components, then these special characters will be deleted. Therefore, the horizontal positions and widths of these special characters should be adjusted before the elimination of inappropriate components. The maximum and minimum possible aspect ratios, $R_{\text{S-max}}$ and $R_{\text{S-min}}$, are defined for characters '1', 'I' and 'J', and $R_{\text{N-max}}$ and $R_{\text{N-min}}$ are defined for other characters. Then, the horizontal position and width of all connected components can be adjusted using the following equations

$$X_A=\begin{cases}X_A+\left(\dfrac{W_A}{2}-\dfrac{H_A}{R_{\text{N-max}}+R_{\text{N-min}}}\right) & \\ \quad\text{if } R_{\text{S-min}}<R_A<R_{\text{S-max}} & \\ X_A & \text{otherwise}\end{cases} \tag{22}$$

$$W_A=\begin{cases}\dfrac{H_A}{(R_{\text{N-max}}+R_{\text{N-min}}/2)} & \text{if } R_{\text{S-min}}<R_A<R_{\text{S-max}}\\ W_A & \text{otherwise}\end{cases} \tag{23}$$

After the horizontal position and the width of the components are adjusted, the ratios and densities of the components modified according to (22) and (23) must be calculated. Then, objects with excessively large or small aspect ratios and densities can be deleted by applying the following equation

$$\text{Thresholding operators}: \begin{cases} R_{\mathrm{A}} > \text{ratio\_threshold}_{\max} \\ R_{\mathrm{A}} < \text{ratio\_threshold}_{\min} \\ D_{\mathrm{A}} > \text{density\_threshold}_{\max} \\ D_{\mathrm{A}} < \text{density\_threshold}_{\min} \end{cases} \tag{24}$$

where the parameters ratio_threshold$_{\max}$, ratio_threshold$_{\min}$, density_threshold$_{\max}$ and density_threshold$_{\min}$ are fixed for most applications.

This study applied ratio_threshold$_{\max} = 6.5$, ratio_threshold$_{\min} = 0.5$, density_threshold$_{\max} = 0.8$ and density_threshold$_{\min} = 0.2$.

3.3.3 Determine candidate components of characters: The characters on a licence plate are typically arranged from left to right. Even when the plate is inclined, the position or size of one character is highly correlated to that of the character to its left or right. Consequently, a sequence of connected components is within the positional tolerance and the size tolerance, and is thus a candidate sequence of character components.

If all connected components are given by $C = \{C_0, C_1, C_2, \ldots\}$, the elements of the candidate sequence V_{C_i} are $\{V^0_{C_i}, V^1_{C_i}, V^2_{C_i}, \ldots, V^{L-1}_{C_i}\}$ and $V_{C_i} \subset C$. The positional tolerance $P(V^k_{C_i})$ and the size tolerance $S(V^k_{C_i})$ of element $V^k_{C_i}$ in candidate sequence V_{C_i} are, respectively, defined by (25) and (26)

$$P(V^k_{C_i}) = \Big\{(x, y) \Big| |x - X_{V^k_{C_i}} - W_{V^k_{C_i}} - r_x(k) \times H_{V^k_{C_i}}| < r_x(k) \times H_{V^k_{C_i}} \quad \text{and} \quad |y - Y_{V^k_{C_i}}| < r_y(k) \times W_{V^k_{C_i}}\Big\} \tag{25}$$

$$S(V^k_{C_i}) = \Big\{(w, h) \Big| |w - r_h(k) \times H_{V^k_{C_i}}| < r_x(k) \times H_{V^k_{C_i}} \quad \text{and} \quad |h - r_y(k) \times W_{V^k_{C_i}}| < r_w(k) \times W_{V^k_{C_i}}\Big\} \tag{26}$$

where $(X_{V^k_{C_i}}, Y_{V^k_{C_i}})$ and $(W_{V^k_{C_i}}, H_{V^k_{C_i}})$ are the position and size of the boundary box of component $V^k_{C_i}$, whereas $r_x(k)$, $r_y(k)$, $r_w(k)$ and $r_h(k) \in \boldsymbol{R}$ are ratios which depend on the order k of the element $V^k_{C_i}$ in a candidate V_{C_i}. Elements in V_{C_i} satisfy a test condition $Z(V^{k-1}_{C_i}, V^k_{C_i})$ if $(X_{V^k_{C_i}}, Y_{V^k_{C_i}}) \subset P(V^{k-1}_{C_i})$ and $(W_{V^k_{C_i}}, H_{V^k_{C_i}}) \subset S(V^{k-1}_{C_i})$; otherwise, they do not. This work sets $r_x(k) = 0.8$ for $k \neq 2$ and $r_x(k) = 1.28$ for $k = 2$, $r_y(k) = 0.6$, $r_w(k) = 0.35$ and $r_h(k) = 0.35$.

The component $V^0_{C_i}$ is the first element in the candidate sequence V_{C_i}. The candidate sequence V_{C_i} increases when a component C_j satisfies the condition $Z(V^0_{C_i}, C_j)$, and will stop increasing when no C satisfies this condition. Equation (27) describes the increase in candidate sequence V_{C_i}

$$V^{L+1}_{C_i} = \begin{cases} C_j & \text{if } Z(V^{\mathrm{L}}_{C_i}, C_j) \text{ is satisfied} \\ \text{no operation} & \text{otherwise} \end{cases} \tag{27}$$

where $V^L_{C_i}$ is the last element in V_{C_i} and $V^{L+1}_{C_i}$ is a new element of V_{C_i} if $Z(V_{C^L_i}, C_j)$ is satisfied. Finally, a candidate sequence V_{C_i} becomes a candidate sequence of character components if V_{C_i} includes four, five or six elements.

3.3.4 Compensation: The compensation pass is performed when the candidate character sequence includes one or two characters fewer than in preceding passes. A sequence of candidates with four or five connected components is very likely to be a character component sequence, with sufficient information to recover the lost character components. Therefore, the compensation pass tries to solve the problems in the two aforementioned cases, in which one or two characters are lost.

In the case of a single lost character, the lost character is either the leftmost or the rightmost character in a character component sequence, because they are easily obscured by nearby dust, screws or frames. Similarly, in a case in which two characters are lost, the lost characters may be the leftmost two, the rightmost two or the leftmost one and the rightmost one.

Lost characters can be compensated for by estimating their possible positions and sizes. To estimate the positions and sizes of possible characters, the positions and sizes of connected components in the corresponding character candidate sequence are considered. More precisely, the estimated positions and sizes are obtained by extrapolation from the positions and sizes of connected components in the sequence. After the positions and sizes of the lost characters have been thus obtained, the sizes are normalised to 20×35, the size of the character patterns herein, and the pattern matching method is applied to identify the characters in regions bounded by estimated positions and sizes. Fig. 4 presents the patterns. The matching score is the number of matching pixels of the patterns and the estimated region. The matching score of a real lost character generally exceeds that of the estimated region that is not a lost character. Therefore, the matching score can be used to determine the real lost characters.

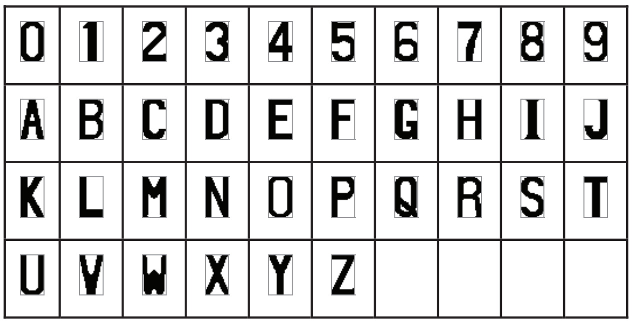


Fig. 4 *Pattern of characters (pixel size = 20 × 35)*

4 Experimental results

This work proposes a modified connected component labelling algorithm. The processing time of the proposed algorithm is calculated by applying it to test images of sizes 128×128, 256×256 and 512×512. The proposed algorithm is compared, in terms of processing time and memory required for storage, with those presented in the work of Samet [26] and Suzuki *et al.* [28]. Tables 1 and 2 show the data for comparison. Table 1 indicates that the proposed algorithm is much faster than the quadtree algorithm, and from two to three times faster than the forward

Table 1: Comparison of average processing times (ms)

Algorithm	Average processing time, ms		
	128 × 128	256 × 256	512 × 512
Quadtree [27]	30	120	1416
Forward and backward [28]	6	20	110
Proposed algorithm	5	10	40

Table 2: Comparison of average memory required for storage (bytes)

Algorithm	Average memory storage size (field)		
	128 × 128	256 × 256	512 × 512
Quadtree [27]	420	1606	4846
Forward and backward [28]	96	359	1990
Proposed algorithm	96	359	1990

and backward algorithm, as the size of the image increases. Table 2 compares the average memory required for storage by the quadtree, forward and backward and proposed algorithms. The proposed algorithm requires less than or as much as memory than the quadtree algorithm or the forward and backward algorithm.

The experimental results presented in Fig. 5 are the outputs of the sub-processes described in Section 3. Clearly, in Fig. 5, the six character components were successfully extracted by the proposed method. The smearing, inclination and lack of sharpness of licence plates can prevent some of the character components from being extracted. Part of the image of the licence plate shown in Fig. 6 is not sharp, so, as Fig. 7*a* indicates, the number of character components after processing is five. Fig. 7*b* reveals that the lost character can be recovered by the proposed compensation.

The system is applied out-of-doors in a building parking lot, to evaluate its recognition rate, reliability and speed. The test involved 228 cars at various capture angles. Of these cars, the identities of 218 were successfully extracted by the proposed method. The extraction rate is 95.6%. A successful extraction is one in which all six characters on the licence plate are determined.

Fig. 8 shows some of the 218 images from which extractions were successful, and the corresponding extracted

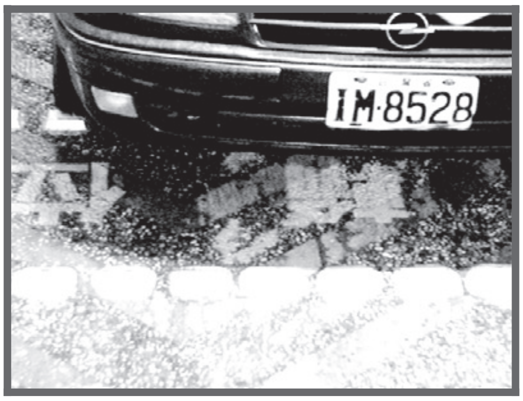


Fig. 6 *Vehicle licence plate with a character component that is not sharp*

characters. The results verify that the system successfully extracts characters in any part of the captured image without using information on the luminance of the surroundings; the size, position and inclination of the licence plate, or the colour of the vehicle; furthermore, the loss characters can be identified.

Fig. 9 presents the images from which the proposed AVI system failed to extract all six characters because the licence plate was misshaped; more than two characters were lost, or a screw, the frame or dust covered the characters. The absence of an observable licence plate or fast motion of the car also prevented character extraction.

The average processing time was 1.032 s. The test was performed using a PC, with a Pentium 500 MHz CPU, 128 MB DRAM and a Windows 2000 Operating System.

The demo program of a commercial AVI system, the SeeCar Licence Plate Recognition product from Hi-Tech Solutions (http://www.htsol.com/SeeCar.html), was used to process the test images herein. Figs. 10 and 11 show some of the results. Fig. 10 presents the processed images, to

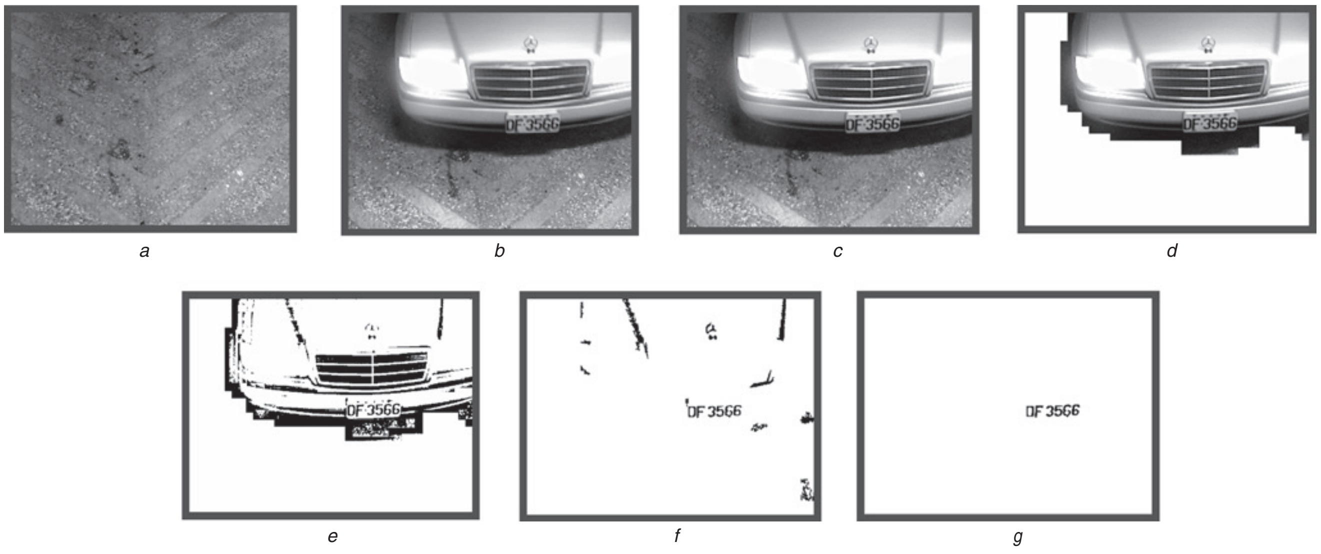


Fig. 5 *Experimental results for the character extraction system*

a Pre-captured background image
b Image of incoming car
c Image after de-interlacing
d Image after unnecessary background has been masked
e Image after segmentation and the labelling of connected components
f Image after components with bad attributes have been eliminated
g Successful extraction of character components

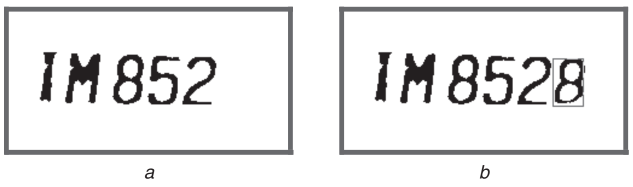


Fig. 7 *Compensation*

a Only five character objects are extracted
b Full character objects are extracted after compensation

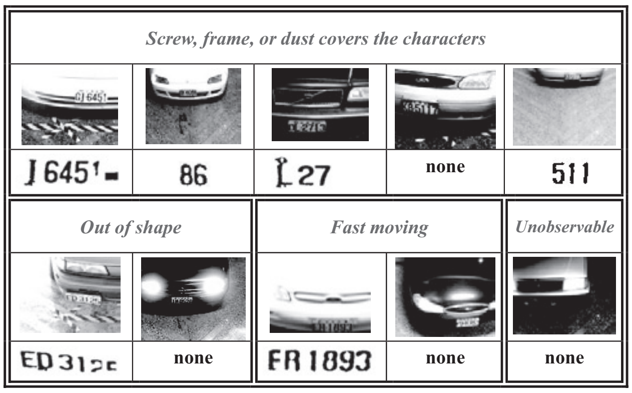


Fig. 9 *Images from which the proposed system failed to extract characters*

which the SeeCar demo program was applied to extract characters; these images were obtained from the test images of Fig. 8; the test images of Fig. 9 are also processed using the demo program, yielding the results presented in Fig. 11.

The results in Fig. 10 reveal that the SeeCar demo program recognised correctly only the characters in ten images, failing to recognise those in 14 images. The characters were not correctly recognised for all the images in Fig. 11. These test results indicate that the SeeCar system is not robust when applied to noisy, blurred or badly illuminated licence plates, and the proposed system outperforms the commercial SeeCar system.

5 Conclusion

This work proposed an AVI system for extracting characters from licence plates. Most applications of such an AVI system are outdoors and are consumer-driven, where the recognition rate, system reliability and processing speed are crucial. Most conventional research on the extraction of characters from licence plates is weak in handling outdoor applications, and the proposed methods require much processing time. Accordingly, an improved character extraction method is proposed herein. The experimental results establish the feasibility of the proposed approach. The advantage of the approach is that it can successfully extract licence plate characters from any part of a captured image without using information on the luminance of the surroundings, the size or inclination of the licence plate, or the colour of the vehicle. Moreover, the processing time of this system is satisfactory for various applications. Future work should develop a more robust segmentation method and a more sophisticated compensation method to prevent character loss.

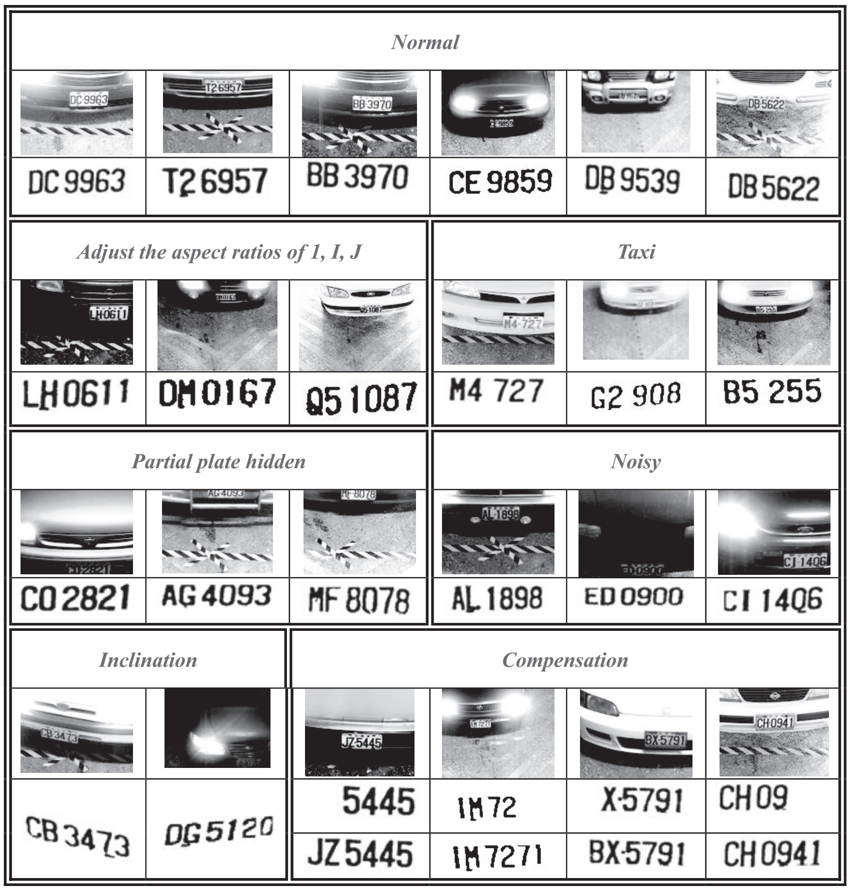


Fig. 8 *Some of the images from which the proposed system successfully extracted characters*

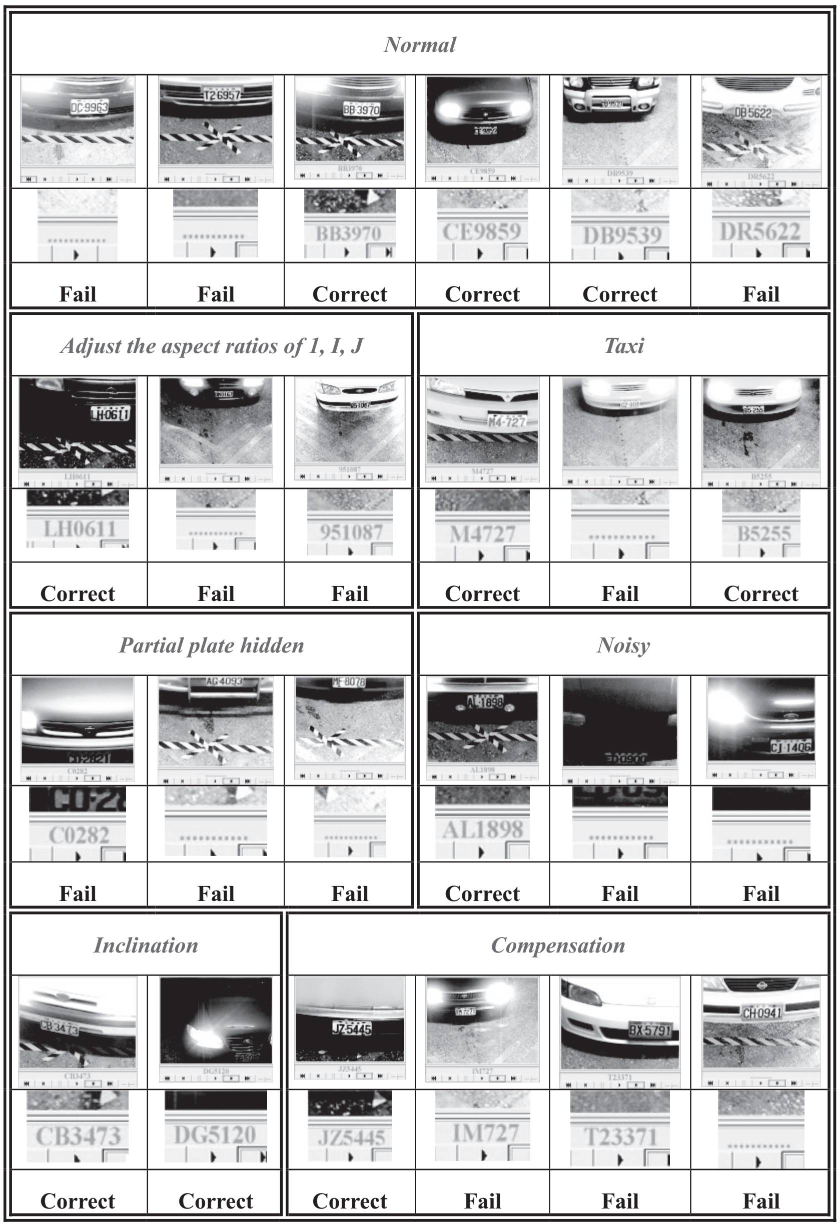


Fig. 10 *Images of Fig. 8 processed by the SeeCar demo program*

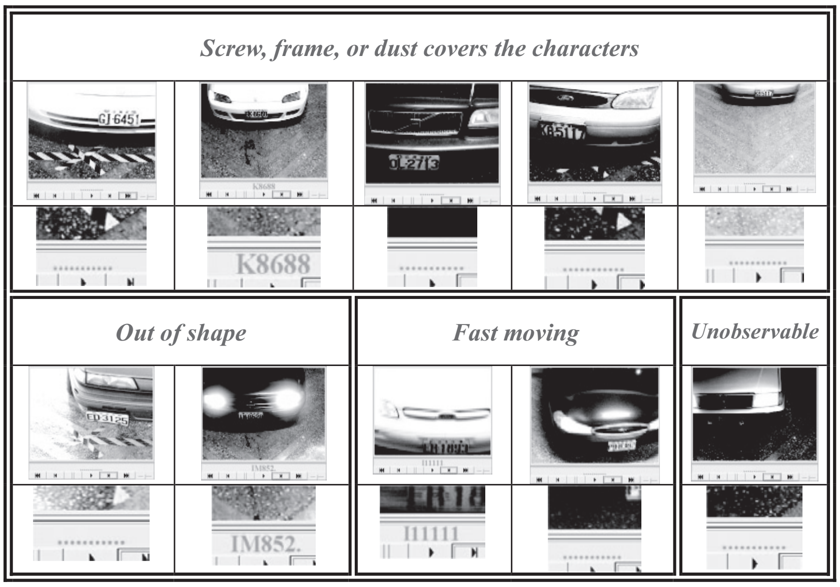


Fig. 11 *Images of Fig. 9 processed by the SeeCar demo program*

6 References

1 Castello, P., Coelho, C., Del Ninno, E., Ottaviani, E., and Zanini, M.: 'Traffic monitoring in motorways by real-time number plate recognition'. IEEE Int. Conf. on Image Anal. Process., 1999, pp. 1128–1131

2 Tanaka, Y., Kanayama, K., and Sugimura, H.: 'Travel-time data provision system using vehicle license number recognition devices'. Proc. IEEE Intelligent Vehicles, July 1992, pp. 353–358

3 Kanayama, K., Fujikawa, Y., Fujimoto, K., and Horino, M.: 'Development of vehicle-license number recognition system using real-time image processing and its application to travel-time measurement'. IEEE Veh. Technol. Conf., May 1991, pp. 798–804

4 Busch, C., Dörner, R., Freytag, C., and Ziegler, H.: 'Feature based recognition of traffic video streams for online route tracing'. IEEE Veh. Technol. Conf., May 1998, vol. 3, pp. 1790–1794

5 Sirithinaphong, T., and Chamnongthai, K.: 'The recognition of car license plate for automatic parking system'. IEEE Signal Process. Appl., August 1999, vol. 1, pp. 455–457

6 Matthews, N.D., An, P.E., Charnley, D., and Harris, C.J.: 'Vehicle detection and recognition in greyscale imagery', *Control Eng. Pract.*, 1996, **4**, (4), pp. 473–479

7 Cui, Y., and Huang, Q.: 'Extracting characters of license plates from video sequences', *J. Mach. Vis. Appl.*, 1998, **10**, (5–6), pp. 308–320

8 Barroso, J., Rafael, A., Dagless, E.L., and Bulas-Cruz, J.: 'Number plate reading using computer vision'. IEEE Int. Symp. Indust. Electron., July 1997, vol. 3, pp. 761–766

9 He, M.G., Harvey, A.L., and Danelutti, P.: 'Car number plate detection with edge image improvement'. IEEE Int. Symp. Signal Process. Appl., August 1996, vol. 2, pp. 597–600

10 He, M.G., Harvey, A.L., and Vinay, T.: 'Hough transform in car number plate skew detection'. IEEE Int. Symp. Signal Process. Appl., August 1996, vol. 2, pp. 593–596

11 Comelli, P., Ferragina, P., Granieri, M.N., and Stabile, F.: 'Optical recognition of motor vehicle license plates', *IEEE Trans. Veh. Technol.*, 1995, **44**, (4), pp. 790–799

12 Lee, E.R., Kim, P.K. and Kim, H.J.: 'Automatic recognition of a car license plate using color image processing'. IEEE Int. Conf. on Image Process., 1994, vol. 2, pp. 301–305

13 Soh, Y.S.: 'Design of real time vehicle identification system'. IEEE Int. Conf. on Syst. Man Cybern., 1994, vol. 3, pp. 2147–2152

14 De Vena, L.: 'Number plate recognition by hierarchical neural networks'. IEEE Int. Conf. Neural Netw., October 1993, vol. 3, pp. 2105–2108

15 Kim, K.K., Kim, K.I., Kim, J.B., and Kim, H.J.: 'Learning-based approach for license plate recognition'. IEEE Int. Workshop on Neural Netw. Signal Process., December 2000, vol. 2, pp. 614–623

16 Wei, W., Li, Y., Wang, M., and Huang, Z.: 'Research on number-plate recognition based on neural networks'. IEEE Int. Workshop on Neural Netw. Signal Process., September 2001, pp. 529–538

17 Zimic, N., Ficzko, J., Mraz, M., and Virant, J.: 'The fuzzy logic approach to the car number plate locating problem'. IEEE Int. Conf. Inf. Intell. Syst., December 1997, pp. 227–230

18 Nijhuis, J.A.G., ter Brugge, M.H., Helmholt, K.A., Pluim, J.P.W., Spaanenburg, L., Venema, R.S., and Westenberg, M.A.: 'Car license plate recognition with neural networks and fuzzy logic'. IEEE Int. Conf. Neural Netw., 1995, vol. 5, pp. 2232–2236

19 Kim, S.K., Kim, D.W., and Kim, H.J.: 'A recognition of vehicle license plate using a genetic algorithm based segmentation'. IEEE Int. Conf. Image Process., 1996, vol. 1, pp. 661–664

20 Gao, D.S., and Zhou, J.: 'Car license plates detection from complex scene'. IEEE Int. Conf. Signal Process. Proc., August 2000, vol. 2, pp. 1409–1414

21 Torres-Méndez, L.A., Ruiz-Suárez, J.C., Sucar, L.E., and Gómez, G.: 'Translation, rotation, and scale-invariant object recognition', *IEEE Trans. Syst. Man Cybern.*, 2000, **30**, (1), pp. 125–130

22 Cowell, J.R.: 'Syntactic pattern recognizer for vehicle identification numbers', *Image Vis. Comput.*, 1995, **13**, (1), pp. 13–19

23 Naito, T., Tsukada, T., Yamada, K., Kozuka, K., and Yamamoto, S.: 'Robust license-plate recognition method for passing vehicles under outside environment', *IEEE Trans. Veh. Technol.*, 2000, **49**, (6), pp. 2309–2319

24 Gonzalez, R.C., and Woods, R.E.: 'Digital image processing' (Prentice-Hall, New Jersey, 2002, 2nd edn.), Chaps. 3 and 10, pp. 89–108 and 595–634

25 Friedman, M., and Kandel, A.: 'Introduction to pattern recognition' (World Scientific, Singapore, 1999), Chaps. 3 and 4, pp. 77–140

26 Samet, H.: 'Connected component labeling using quadtrees', *J. Assoc. Comput. Mach.*, 1981, **28**, (3), pp. 487–501

27 Rosenfeld, A., and Pfaltz, J.L.: 'Sequential operations in digital picture processing', *J. Assoc. Comput. Mach.*, 1966, **13**, (4), pp. 471–494

28 Suzuki, J., Horiba, K., and Sugie, N.: 'Fast connected-component labeling based on sequential local operations in the course of forward raster scan followed by backward raster scan', *Pattern Recognit.*, 2000, **2**, pp. 434–437

29 Khanna, V., Gupta, P., and Hwang, C.J.: 'Finding connected components in digital images'. Int. Conf. Inf. Technol., 2001, pp. 652–656