Automatic License Plate Recognition

Shyang-Lih Chang, Li-Shien Chen, Yun-Chung Chung, and Sei-Wan Chen, *Senior Member, IEEE*

***Abstract*—Automatic license plate recognition (LPR) plays an important role in numerous applications and a number of techniques have been proposed. However, most of them worked under restricted conditions, such as fixed illumination, limited vehicle speed, designated routes, and stationary backgrounds. In this study, as few constraints as possible on the working environment are considered. The proposed LPR technique consists of two main modules: a license plate locating module and a license number identification module. The former characterized by fuzzy disciplines attempts to extract license plates from an input image, while the latter conceptualized in terms of neural subjects aims to identify the number present in a license plate. Experiments have been conducted for the respective modules. In the experiment on locating license plates, 1088 images taken from various scenes and under different conditions were employed. Of which, 23 images have been failed to locate the license plates present in the images; the license plate location rate of success is 97.9%. In the experiment on identifying license number, 1065 images, from which license plates have been successfully located, were used. Of which, 47 images have been failed to identify the numbers of the license plates located in the images; the identification rate of success is 95.6%. Combining the above two rates, the overall rate of success for our LPR algorithm is 93.7%.**

***Index Terms*—Color edge detector, fuzzification, license number identification, license plate locating, license plate recognition (LPR), self-organizing (SO) character recognition, spring model, topological sorting, two-stage fuzzy aggregation.**

I. Introduction

AUTOMATIC license plate recognition (LPR) plays an important role in numerous applications such as unattended parking lots [31], [35], security control of restricted areas [8], traffic law enforcement [7], [33], congestion pricing [5], and automatic toll collection [20]. Due to different working environments, LPR techniques vary from application to application. Most previous works have in some way restricted their working conditions [9], such as limiting them to indoor scenes, stationary backgrounds [30], fixed illumination [7], prescribed driveways [22], [26], limited vehicle speeds [1], or designated ranges of the distance between camera and vehicle [23]. The aim of this study is to lessen many of these restrictions.

Of the various working conditions, outdoor scenes and nonstationary backgrounds may be the two factors that most influence the quality of scene images acquired and in turn the complexity of the techniques needed. In an outdoor environment, illumination not only changes slowly as daytime progresses, but may change rapidly due to changing weather conditions and passing objects (e.g., cars, airplanes, clouds, and overpasses). In addition, pointable cameras create dynamic scenes when they move, pan or zoom. A dynamic scene image may contain multiple license plates or no license plate at all. Moreover, when they do appear in an image, license plates may have arbitrary sizes, orientations and positions. And, if complex backgrounds are involved, detecting license plates can become quite a challenge.

Typically, an LPR process consists of two main stages: 1) locating license plates and 2) identifying license numbers. In the first stage, license plate candidates are determined based on the features of license plates. Features commonly employed have been derived from the license plate format and the alphanumeric characters constituting license numbers. The features regarding license plate format include shape, symmetry [15], height-to-width ratio [23], [25], color [17], [25], texture of grayness [2], [25], spatial frequency [26], and variance of intensity values [8], [10]. Character features include line [34], blob [13], the sign transition of gradient magnitudes, the aspect ratio of characters [12], the distribution of intervals between characters [28], and the alignment of characters [32]. In reality, a small set of robust, reliable, and easy-to-detect object features would be adequate.

The license plate candidates determined in the locating stage are examined in the license number identification stage. There are two major tasks involved in the identification stage, character separation and character recognition. Character separation has in the past been accomplished by such techniques as projection [11], [30], morphology [2], [10], [28] relaxation labeling, connected components [25], and blob coloring. Every technique has its own advantages and disadvantages. Since the projection method assumes the orientation of a license plate is known and the morphology method requires knowing the sizes of characters, these two approaches are not appropriate for our application because of their required assumptions. Relaxation labeling is by nature iterative and often time consuming. In this study, a hybrid of connected components and blob coloring techniques is considered for character separation.

There have been a large number of character recognition techniques reported. They include genetic algorithms [17], artificial neural networks [2], [16], [26], fuzzy c-means [25], support vector machine [16], Markov processes [6], and finite automata [1]. These methods can be broadly classified into iterative and noniterative approaches. There is a tradeoff between these two groups of approaches; iterative methods achieve better accuracy, but at the cost of increased time complexity. In this study, we pay more attention to accuracy than time complexity whenever

Manuscript received December 11, 2002; revised December 8, 2003. This work was supported by the National Science Council, Republic of China, under Contract NSC-89-2218-E-003-002. The Associate Editor for this paper was A. Broggi.

S.-L. Chang, L.-S. Chen, and Y.-C. Chung are with the Department of Information and Computer Education, National Taiwan Normal University, Taipei, Taiwan, R.O.C.

S.-W. Chen is with the Graduate Institute of Computer Science and Information Engineering, National Taiwan Normal University, Taipei, Taiwan, R.O.C. (e-mail: schen@csie.ntnu.edu.tw).

Digital Object Identifier 10.1109/TITS.2004.825086

TABLE I
STYLES OF LICENSE PLATES UNDER CONSIDERATION

Vehicle category	Plate color	Character color	Example
Private automobile	White	Black	**E1•2345**
Taxi	White	Red	E1•234
Tour bus	Red	White	E1•234
Truck	Green	White	E1•234
Government vehicle	White	Green	E1•234

a choice has to be made between them. For this, we developed our own character recognition technique, which is based on the disciplines of both artificial neural networks and mechanics.

The rest of this paper is organized as follows. In Section II, the types of license plates to be considered are described, followed by the fundamental idea of the proposed LPR technique. The two primary stages of the proposed technique, license plate location and license number identification, are discussed in detail in Sections III and IV, respectively. Experimental results are presented in Section V. Concluding remarks and ideas for future work are given in Section VI.

II. LPR

In this section, the styles of license plate that are considered in this study are discussed, followed by a brief description of the proposed LPR process. Table I shows assorted styles of license plates found on vehicles in Taiwan. Each style is associated with a particular class of vehicle. The classes include private automobile, taxi, tour bus, truck, and government vehicles. Other categories of vehicles, such as diplomatic cars and military vehicles, are not addressed since they are rarely seen. Styles of license plates can easily be distinguished based on two attributes: 1) the combination of colors used and 2) the compositional semantics of license numbers.

As shown in Table I, each style has a different foreground and/or background color. However, in all only four distinct colors (white, black, red, and green) are utilized in these license plates. We shall pay attention to these four colors when searching for license plates in an input image. The compositional semantics of license numbers provides additional information for differentiating styles of license plates. As can be seen in Table I, every license number is composed of two parts separated by a hyphen (e.g., E1-2345). The first part consists of two characters, one of which must be an alphabetical character (e.g., E1, 2F, and EF). The second part may contain four (e.g., 2345) or three (e.g., 234) numerals, the former being used only on private automobiles, and the latter being used on the other vehicle classes.

Fig. 1 shows the proposed LPR process. We assume that the process is incorporated in an event detection system, e.g., a vehicle detector or a traffic law enforcement system. Once the system detects an event, the camera along with the system is activated. The image acquired by the camera is then sent to the LPR process, in which potential license plates are extracted from the image. If no license plate is found, the process returns to await another input image. However, oftentimes multiple license plate candidates are detected. They are closely examined

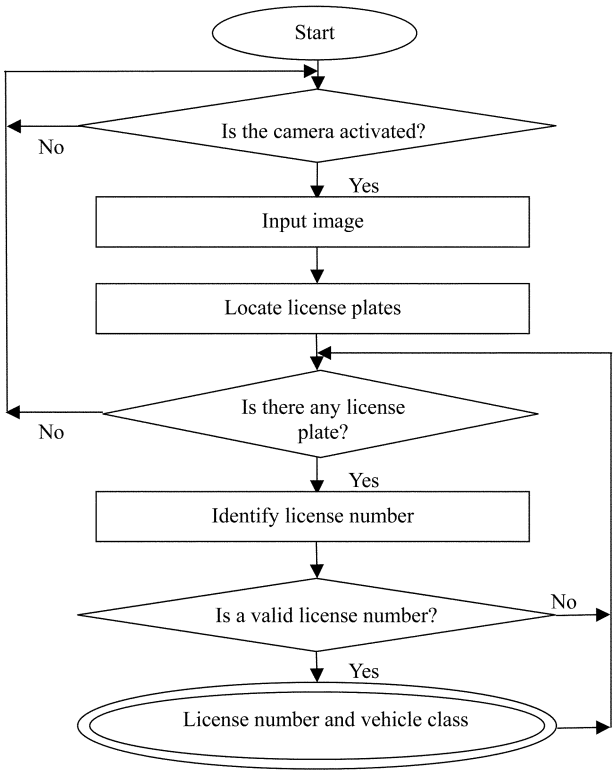

Fig. 1. Diagram of the proposed LPR process.

at the license number identification stage. There are two essential tasks involved in this stage, character segmentation and recognition. These two tasks are alternatively invoked in order to achieve optimal results for both segmentation and recognition. The characters recovered from a license plate candidate at this stage are next verified at the confirmation stage. The group of characters will be deemed to form a valid license number if it agrees with the compositional semantics of license numbers mentioned earlier. Both the valid license number and the associated vehicle class will be returned by the LPR process. The identification and confirmation stages repeat for all of the license plate candidates. Afterwards, the process returns to await another input image.

In Sections III and IV, we look at the details of the license plate locating module and the license number identification module.

III. LICENSE PLATE LOCATING MODULE

A. Basic Concepts

A flowchart for the license plate locating module is shown in Fig. 2. The input to this module is an RGB color image. Recall that only four colors (white, black, red, and green) are utilized in the license plates that we consider. Note also that there are many edges, which are in close mutual proximity and are dispersed in a repetitive manner, contained in a license plate. The above observations motivates us to develop a color edge detector. The edge detector is sensitive to only three kinds of edges, black-white, red-white, and green-white (see the last column of Table I). By ignoring other types of edges in an image, very few edges due to objects other than license plates are detected, even

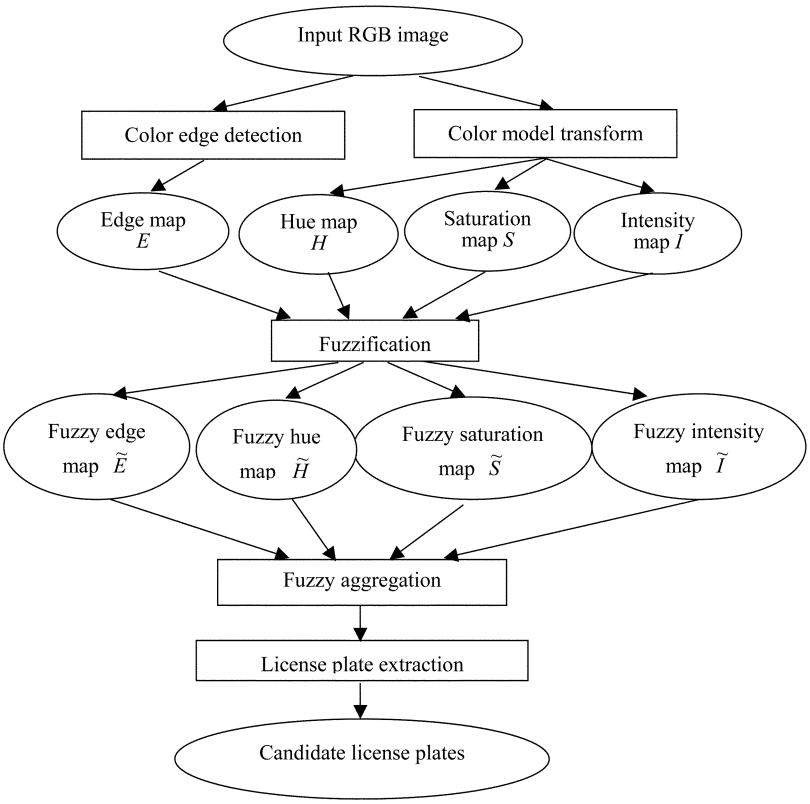

Fig. 2. Flowchart for the license plate locating module.

when the image background is very cluttered. Let $\mathbf{E}$ denote the edge map computed from the input image using the color edge detector.

Next, the RGB space of the input color image is transformed into the HSI space. Let (R, G, B) and (H, S, I) denote the (red, green, blue) and (hue, saturation, intensity) values of an image pixel, respectively. The transform from (R, G, B) to (H, S, I) [3] is

$$I = \frac{(r+g+b)}{3}$$
$$S = 1 - \frac{\min\{r,g,b\}}{I}$$
$$H = \cos^{-1}\left\{\frac{(r-g)+(r-b)}{2\left[(r-g)^2+(r-b)(g-b)\right]^{1/2}}\right\} \quad (1)$$

where $r = R/255$, $g = G/255$ and $b = B/255$. There are a number of intriguing characteristics associated with the HSI color model which are useful for our application, including the invariance of hue to both illumination and shading, and the invariance of saturation to both viewing direction and surface orientation. Let $\mathbf{H}$, $\mathbf{S}$, and $\mathbf{I}$ be the maps preserving the hue, saturation, and intensity components of the transformed image, respectively.

It is inevitable that maps $\mathbf{E}$, $\mathbf{H}$, $\mathbf{S}$, and $\mathbf{I}$ are less than perfect in view of noise, measurement error, and imperfect processing. In order to compensate for this drawback, we appeal to the soft computing techniques rooted in fuzzy (for license plate location) and neural (for license number identification) disciplines. Let $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, $\tilde{\mathbf{I}}$, and $\tilde{\mathbf{E}}$ be the fuzzy versions of $\mathbf{H}$, $\mathbf{S}$, $\mathbf{I}$, and $\mathbf{E}$. The entries in the fuzzy maps represent the degrees of belonging to a license plate. A two-stage fuzzy aggregator is introduced to integrate the maps. In the first stage, fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$ are integrated. The resulting map next combines with $\tilde{\mathbf{E}}$ in the second stage leading to a single map, denoted $\tilde{\mathbf{M}}$. The reason of using the two-stage aggregator is because the intrinsic characteristics (related to color) of $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$ are different from that (related to edge magnitude) of $\tilde{\mathbf{E}}$. Afterwards, based on map $\tilde{\mathbf{M}}$, interesting regions are located in the input image, which have locally maximal $\tilde{m}$ values. License plate candidates are then determined to be those interesting areas whose sizes are large enough.

B. Color Edge Detection

The color edge detector focuses on only three kinds of edges (i.e., black-white, red-white and green-white edges). Consider a black-white edge, and suppose that the input RGB color image has been normalized into an rgb image. Ideally, the (r, g, b) values of a white pixel and a black pixel should be $(1, 1, 1)$ and $(0, 0, 0)$, respectively. Their differences $(\Delta r, \Delta g, \Delta b)$ are either $(1, 1, 1)$ or $(-1, -1, -1)$, so all the components of the difference vector between a white and a black pixel will have the same sign. This property is considerably stable under environmental variations. A black-white edge pixel is then defined based on this property as follows. An image pixel is regarded as a black-white edge point if all of the signs of components of the difference vector between the pixel and one of its neighbors are the same, i.e., $\mathrm{sign}(\Delta r_i) = \mathrm{sign}(\Delta g_i) = \mathrm{sign}(\Delta b_i)$, $i \epsilon N$, where N is the set of neighbors of the image pixel. We also store its edge magnitude defined as $\min\{|\Delta r_i|, |\Delta g_i|, |\Delta b_i|\}$. Edge magnitudes will be exploited later to derive fuzzy edge maps.

In a similar way, an image pixel is considered to be a red-white edge point if its difference vector $(\Delta r_i\ \Delta g_i\ \Delta b_i)$ for some $i \epsilon N$ satisfies the following conditions: 1) $\text{sign}(\Delta r_i) = \text{sign}(\Delta g_i) = \text{sign}(\Delta b_i)$ and 2) $|\Delta r_i| < |\Delta g_i|$ and $|\Delta r_i| < |\Delta b_i|$. The magnitude of the edge pixel is defined as $\min\{|\Delta g_i|, |\Delta b_i|\}$. Finally, an image pixel is regarded to be a green-white edge pixel if for some $i \epsilon N$, 1) $\text{sign}(\Delta r_i) = \text{sign}(\Delta g_i) = \text{sign}(\Delta b_i)$ and 2) $|\Delta g_i| < |\Delta r_i|$ and $|\Delta g_i| < |\Delta b_i|$. Its edge magnitude is determined by $\min\{|\Delta r_i|, |\Delta b_i|\}$. Image pixels, which are not edge points, are given zero edge magnitudes.

C. Fuzzy Maps

The basic idea of generating a fuzzy map from a given map (e.g., $\mathbf{H}$, $\mathbf{S}$, $\mathbf{I}$, or $\mathbf{E}$) is as follows. Since every map encodes some characteristic about the scene, the entry of any cell in the map expresses the degree of the cell possessing the property. In order to highlight the cells corresponding to the objects of interest (e.g., license plates), we assign large entries to those cells that are compatible with the known characteristics of the objects. Such large entries indicate a high degree of existence of an interesting object. We call the resultant map the characteristic map of the original map.

Since the input data (both the given map and the object characteristics) are not perfect, uncertainties should be taken into account during the computation of the characteristic map. Fuzzy sets have been known to provide an elegant tool for modeling uncertainties [14], [18], [27]. In this study, we introduce fuzzinesses into the entries of the characteristic map and refer to the result as the fuzzy map. There are several ways to realize fuzziness. We define a generalized fuzzy set, termed "like a license plate," on the respective sets of hue, saturation, intensity, and edge magnitude. Each of the four sets serves as a universal set of the fuzzy set.

1) $\tilde{\mathbf{H}}$ *Map:* Consider the universal set of hue values. Suppose that the object of interest has a color C whose corresponding hue value is h_c. Given an entry in map $\mathbf{H}$, say h, the membership degree, μ_c, of the entry belonging to fuzzy set "like the object" can be written

$$\mu_c(h) = \exp\left(-a\,|h - h_c|\right)$$

where a is a positive constant. If the given entry h is equal to that of the interesting object h_c, then the degree of membership is 1. As the difference between the hues increases, the degree of membership decreases to an asymptotic value of 0. Recall that there are four colors (black, white, red and green) utilized in the license plates that are of interest. Let h_r and h_g be the hue values for red and green, respectively. Note that the hues of achromatic colors (i.e., all levels of grey, including black and white) are not defined since the denominator of the equation for hue in (1) is zero. Therefore, we will highlight red and green, but not white and black based on map $\mathbf{H}$. The membership function of fuzzy map $\tilde{\mathbf{H}}$ is finally defined as

$$\mu_{\tilde{H}}(h) = u\left(\mu_r(h), \mu_g(h)\right) \tag{2}$$

where u can be any fuzzy union operator (any t-conorm function).

2) $\tilde{\mathbf{S}}$ *Map:* Since fuzzy map $\tilde{\mathbf{H}}$ can only express the colors red and green, we need other means to handle black and white. According to the S in (1), all achromatic colors have the same saturation S. In addition, this value is smaller than that of any chromatic color. Based on these two facts, we generate a fuzzy map $\tilde{\mathbf{S}}$ from map $\mathbf{S}$ for distinguishing between chromatic and achromatic colors. The membership function of $\tilde{\mathbf{S}}$ is defined as

$$\mu_{\tilde{S}}(h) = \exp(-as). \tag{3}$$

This states that the smaller a given saturation value the more likely that it comes from some achromatic color.

3) $\tilde{\mathbf{I}}$ *Map:* While chromatic and achromatic colors can be separated from each other based on their saturation values, black and white have to be further differentiated from other achromatic colors. For this, we count on the intensity map $\mathbf{I}$. Since the intensity values of black and white correspond to the two extreme values on the intensity axis of the HSI coordinate system, the following function emphasizes the colors with intensity values close to the two extremes

$$f(i) = 1 - \exp[-a(i - 0.5)].$$

This assumes that the working environment has an average intensity of 0.5. However, both black and white will be distorted under some circumstances. For example, a white color may appear grey in a dark environment, as will a black color in a bright environment. To compensate for such distortion, the constant 0.5 in the above equation may be replaced with the average value, $\bar{i}$, of map $\mathbf{I}$. We then define the membership function of fuzzy map $\tilde{\mathbf{I}}$ as

$$\mu_{\tilde{I}}(i) = 1 - \exp(-a(i - \bar{i})). \tag{4}$$

4) $\tilde{\mathbf{E}}$ *Map:* Based on fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$ image areas with black, white, red, or green colors can be distinguished. However, a large portion of these areas has nothing to do with license plates. Edge maps play a crucial role in discriminating against irrelevant regions. Since there are many close-by edges in a license plate and distributed in a repetitive manner, an image pixel whose neighbors possess large edge magnitudes will have a high possibility that it belongs to a license plate. Hence, we define the membership function of fuzzy edge map $\tilde{\mathbf{E}}$ as

$$\mu_{\tilde{E}}(e_p) = \sum_{k \epsilon N_p} e_k \exp(-a d_{pk}) \tag{5}$$

where N_p is the horizontal neighborhood of the image pixel p under consideration, e_k is the edge magnitude of pixel k in N_p, and d_{pk} is the Euclidean distance between pixels p and k. In the above function we do not care about the edge magnitude e_p of pixel p itself.

D. Fuzzy Aggregation

Each fuzzy map provides information for locating license plates in the input image. There are two ways to draw a conclusion from a set of maps. In the first, intermediate decisions are made on the basis of individual maps and a final conclusion is drawn from the intermediate decisions. In the second, multiple maps are first integrated into a single map, and a final conclusion

is then drawn from the integrated map. Since the first method involves both numerical computations and symbolic decisions, we prefer the second approach, which includes only numerical computations. Following the second approach, fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, $\tilde{\mathbf{I}}$, and $\tilde{\mathbf{E}}$ are integrated into a single map, $\tilde{\mathbf{M}}$, with decisions being made on the basis of $\tilde{\mathbf{M}}$. A two-stage fuzzy aggregator is introduced for this purpose.

In the first stage of the aggregator, fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$ are integrated cell by cell. Let $\tilde{h}$, $\tilde{s}$, and $\tilde{i}$ be the entries of the corresponding cells in fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$. The aggregator integrates the entries by

$$\tilde{g} = u(w_h\tilde{h}, w_s\tilde{s}, w_i\tilde{i}) \tag{6}$$

where u is a fuzzy union operator and w_h, w_s, and w_i are weights reflecting the relative degrees of importance among fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$. The weights are described next.

Recall that in the definition of a fuzzy map a large entry indicates a high degree of possibility that the entry belongs to a license plate. However, if the majority of cells having small variations in the entry (i.e., having nearly uniform distribution of entries) are in a fuzzy map, the usefulness of the map for detecting license plates deteriorates. To see this, consider a picture taken in the evening or on a rainy day. On the whole, the picture will look dim. The overall intensities of image pixels tend to be small, which in turn leads to large saturation values throughout the picture [see (1)]. Both the intensity and saturation maps contribute little to locating license plates because entries are comparable. In all fuzzy maps, it is desirable that a relatively small portion of a map possesses large entries, while the remaining areas contain small values. Such a map will be given a large weight to reflect a high degree of importance of the map. Let $\tilde{A} = [\tilde{a}_{ij}]$ be any fuzzy map of size M by N. Its weight (or degree of importance) is then determined by

$$w_a = 1 - \frac{\left(\sum_{i=1}^{M}\sum_{j=1}^{N} b_{ij}\right)}{MN} \tag{7}$$

where

$$b_{ij} = \begin{cases} 1, & \text{if } \tilde{a}_{ij} \geq t \\ 0, & \text{otherwise} \end{cases}$$

in which threshold $t = (\tilde{a}_{\max} + \tilde{a}_{\min})/2$ and $\tilde{a}_{\max}$ and $\tilde{a}_{\min}$ are the maximum and minimum values in $\tilde{A}$

After combining fuzzy maps $\tilde{\mathbf{H}}$, $\tilde{\mathbf{S}}$, and $\tilde{\mathbf{I}}$, at the second stage the resulting map, say $\tilde{\mathbf{G}}$, and fuzzy map $\tilde{\mathbf{E}}$ are linearly combined into $\tilde{\mathbf{M}} = [\tilde{m}]$ by

$$\tilde{m} = w_g\tilde{g} + w_e\tilde{e}, \tag{8}$$

where w_g and w_e are the weights of maps $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{E}}$, which are determined similar to (7).

IV. License Number Identification Module

A. Fundamental Idea

Fig. 3 gives the flowchart for the identification module. There are two major components constituting the module, preprocessing and recognition. The preprocessing component consists of three tasks, binarization, connected component labelling, and noise removal, which are arranged in sequence. The recognition component is composed of two main procedures, character segmentation and recognition. To obtain optimal results for both the procedures, they are alternatively invoked.

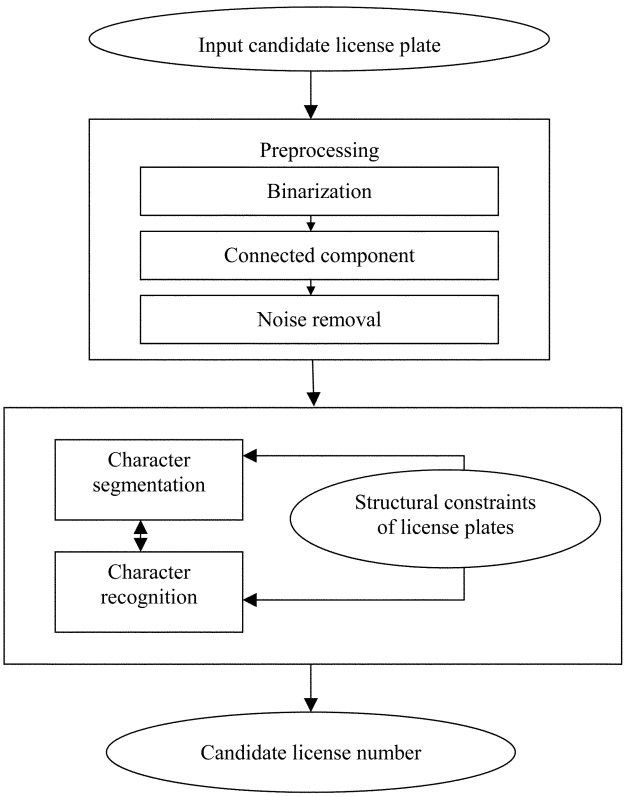

Fig. 3. Flowchart for the license number identification module.

Since the camera may be rolled and/or pitched with respect to license plates, it is desirable that their images be transformed to a predefined size and orientation before performing license number identification. However, without information about relationships between the camera and working environments, the transformations can only be conducted blindly or by trial-and-error. In the proposed method since the transformation step is omitted, it is inevitable that difficulties in the subsequent steps will increase.

Considering a license plate candidate, it is first binarized. Since some information will somehow be lost during binarization, a variable thresholding technique previously proposed by Nakagawa and Rosenfeld [24] is employed. The technique determines a local optimal threshold value for each image pixel so as to avoid the problem originating from nonuniform illumination. Although variable thresholding cannot completely compensate for the information loss mentioned above, it at least preserves information that may be lost when using a constant binarization method. There are two purposes for the binarization step: highlighting characters and suppressing background. However, both desired (e.g., characters) and undesired (e.g., noise and borders of vehicle plates) image areas often appear during binarization.

In order to eliminate undesired image areas, a connected component algorithm is first applied to the binarized plate candidate. The aspect ratios of connected components are then calculated. The components whose aspect ratios are outside a prescribed range are deleted. Then an alignment of the remaining components is derived by applying the Hough transform to the centers of gravity of components. The components disagreeing with the alignment are removed. If the number of remaining components is still larger than a prescribed number (eight in practice), connected components are deleted one at a time starting with the smallest. Here, we choose eight as the prescribed number because a license number consists of five or six characters and characters may be broken. The removal process continues until either of two conditions is satisfied. Either the number of remaining components equals the prescribed number, or a dramatic change in size from the previously removed component to the current one under consideration is encountered. We assume that noise components are much smaller than characters.

The above procedure does not guarantee that each of the surviving components will correspond to an individual character. A component may be due to noise, an incomplete character, a distorted character, or characters that appear to touch. To distinguish them, we utilize attributes of license plates, including the aspect ratios of individual characters, the regular intervals between characters, and the number of characters constituting license numbers. We refer to these attributes collectively as the structural constraints of license plates. We also introduce the operators of delete, merge, split and recover into the character segmentation procedure. Note that characters may be missed during license plate location and binarization. The recover operator is introduced to retrieve missing characters.

During processing the segmentation procedure applies the first three operators (delete, merge, and split) to the set of surviving components in an attempt to determine if a component satisfies the structural constraints of license plates. If such a component can be determined, the character recognition procedure is invoked to identify a character from the component. The above process repeats until no character can be extracted from the set of surviving components. Thereafter, if the number of extracted characters is less than the number of characters in license numbers, the recover operator starts at the outermost characters of those detected and searches for characters along the alignment of the known characters. The search continues until no character can be retrieved within an extent determined by the average width of characters as well as intervals between characters. Next, the collection of identified characters is verified in the confirmation stage, where the compositional semantics of license numbers plays an important role. The set of characters will be deemed to form a valid license number if it agrees with the compositional semantics.

B. Optical Character Recognition

In this subsection we discuss the character recognition procedure. Since, as already mentioned, license plates may be bent and/or tilted with respect to the camera, characters extracted from such license plates may be deformed. Furthermore, input characters may be noisy, broken or incomplete. Character recognition techniques should be able to tolerate these defects. In this study, we develop our own character recognition approach to suit our particular application. The proposed approach consists of three steps: character categorization, topological sorting, and self-organizing (SO) recognition. In the first step, the input character is distinguished as numerical or alphabetical. This is easily accomplished by referring to the compositional semantics of license numbers. In the next step, the topological features of the input character are computed and are compared with those of prestored character templates. Compatible templates will form a test set, in which the character template that best matches the input character is determined. The template test is performed by a SO character recognition procedure.

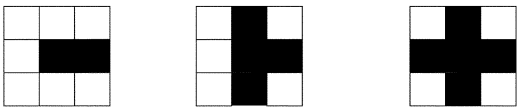

Fig. 4. Nodal types: (a) end-point, (b) three-way node, and (c) four-way node.

1) Topological Sorting: The topological features of characters utilized in this study include the number of holes, endpoints, three-way nodes, and four-way nodes (see Fig. 4 for their definitions). These features are invariant to spatial transformations (including rotation, translation and scale change). Moreover, these features, which are qualitative in nature, can be easily and reliably detected compared to quantitative features. However, input characters are usually imperfect; extra or missing features may occur. The following rule is employed for topological sorting. A character template is compatible with a given character whenever 1) their difference in the numbers of holes is within the range $[-1, 1]$ and 2) their difference between the numbers of nodes of any type is within the range $[-2, 2]$. Here, a smaller range is given to the hole feature because it is generally more reliable in detection than nodes. In our experiments no more than three out of ten numerical character templates and six out of 26 alphabetical character templates have passed topological sorting for any given character. This has greatly reduced the number of templates in the test set and hence the time complexity for character recognition.

2) Template Test: The templates in the test set are matched against the input character and the best match is determined. The template test is primarily accomplished using a SO character recognition approach, which is based on Kohonen's SO neural network [19]. The idea behind the proposed technique is as follows. Given an unknown character and a character template, the input character is encoded in terms of synaptic weights in the between-layer links of the neural network. The character template here serves as a stimulus, which repeatedly innervates the neural network, causing the synaptic weights of the neural network to gradually change. This process continues until the weights stabilize. We sum up the changes of synaptic weights during processing. The total change in weight in a sense reflects the level of dissimilarity between the unknown character and the character template.

Let $C = \{c_1, \ldots, c_L\}$ be the set of character templates surviving from the topological sorting of an unknown input character. Let $d_1, \ldots, d_L$ denote the computed dissimilarities between the unknown character and the character templates. It is natural that the character template having the smallest dissim-

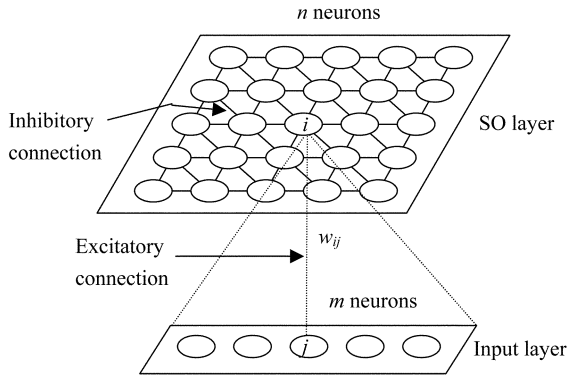

Fig. 5. Kohonen SO neural model.

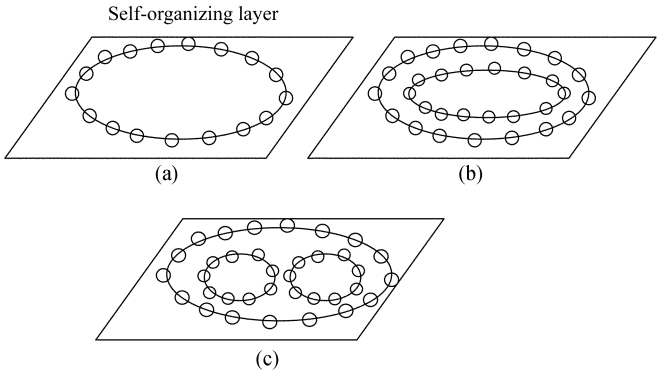

Fig. 6. SO layers: (a) 0-hole, (b) 1-hole, and (c) 2-hole SO layers.

ilarity with the unknown character is taken to be the class to which the unknown character belongs.

a) SO Neural Model: In this subsection, we brief the key components of the Kohonen SO neural model, which will be used in the later practical implementation. Referring to Fig. 5, the underlying configuration of the SO neural network consists of two layers, an input layer and an SO layer.

Let w_{ij} be the weight of the link between SO neuron n_i and input neuron n_j. The weight vector for n_i is $\mathbf{w}_i = (w_{i1}, w_{i2}, \ldots, w_{im})$, where m is the number of input neurons. Let $\mathbf{v}_i = (v_1, v_2, \ldots, v_m)$ denote an external stimulus. The input to n_i due to the stimulus is

$$I_i^s = \mathbf{W}_i \cdot \mathbf{V} = \sum_{k=1}^{m} w_{ik} v_k. \tag{9}$$

The lateral interaction among SO neurons is characterized by a "Mexican-hat" function [21], denoted $h(\mathbf{r})$, where $\mathbf{r}$ represents a position vector. Let u_{ik} be the weight of the connection between SO neurons n_i and n_k located at $\mathbf{r}_i$ and $\mathbf{r}_k$, respectively. The input to n_i due to lateral interaction is

$$I_i^l = \sum_{n_k \epsilon N, k \neq i} a_k \mathbf{u}_{ik} h(\mathbf{r}_k - \mathbf{r}_i) \tag{10}$$

where N is the set of SO neurons and $a_k = \psi(net_k)$ is the activation of n_k, in which net_k to be defined is the net input to n_k and ψ is the output function defined by a sigmoid function.

A leakage term $e(a)$, which dissipates activations of SO neurons once a stimulus has been removed, is introduced for every SO neuron. The net input to n_i then sums the inputs from the stimulus, lateral interaction, and leakage

$$net_i = I_i^s + I_i^l + e(a_i). \tag{11}$$

During competition among SO neurons, the winner n_c is determined by $net_c = \max_{1 \leq i \leq n} \{net_i\}$. Next, the winner together with its neighbors, say set N_c, engage in a group learning process. During this process a neuron close to the winner will gain a high rate of learning while a neuron located far from the winner will have a low rate of learning. The rate of learning is governed by the Gaussian function g. The learning rule for the neurons in N_c is then defined as

$$\Delta \mathbf{w}_k = (\mathbf{v} - \mathbf{w}_k) g(\mathbf{r}_k - \mathbf{r}_c), \qquad k \epsilon N_c. \tag{12}$$

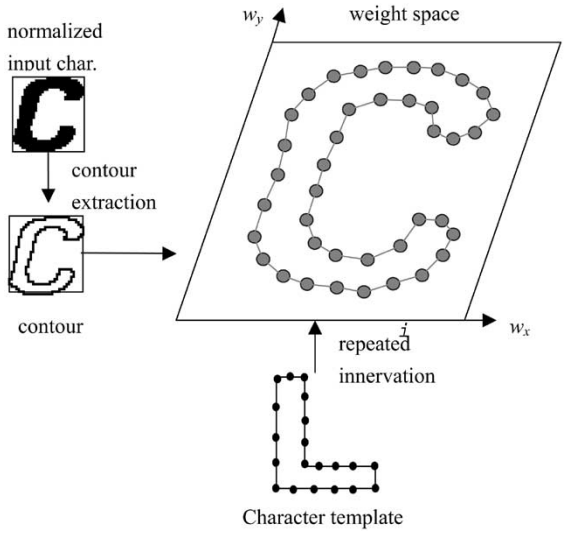

Fig. 7. Example of SO character recognition.

Finally, the updating equation for SO neuron n_k is

$$\mathbf{w}_k^{t+1} = \mathbf{w}_k^t + \rho(t) \Delta \mathbf{w}_k^t, \qquad k \epsilon N_c \tag{13}$$

where $\rho(t)$ is the step size, which decreases monotonically with increasing t.

b) Practical Implementation: To begin, we group characters into three categories, referred to as 0-hole, 1-hole, and 2-hole, according to the number of holes contained in the characters. Each category has its own associated SO neural network, which contains 40 SO neurons and two input neurons. The difference among the three neural networks is primarily in their configurations of SO layer (see Fig. 6).

Referring to the example shown in Fig. 7, suppose that we are given an unknown character ("C" in this example). The character is normalized in size (16 by 16 pixels) in order to be consistent with the character templates. The number of holes in the character is computed. Here, we always choose the neural network according to the computed number regardless of whether the computed number is the true number of holes for that character. Since the input character ("C") has no hole, the neural network with the 0-hole SO layer is chosen. Next, the contour and its length of the unknown character are found. The length is divided into 40 approximately equally spaced intervals. Starting at any arbitrary point along the contour, the two dimensional

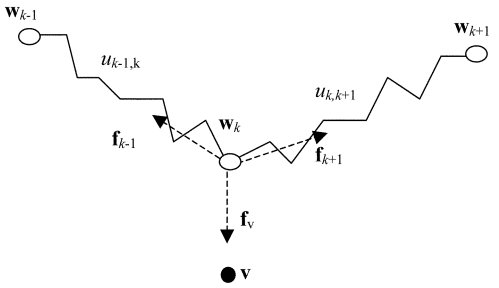

Fig. 8. Spring model.

(2-D) position vectors, i.e., the (row, column) coordinates, of the 40 interval boundaries are extracted. See the right side of Fig. 7. We choose 40 points because they are about half the average number of contour points in the character templates. The position vectors of the 40 contour points are then assigned, one by one in the order of extraction, to the weight vectors of the 40 SO neurons of the chosen neural network. Note that since all the configurations of SO layer of the three neural networks are symmetrical which SO neuron should be assigned first is not important.

Consider a 2-D space with axes corresponding to the two components of weight vectors of SO neurons. The weight vector of each SO neuron can be represented as a point in the space. This space is referred to as the weight space of the neural network. Since the weight vectors of SO neurons are set to the position vectors of contour points of the input character, the contour will be recreated in the weight space when we represent the weight vectors of SO neurons as points in the weight space. See the picture on the right hand side of Fig. 7.

Suppose that a template ("L" in the example) chosen from the test set for the input character is to be matched against the character. Rather than the entire template, just its contour serves as the stimulating pattern, which repeatedly innervates the neural network until its synaptic weights stabilize. In our implementation the contour points are fed into the input layer of neural network one at a time. Consider an input contour stimulus point $\mathbf{v}$. The SO neurons of the network compete for the stimulus. The winner n_c is determined by $n_c = \arg\left(\max_{1\le i\le 40}\{\mathbf{w}_i \cdot \mathbf{v}\}\right)$, where $\mathbf{w}_i$'s are the weight vectors of the 40 SO neurons. The winner and its first- and second-order neighbors, call them set N_c, join in the following learning process

$$\Delta \mathbf{w}_k = \mathbf{d}_k g(\mathbf{r}_k - \mathbf{r}_c), \qquad k \epsilon N_c. \tag{14}$$

Note that $(\mathbf{v} - \mathbf{w}_k)$ in (12) has been replaced by $\mathbf{d}_k$ in (14). The $\mathbf{d}_k$ is computed as follows. We use a model, called the spring model, taken from [4], in which SO neurons are assumed being connected with springs. The elastic spring coefficients are simulated with synaptic weights between neurons. Referring to Fig. 8, we denote the SO neurons with their weight vectors. Weight vectors $\mathbf{w}_{k-1}$, $\mathbf{w}_k$, and $\mathbf{w}_{k+1}$ represent SO neurons n_{k-1}, n_k, and n_{k+1}, respectively, where n_k is any learner in N_c and n_{k-1} and n_{k+1} are the two first-order neighbors of n_k. The learner is connected to its two neighbors with the springs, whose coefficients are $u_{k-1,k}$ and $u_{k,k+1}$. The point stimulus is denoted $\mathbf{v}$ in the figure.

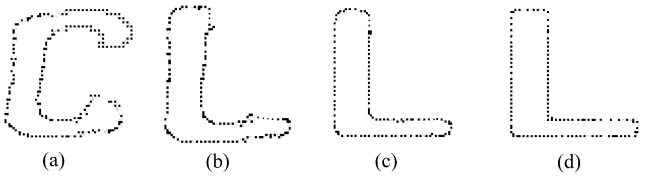

Fig. 9. Some intermediate results of shape transformation. (a) 1st iteration. (b) 5th iteration. (c) 10th iteration. (d) 42nd iteration.

The $\mathbf{v}$ here serves as an attractive source, which during the learning process attempts to pull $\mathbf{w}_k$ toward it with force $\mathbf{f}_v$. However, the springs exerting forces $\mathbf{f}_{k-1}$ and $\mathbf{f}_{k+1}$ on $\mathbf{w}_k$ try to pull $\mathbf{w}_k$ toward its neighbors $\mathbf{w}_{k-1}$ and $\mathbf{w}_{k+1}$. In addition, there is a damping force, $\mathbf{f}_d$ (not shown in the figure), for dissipating the energy in the spring model so that the entire system will eventually converge to an equilibrium state with an external force $\mathbf{f}_e$, i.e.

$$\mathbf{f}_v + \mathbf{f}_{k-1} + \mathbf{f}_{k+1} + \mathbf{f}_d = \mathbf{f}_e \tag{15}$$

where

$$\begin{aligned}
\mathbf{f}_v &= \frac{k_a}{(\|\mathbf{v} - \mathbf{w}_k\| + \varepsilon)^2} \frac{\mathbf{v} - \mathbf{w}_k}{\|\mathbf{v} - \mathbf{w}_k\|} \\
\mathbf{f}_{k-1} &= u_{k-1,k} \left(\|\mathbf{w}_{k-1} - \mathbf{w}_k\| - l\right) \frac{\mathbf{w}_{k-1} - \mathbf{w}_k}{\|\mathbf{w}_{k-1} - \mathbf{w}_k\|} \\
\mathbf{f}_{k+1} &= u_{k,k+1} \left(\|\mathbf{w}_k - \mathbf{w}_{k+1}\| - l\right) \frac{\mathbf{w}_k - \mathbf{w}_{k+1}}{\|\mathbf{w}_k - \mathbf{w}_{k+1}\|} \\
\mathbf{f}_d &= -k_d \|\mathbf{f}_v + \mathbf{f}_{k-1} + \mathbf{f}_{k+1}\|^{1/2} \frac{\mathbf{f}_v + \mathbf{f}_{k-1} + \mathbf{f}_{k+1}}{\|\mathbf{f}_v + \mathbf{f}_{k-1} + \mathbf{f}_{k+1}\|}
\end{aligned} \tag{16}$$

in which k_a is the gravitational coefficient, k_d is the damping coefficient, l is the natural length of the springs, and ε is a small value to prevent $\mathbf{f}_v$ from becoming infinite as $\mathbf{w}_k$ approaches $\mathbf{v}$. Substituting the forces in (16) into (15) for $\mathbf{f}_e$, the displacement of neuron n_k is

$$\mathbf{d}_k = \mathbf{v}_{0k} + \frac{f_e}{m} \Delta t^2. \tag{17}$$

For simplicity, we assume that the initial velocity of neuron n_k, $\mathbf{v}_{0k}$, is zero, the neural mass m is one, and the time interval Δt is one. The result is

$$\mathbf{d}_k = \mathbf{f}_e = \mathbf{f}_v + \mathbf{f}_{k-1} + \mathbf{f}_{k+1} + \mathbf{f}_d. \tag{18}$$

Note that the calculated displacement $\mathbf{d}_k$ has to be modified by the degree of learning of neuron n_k, $g(\mathbf{r}_k - \mathbf{r}_c)$, and a learning step $\rho(t)$. The actual displacement of neuron n_k in the weight space is $\rho(t)\mathbf{d}_k g(\mathbf{r}_k - \mathbf{r}_c)$.

Accumulating the displacements of all the neurons in N_c accomplishes the innervation of point stimulus $\mathbf{v}$. Repeating the above process for all point stimuli of the input stimulus pattern completes one iteration for the pattern. Iterating continues until no significant displacement of SO neuron is observed (i.e., stabilized). The total displacement of SO neurons serves as a measure of dissimilarity between the unknown character and the character template. Fig. 9 illustrates some intermediate results of shape transformation from "C" to "L" during iteration. In this example the total displacement of neurons amounts to 147 pixels. Empirically, displacements have been distributed

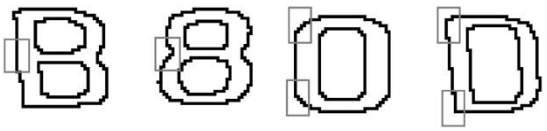

Fig. 10. Distinguishing parts of ambiguous characters.

over the interval $[23, 67]$ when correct templates were chosen for testing.

c) Remarks: The proposed character recognition approach has difficulty distinguishing character pairs (8, B) and (O, D) especially when they are distorted. To overcome this, we predefine an ambiguity set containing the characters 0, 8, B and D. For each character in the set, the nonambiguous parts of the character are specified (see Fig. 10). During character recognition, once an unknown character is classed as one of the characters in the ambiguity set, an additional minor comparison between the unknown character and the classed character is performed. The comparison focuses on only the nonambiguous parts of the character.

Our character recognition method gives different measurements of dissimilarity for the same character with different tilt angles with respect to the camera. Currently, this issue has not troubled us because the characters extracted from the images of license plates are all in a nearly upright position. But, we may improve our algorithm to deal with this by introducing a normalization step to transform license plates into a prescribed orientation prior to license number identification.

V. Experimental Results

Two groups of images have been collected for our experiments. The first contains 639 images (640 by 480 pixels) taken from 71 cars of different classes. For each car, nine images were acquired from fixed viewpoints whose positions are illustrated in Fig. 11(a). Fig. 11(b) shows two images of one car taken from viewpoints a_1 and c_3. The experimental results with the first group of images are summarized in Table II. In this table columns correspond to viewpoints, rows to the classes of vehicle (or the types of license plate), and the entries are the number of correctly located license plates. The percent of correctly located license plates (the success rate) is given in the bottom row of the table. The success rates for viewpoints $a1$, $a2$, and $a3$ (i.e., straight on) are 100%, independent of the type of license plate and viewing distance. However, as the viewing angle increases the success rate declines. In the worst case, viewpoints $c1$, $c2$, and $c3$, the success rates are 97.2%, 98.6%, and 95.8%, respectively. The overall average success rate with the first group of images is 98.8%.

The second group contains 449 images (768 by 512 pixels), some of which are shown in Fig. 12. The images are taken from (a) complex scenes, in which several objects look like license plates, (b) various environments (street, roadside and parking lot), (c) different illumination (dawn, sunshine, rain, back lighting, shadow, and nighttime), and (d) damaged license plates (such as being bent). In these images, all the license plates were successfully located as shown by the bounding boxes. Fig. 13 shows two images in which our locating module failed to detect license plates. In the example on the left, the

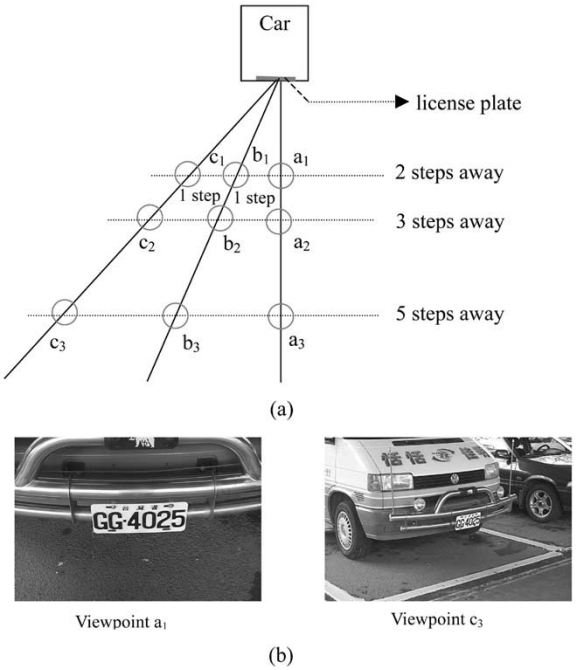

Fig. 11. (a) Nine different viewpoints for the first group of images. The length of one step is about 60 cm. (b) Two images of a car taken from viewpoints a_1 and c_3.

TABLE II
Experimental Results With the First Group of Images

Viewpoint / #vehicles / Vehicle class	a_1	a_2	a_3	b_1	b_2	b_3	c_1	c_2	c_3
Private automobile	30	30	30	30	30	30	30	30	29
Taxi	20	20	20	19	20	20	19	19	20
Tour bus	7	7	7	7	7	7	6	7	7
Truck	8	8	8	8	8	7	8	8	6
Government vehicle	6	6	6	6	6	6	6	6	6
Success rate(%)	100	100	100	98.6	100	98.6	97.2	98.6	95.8

Average success rate (%): 98.8

actual license plate is connected with the reflection of a license plate-like image on the front of another car. The area that includes the license plate and the reflection was not consistent with the constraint on aspect ratio for license plates. In the example on the right, the license plate is surrounded by a region, which possesses features (including color and edges) resembling those of actual license plates. The whole area containing the license plate and the surrounding region was regarded by the locating module as a potential license plate. However, since the aspect ratio of the area was out of range, the locating module rejected it. The above two examples are actually unusual cases that are rarely encountered.

A common failure of the locating module is the failure to detect the boundaries of license plates. This occurs when vehicle bodies and their license plates have similar colors. For this, boundaries of license numbers were often extracted instead of the entire license plates. License plate images may be kept for further testing if the computed aspect ratio of the rectangular area containing the license numbers agrees with the constraint. The location success rate achieved with the second group of images (449 images) is 96.7%. Combining this rate with the rate

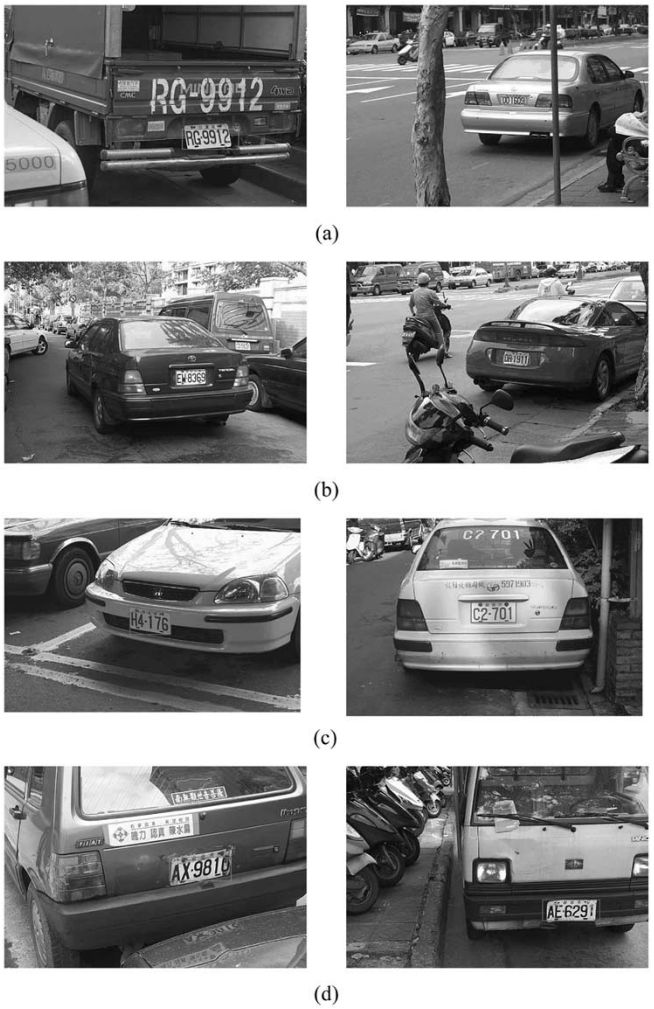

Fig. 12. Example images in the second group: (a) Complex scenes. (b) Various environments. (c) Different illuminations. (d). Damaged license plates.

Fig. 13. Examples of failures to find license plates.

(98.8%) of the first group (639 images), the overall rate of success for the license plate locating module is 97.6%.

The license plates (1061 plates in all), whose dimensions range from (320×95) to (80×45) pixels, extracted in the previous experiment were used in the experiment for license number identification. The images of license plates are rarely perfect (see Fig. 14); views may be skewed, or they may be over-segmented or under-segmented. The first two cases have not bothered the license number identification module. However, a significant degradation in performance of the module is due to under-segmentation [see the example shown in Fig. 14(c)]. In this case, problems with incomplete and missing characters may occur. On the other hand, as Fig. 15 demonstrates, even under-segmented plates can be correctly

Fig. 14. Examples of imperfect license plate location: (a) skewed view, (b) over-segmentation, and (c) under-segmentation.

Fig. 15. Examples of successful license number identification: (a) located license plates, (b) character segmentation, and (c) license number identification.

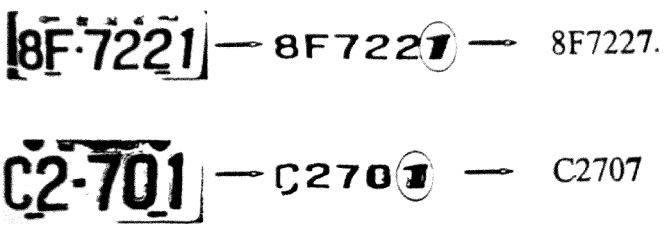

Fig. 16. Examples of failed license number identification.

recognized. The figure shows the extracted license plates, the results of character segmentation, and the identified license numbers.

Another problem which often caused trouble for the identification module is the misidentification of numeral "1" as "7" (see Fig. 16). This is essentially caused by the fact that the module recognizes characters based on their contours and that contours preserve only partial information about object shapes. However, the reverse of misidentifying "7" as "1" never occurred in the experiment. We refer to these collectively as the "1&7 confusion." This is intrinsically different from an ambiguity mentioned earlier and cannot be solved by means of the ambiguity set. However, it may be resolved based on the observation that both templates "1" and "7" will be in the test set if the input character is either "1" or "7."

The success rate for identification with the set of 1061 license plates is 95.6%. Combining this rate with the location success rate (97.9%), the overall rate of success for our LPR algorithm is 93.7%.

Currently, our LPR algorithm is running on a Pentium IV-1.6 GHz PC. The license plate location module takes about 0.4 seconds to find all license plate candidates in an image. However,

the license number identification module takes about two seconds (primarily due to the neural-based OCR process running on a sequential computer) to process one license plate candidate. If there are multiple plate candidates located in an image, the total processing time may not be adequate for real-time applications. Several things may be considered to compensate for the high time complexity. Firstly, since the proposed technique assumes no prior information about license numbers, if such information is available (e.g., databases of license numbers), both the processing time and the recognition rate will improve. Secondly, if parallel machines, such as transputers, *n*-cubes, or PC clusters, can be used, multiple license plate candidates and character templates will be able to be processed in parallel. As a consequence, the topological sorting step becomes unnecessary. Finally, some firmware components of the algorithm could be replaced by hardware.

VI. Concluding Remarks and Future Work

Compared to most previous work that in some way restricted their working conditions, the techniques presented in this paper are much less restrictive. The proposed LPR algorithm consists of two modules, one for locating license plates and one for identifying license numbers. Soft computing techniques rooted in fuzzy (for license plate location) and neural (for license number identification) disciplines were introduced to compensate for uncertainties caused by noise, measurement error and imperfect processing. Although the proposed algorithm is concerned with the license plates of one specific country, many parts in the algorithm are readily extended to use with license plates of other countries. Specifically, since color and edge are two fundamental features of license plates, the color edge detector introduced in the locating module is readily adapted to other color schemes by replacing the color parameters embedded in the detector. Since numerals and Roman letters are commonly used to form license numbers, the proposed SO OCR technique is applicable to any similarly constituted license plates.

It is well known that a mixture of top-down (expectation-driven) and bottom-up (data-driven) procedures often perform better than either in isolation. Currently, the locating and identification modules both perform in somewhat of a hybrid top-down and bottom-up manner. Location determination is guided by both the color information of license plates and the compositional semantics of license numbers, while identification is based on prebuilt templates and the compositional semantics. A higher degree of combining top-down with bottom-up processing may be used in some applications, such as the control of restricted or secure areas, the detection of stolen vehicles, and the management of car pools, where license information of the cars of interest can be known a priori.

In our future work, techniques for deriving intrinsic images (e.g., illumination, reflectance and depth images) from a scene image or a number of input images are recommended. Intrinsic images containing only one intrinsic characteristic of the scene are viewpoint dependent and can be of great use for many visual inferences, such as image segmentation, view-based template matching, and object reconstruction. Recall that at the identification stage we have omitted a normalization step to transform extracted license plates to a prescribed size and orientation. Adding this step would improve the performance of license number identification. However, normalization requires knowing the boundaries of either license plates or license numbers. The former may be invisible if vehicle bodies and license plates have similar colors, while detecting boundaries of license numbers can be error-prone. We leave these issues to be considered in future study. Furthermore, the proposed neural approach for character recognition is basically unsupervised. In general, supervised methods can outperform unsupervised ones if rich training sets are available. We may later investigate supervised approaches.

A number of strategies have been introduced to reduce the time complexity of the proposed LPR algorithm. The color edge detector reduces the processing time by ignoring irrelevant edges at an early stage; the topological sorter limits the set of template candidates for character test at the identification stage. Obviously, there are more things that can be done to improve the processing time. However, in order to make our techniques applicable to real-time applications in less restrictive working conditions, the topics regarding replacing firmware components with hard-wired ones and using parallel machines should be studied.

Acknowledgment

The authors gratefully acknowledge the assistance of Prof. R. R. Bailey of the National Taiwan Normal University, Taipei, Taiwan, R.O.C., for his helpful comments and for editing the English of this paper.

References

[1] G. Adorni, F. Bergenti, and S. Cagnoni, "Vehicle license plate recognition by means of cellular automata," in *Proc. IEEE Int. Conf. Intelligent Vehicles*, 1998, pp. 689–693.

[2] M. H. T. Brugge, J. H. Stevens, J. A. G. Nijhuis, and L. Spaanenburg, "License plate recognition using DTCNNs," in *Proc. 5th IEEE Int. Workshop on Cellular Neural Networks and Their Applications*, 1998, pp. 212–217.

[3] K. R. Castleman, *Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1996, pp. 550–554.

[4] S. W. Chen, G. C. Stockman, and K. E. Chang, "SO dynamic deformation for building of 3-D models," *IEEE Trans. Neural Networks*, vol. 7, pp. 374–387, June 1996.

[5] J. R. Cowell, "Syntactic pattern recognizer for vehicle identification numbers," *Image and Vision Comput.*, vol. 13, no. 1, pp. 13–19, 1995.

[6] Y. Cui and Q. Huang, "Character extraction of license plates from video," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1997, pp. 502–507.

[7] P. Davies, N. Emmott, and N. Ayland, "License plate recognition technology for toll violation enforcement," *Inst. Elect. Eng. Colloquium Image Analysis for Transport Applications*, pp. 7/1–7/5, 1990.

[8] S. Draghici, "A neural network based artificial vision system for license plate recognition," *Int. J. Neural Systems*, vol. 8, pp. 113–126, 1997.

[9] D. M. Emiris and D. E. Koulouriotis, "Automated optic recognition of alphanumeric content in car license plates in a semi-structured environment," in *Proc. Int. Conf. Image Processing*, vol. 3, 2001, pp. 50–53.

[10] D. S. Gao and J. Zhou, "Car license plates detection from complex scene," in *Proc. 5th Int. Conf. Signal Processing*, vol. 2, 2000, pp. 1409–1414.

[11] H. A. Hegt, R. J. De la Haye, and N. A. Khan, "A high performance license plate recognition system," in *Proc. IEEE Int. Conf. System, Man, and Cybernetics*, vol. 5, 1998, pp. 4357–4362.

[12] X. F. Hermida, F. M. Rodriguez, J. L. F. Lijo, F. P. Sande, and M. P. Iglesias, "A system for the automatic and real time recognition of VLP's (Vehicle License Plate)," in *Proc. Lecture Notes in Computer Science*, 1997, vol. 1311, pp. 552–558.

[13] H. Hontani and T. Koga, "Character extraction method without prior knowledge on size and position information," in *Proc. IEEE Int. Conf. Vehicle Electronics*, 2001, pp. 67–72.
[14] J. M. Keller and R. Krishnapuram, "Fuzzy decision models in computer vision," in *Fuzzy Sets, Neural Networks, and Soft Computing*, R. R. Yager and L. A. Zadeh, Eds. New York: Van Nostrand, 1994, pp. 213–232.
[15] D. S. Kim and S. I. Chien, "Automatic car license plate extraction using modified generalized symmetry transform and image warping," in *Proc. IEEE Int. Symp. Industrial Electronics*, vol. 3, 2001, pp. 2022–2027.
[16] K. K. Kim, K. I. Kim, J. B. Kim, and H. J. Kim, "Learning-based approach for license plate recognition," in *Proc. IEEE Signal Processing Society Workshop*, vol. 2, 2000, pp. 614–623.
[17] S. K. Kim, D. W. Kim, and H. J. Kim, "A recognition of vehicle license plate using a genetic algorithm based segmentation," in *Proc. Int. Conf. Image Processing*, vol. 2, 1996, pp. 661–664.
[18] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic, Theory and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
[19] T. Kohonen, *Self-Organization and Aassociative Memory*. New York: Springer-Verlag, 1989.
[20] R. A. Lotufo, A. D. Morgan, and A. S. Johnson, "Automatic number-plate recognition," *Inst. Elect. Eng. Colloquium on Image Analysis for Transport Applications*, pp. 6/1–6/6, 1990.
[21] D. Marr, *Vision*. New York: Freeman, 1982.
[22] K. Miyamoto, K. Nagano, M. Tamagawa, I. Fujita, and M. Yamamoto, "Vehicle license plate recognition by image analysis," in *Proc. Int. Conf. Industrial Electronics, Control and Instrumentation*, 1991, pp. 1734–1738.
[23] T. Naito, T. Tsukada, K. Yamada, K. Kozuka, and S. Yamamoto, "Robust license-plate recognition method for passing vehicles under outside environment," *IEEE Trans. Veh. Technol.*, vol. 49, pp. 2309–2319, Nov. 2000.
[24] Y. Nakagawa and A. Rosenfeld, "Some experiments on variable thresholding," *Pattern Recognition*, vol. 11, no. 3, pp. 191–204, 1979.
[25] J. A. G. Nijhuis, M. H. T. Brugge, K. A. Helmholt, J. P. W. Pluim, L. Spaanenburg, R. S. Venema, and M. A. Westenberg, "Car license plate recognition with neural networks and fuzzy logic," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 5, 1995, pp. 2232–2236.
[26] R. Parisi, E. D. D. Claudio, G. Lucarelli, and G. Orlandi, "Car plate recognition by neural networks and image processing," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 3, 1998, pp. 195–198.
[27] S. W. Perry, H. S. Wong, and L. Guan, *Adaptive Image Processing, A Computational Intelligence Perspective*. Boca Raton, FL: CRC, 2002.
[28] J. C. H. Poon, M. Ghadiali, G. M. T. Mao, and L. M. Sheung, "A robust vision system for vehicle license plate recognition using grey-scale morphology," in *Proc. IEEE Int. Symp. Industrial Electronics*, vol. 1, 1995, pp. 394–399.
[29] H. Ritter, T. Martinetz, and K. Schulten, *Neural Computation and Self-organizing Maps an Introduction*. New York: Addison-Wesley, 1992.
[30] L. Salgado, J. M. Menendez, E. Rendon, and N. Garcia, "Automatic car plate detection and recognition through intelligent vision engineering," in *Proc. IEEE Int. Carnahan Conf. Security Technology*, 1999, pp. 71–76.
[31] T. Sirithinaphong and K. Chamnongthai, "The recognition of car license plate for automatic parking system," in *Proc. 5th Int. Symp. Signal Processing and its Applications*, 1998, pp. 455–457.
[32] Y. S. Soh, B. T. Chun, and H. S. Yoon, "Design of real time vehicle identification system," in *Proc. IEEE Int. Conf. System, Man, and Cybernetics: Humans, Information, and Technology*, vol. 3, 1994, pp. 2147–2152.
[33] K. Yamaguchi, Y. Nagaya, K. Ueda, H. Nemoto, and M. Nakagawa, "A method for iidentifying specific vehicles using template matching," in *Proc. IEEE Int. Conf. Intelligent Transportation Systems*, 1999, pp. 8–13.
[34] M. Yu and Y. D. Kim, "An approach to Korean license plate recognition based on vertical edge matching," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, 2000, pp. 2975–2980.
[35] N. H. C. Yung, K. H. Au, and A. H. S. Lai, "Recognition of vehicle registration mark on moving vehicles in an outdoor environment," in *Proc. IEEE Int. Conf. Intelligent Transportation Systems*, 1999, pp. 418–422.

Shyang-Lih Chang was born in Taipei, Taiwan, R.O.C., in 1962. He received the B.S. degree in electronics engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C., in 1987 and the M.S. degree from the Department of Computer Science and Information Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., in 1990. He is currently working toward the Ph.D. degree at the Department of Information and Computer Education, National Taiwan Normal University, Taipei, Taiwan, R.O.C.

Since August 1993, he has been a Lecturer at the Department of Electronics Engineering, St. John's and St. Mary's Institute of Technology, Taiwan, R.O.C. His research interests include image processing, computer vision, and the design of microprocessor systems.

Li-Shien Chen received the B.Sc. and M.Sc. degrees in information and computer education from National Taiwan Normal University, Taiwan, R.O.C., in 1998 and 2000, respectively.

Currently, she is a Teacher at the Department of Data Processing, National Ilan Commercial Vocational High School, Ilan, Taiwan, R.O.C. Her areas of research interest include fuzzy systems, neural networks, computer vision, and image processing.

Yun-Chung Chung received the B.E. degree in naval architecture and ocean engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 1993 and the M.E. degree in information and computer education from National Taiwan Normal University, Taipei, Taiwan, R.O.C., in 1995, where he is currently working toward the Ph.D. degree in the Department of Information and Computer Education.

From 1997 to 1998, he worked as an Associate Engineer at the Information Science and Technology Exhibition Center, Institute for Information Industry, Taipei, Taiwan. He is currently a Teacher in the Department of Data Processing, Taipei Municipal Shihlin Commercial High School, Taipei, Taiwan, R.O.C. His research interests include traffic monitoring system, visual tracking, and surveillance and digital image processing.

Sei-Wan Chen (S'85–M'89–SM'97) received the B.Sc. degree in atmospheric and space physics and the M.Sc. degree in geophysics from National Central University, Taiwan, R.O.C., in 1974 and 1976, respectively, and the M.Sc. and Ph.D. degrees in computer science and engineering from Michigan State University, East Lansing, in 1985 and 1989, respectively.

From 1977 to 1983, he worked as a Research Assistant in the Computer Center of Central Weather Bureau, Taiwan, Republic of China. In 1990, he was a Researcher in the Advanced Technology Center, Computer and Communication Laboratories, Industrial Technology Research Institute, Hsinchu, Taiwan, R.O.C. From 1991 to 1994, he was an Associate Professor in the Department of Information and Computer Education, National Taiwan Normal University, Taipei, Taiwan, R.O.C., where from 1995 to 2001, he was a Full Professor. Currently, he is a Professor at the Graduate Institute of Computer Science and Information Engineering at the same university. His areas of research interest include neural networks, fuzzy systems, pattern recognition, image processing, and computer vision.