License Plate Recognition System Using Artificial Neural Networks

İbrahim Türkyılmaz and Kirami Kaçan

A high performance license plate recognition system (LPRS) is proposed in this work. The proposed LPRS is composed of the following three main stages: (i) plate region determination, (ii) character segmentation, and (iii) character recognition. During the plate region determination stage, the image is enhanced by image processing algorithms to increase system performance. The rectangular license plate region is obtained using edge-based image processing methods on the binarized image. With the help of skew correction, the plate region is prepared for the character segmentation stage. Characters are separated from each other using vertical projections on the plate region. Segmented characters are prepared for the character recognition stage by a thinning process. At the character recognition stage, a three-layer feedforward artificial neural network using a backpropagation learning algorithm is constructed and the characters are determined.

Keywords: Image processing, Character segmentation, Character recognition, Artificial neural network, License plate recognition.

I. Introduction

Information technology is rapidly advancing with respect to automated systems. In such systems, people utilize computer-based expert systems to analyze and handle real-life problems such as intelligent transportation systems. In daily life, license plate recognition (LPR) is used widely in various fields such as vehicle tracking, traffic monitoring, automatic payment of tolls on highways or bridges, surveillance systems, toll collection points, and parking management systems. Vehicles are recognized by their license plate (LP) numbers, which are assigned by the authorities. Until recently, vehicle LPs have been read, interpreted, and used for various operations by different authorities. Although human observation appears to be the easiest way to read a vehicle's LP, reading error due to tiredness is a major weakness of manual systems. The development of fast and effective automated systems that eliminate such errors caused by human factors and produce consistent results are needed. Nowadays, LPR is a key technique in many automated transportation systems. LPR is simply a pattern recognition technique to recognize the LPs of vehicles. In fact, it uses a sequence of operations for locating a vehicle's LP inside an image and reading characters on the LP. An LPR system (LPRS) usually consists of three main stages:

1) *Plate Region Determination*: detection of the rectangular LP region and extraction from the image.

2) *Character Segmentation*: separation of the characters on the LP region for the character recognition stage.

3) *Character Recognition*: conversion of image-based characters into a textual expression.

For the proposed LPRS, image processing techniques are employed in the first two stages, and in the final stage, an artificial neural network (ANN) is used. Many methods have been developed for these stages with different advantages and disadvantages. The success of the developed methods is based

Manuscript received Aug. 26, 2015; revised Apr. 21, 2016; accepted June 7, 2016.

İbrahim Türkyılmaz (corresponding author, iturkyilmaz@comu.edu.tr) is with the Department of Computer Engineering, Çanakkale Onsekiz Mart University, Turkey.

Kirami Kaçan (kirami.kacan@ufukyazilim.com) is with Software Development, Ufuk Bilim Yazılım Teknoloji, Ankara, Turkey.

https://doi.org/10.4218/etrij.17.0115.0766

on their robustness to environmental conditions such as lighting, complex backgrounds, defects on the plate surface (damaged or dirty plates), and a range of distances and viewpoints between the vehicle and camera. Most LPRSs can work effectively under conditions such as constant backgrounds, controlled lightning, a predetermined camera-car distance and angle, and limited plate types. The aim of this work is to develop an LPRS that takes less processing time, uses low computing power, and has better recognition rates under fewer restrictions using ANNs and various image processing techniques.

The rest of this paper is organized as follows. The next section reviews previous LPRSs. An improved edge-based technique for LP region determination is presented in Section III. A modified character segmentation method is presented in Section IV. A character thinning algorithm analyzing the predominant structural features separating the characters is presented in Section V. In Section VI, an ANN is designed for character recognition and the results from this ANN are presented. In Section VII, the GUI of the LPRS is presented. Section VIII concludes the paper.

II. Previous Work

Various methods have been developed and implemented for each LPRS stage. Thus, each method must be examined separately.

There are many methods that are used for LP region determination, and they can be grouped into four categories: color-, gray-level-, binary-, and classifier-based methods [1], [2]. Color-based methods used in LP region determination use color criteria. Color-based criteria for human object recognition are very powerful, but the color features in an image may have different values because of various lighting conditions and the quality of image acquisition systems. Such color values, which can represent different features under different conditions, can adversely affect system performance. Hence, the use of color-based methods has become very limited [3]. These methods include known color model transformations. Shi and others [3] determined the plate region by categorizing the color combinations of Chinese LPs. Zimic and others [4] used fuzzy set theory and identified the LP area by identifying fuzzy rules based on a human's color perception.

The gray-level methods are preferable and more widely used than color-based methods because they perform better. However, in some cases, they have disadvantages such as high processing loads and long calculation times. Thus, it is almost impossible to obtain significant characters from images at high resolutions. Many methods have been developed using vector quantization, the Hough transform, Gabor transform, and wavelet analysis in this category. Zunino and Rovetta [5] noted that there are many regions with high densities of dark pixels on a white background in the LP region and identified it using adaptive vector quantization. Duan and others [6] found plate boundaries by applying the Hough transform on image edges. Kahraman and others [7] determined the plate region using Gabor filters and vector quantization using a binary tree decomposition. Hsieh and others [8] determined the plate region by identifying image contrast using wavelet analysis.

In classifier-based methods, the system is trained with rectangular plate samples and the coordinates of a new LP region are determined by testing the new image [9]. Large plate regions increase the computing time and hence such systems are slow [10], Genetic algorithms and ANNs are also examples of methods in this category. Nijhuis and others [11] recognized LPs using fuzzy logic and ANNs. Kim and others [12] and Xiong and others [13] used genetic algorithms to determine the LP region.

In addition to the above methods, binary image processing methods are used for LP region determination. These methods, which are simpler and faster, are not affected by certain features, such as the color, shape, and size of the LP region. These methods are instead based on combinations of edge statistics and mathematical morphology [1], [14]. The average success rate of these methods is around 97% [1]. Zheng and others [15] found that the plate area contains rich edge and texture information and determined the plate region using edge statistics and morphological techniques.

Other methods exist for the character segmentation stage such as horizontal and vertical projections, mathematical morphology, and contour tracking. The most common methods are projection methods. Shi and others [3] separated characters from each other using vertical projection at binary level. Nomura and others [16] proposed a new adaptive morphological method based on histogram equalization for character segmentation, and separated characters by searching for natural segmentation points in a histogram of the projection and combining these points with the appropriate parts of the same character. Capar and Gokmen [17] developed an active contour model for the separation of characters.

To overcome the difficulties of processing high-resolution imagery in real-time, a novel cascade structure was found to be the fastest classifier [18]. It rejects false positives in the most efficient way. The classifier is trained using the core patterns of various types of LPs. Thus, both the computation load and accuracy of LP detection were improved. In [19], the best combination of binarization methods and parameters was determined, and an in-depth analysis of a multi-method binarization scheme for better character segmentation was presented. The authors' extensive quantitative evaluation shows a significant improvement over conventional single-method binarization methods. A survey on automatic number plate recognition was presented in [20]. In it, the authors briefly summarized LP detection algorithms for moving vehicles and listed the performance of some algorithms.

At the character recognition stage, methods such as Markov models, support vector machines, ANNs, and template matching have been used in different studies. Duan and others [6] used Markov models, Kim and others [21] used support vector machines, Anagnastapulos and others [2] used ANNs, and Comelli and others [22] and Huang and others [23] used pattern recognition methods in their character recognition systems.

In this work, we use edge-based image processing techniques for LP region determination, vertical projections in binarized images for character segmentation, and ANNs for character recognition.

III. License Plate Region Determination

At this stage, our purpose is to determine the LP region from the original image using edge-based image processing techniques. These techniques process the binary edge image obtained by identifying points that have sharp changes in brightness in the original image. The variation of brightness in the LP region is more noticeable and more frequent than elsewhere in the original image. Hence, the rectangular LP region contains richer edge and texture features than other regions. Therefore, this information is used to determine the LP region. In these methods, the LP determination rate is relatively high compared to other methods such as [1]. However, these methods have the disadvantage that they are excessively sensitive to unwanted edges, which means they are not suitable for complex images. To handle this problem, we use image enhancement and morphological methods.

1. Image Enhancement

An image with complex background usually contains noise that leads to undesired edge densities in the edge image. We used higher resolution images (640 × 480 pixels, 8-bit gray levels) in JPEG format, in contrast to 384 × 288 pixel images with 256 gray levels [15]. Edge-based methods are affected by noise. Therefore, noise must be eliminated before edge detection. To eliminate noise, mean filtering is used [24]. For this purpose, the original image is convolved with a 3 × 3 mean filter.

Although unwanted edges caused by noise, especially those arising from paving stones, are shown in Fig. 1, these edges are eliminated in Fig. 2 by the image enhancement. In addition, the vertical edges in the plate region are preserved.

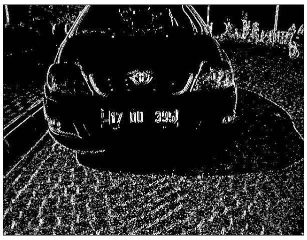


Fig. 1. Vertical edge image obtained from original image.

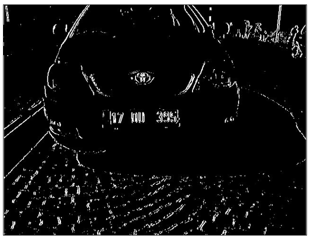


Fig. 2. Vertical edge image obtained from improved image.

2. Vertical Edge Detection

Edge detection methods are based on the principle of finding pixels with sharp changes in brightness, which correspond to the borders of objects in an image. There are many ways to obtain edges in an image. However, most methods may be grouped into two categories: gradient and Laplacian methods [24]. We use the gradient method to detect edges. First, we convolve the image with a Sobel filter (3 × 3) to find the approximate absolute gradient magnitude at each point in the image. We then detect edge points by thresholding. The edge-detected images are shown in Figs. 1 and 2.

Despite the image enhancement procedures prior to the edge detection process, in some cases, vertical edges apart from those in the plate region may be more intense. Hence, the LPRS may incorrectly detect other local regions as the plate region. Hence, we should eliminate all vertical edges apart from those in the plate region. We use an elimination method proposed by Zheng and others [15]. This method supposes that the length of vertical edges in the plate region changes within the range of 5 to 28 pixels and vertical edges outside this length range should be eliminated. We realized that the maximum value of the specified range is too low in our study because we used higher resolution images (640 × 480) than [15]. As a result of several tests carried out on a number of images, we set the maximum value of the specified range to 40 pixels. The result is shown in Fig. 3.

3. Rectangular LP Region Localization

The local region that has the highest vertical edge density in

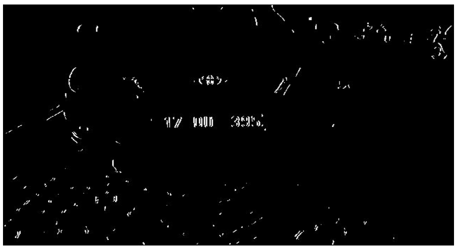


Fig. 3. New eliminated edge image.

the eliminated vertical edge image is the LP region. To determine the LP region, the local vertical edge densities must be calculated in this image. Hence, the eliminated vertical edge image is divided into local 16 × 16 blocks. The number of white pixels in each block is calculated and stored in a new 30 × 40 image. Each block in the edge image corresponds to a pixel in the new image. The brightness of the light dots is determined by the numeric value of the local intensity level in the new image (matrix **B** shown in Fig. 4).

The light pixels with the highest value in matrix **B** will be the pixels in the LP region. The other, unwanted light pixels must be eliminated, so the values of matrix **B** that are lower than a predefined threshold are set to zero. Hence, it is important to determine an appropriate threshold value for each different situation. Therefore, we added a function that returns a threshold value for each image. This transaction is performed as follows. First, the function obtains the average of the non-zero matrix elements and then the following steps are performed.

1) Matrix **B** is processed by this function and the first threshold value is obtained.

2) If the resulting threshold value is less than half of the maximum element of matrix **B**, matrix elements smaller than the threshold are set to zero.

3) The altered matrix **B** is processed by the function again, the threshold is calculated, and then the second step is repeated. These actions are repeated until the threshold is less than half of the maximum value of the first matrix.

The image of the new matrix **B** obtained after the elimination is shown in Fig. 5. After the elimination, we convolve the new matrix **B** with a 3 × 11 filter that consists of a matrix of ones to locate the candidate of LP region. The filter size was determined by an analysis of the test images used in this study. The final candidate LP regions emerge clearly in matrix **Z**, as shown in Fig. 6.

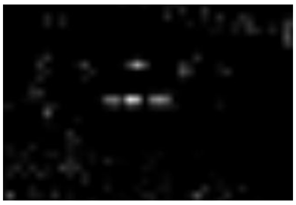


Fig. 4. Image of matrix **B** with local vertical edge densities.

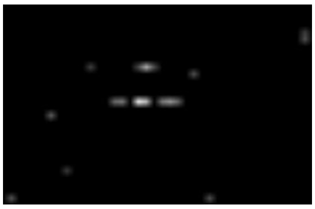


Fig. 5. Image of matrix **B** after elimination.

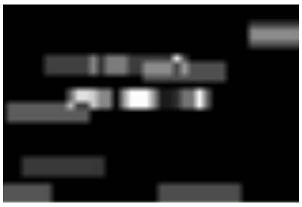


Fig. 6. Image of matrix **Z** with candidate LP regions.

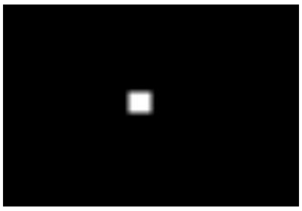


Fig. 7. Image of matrix **Z** after resetting.

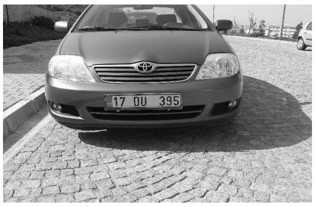


Fig. 8. Original car image.

Because the brightest region in the figure is the LP region, the next phase of the process focuses on this region. Hence, other regions with lower brightness values must be eliminated as much as possible. This process is performed by resetting values that are smaller than the maximum value of the matrix to zero. The image of the new matrix **Z** obtained after resetting is shown in Fig. 7. The bright area seen in the figure corresponds to the location of the LP region on the original image.

To determine the actual location on the original image in Fig. 8, the pixel values of the local regions (16 × 16) in the original image corresponding to each white pixel in matrix **Z** are set to one and the pixels of the remaining local regions are set to zero. Finally, by determining the coordinates of the rectangular plate region from Fig. 9, we can obtain the rectangular LP region shown in Fig. 10.

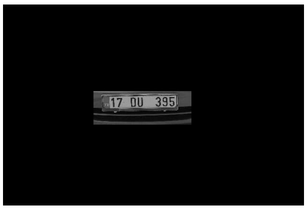


Fig. 9. LP region.

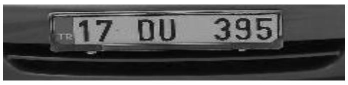


Fig. 10. LP region obtained from the original car image.

4. Skew Correction

To produce effective results in the plate character segmentation process, the LP should be made horizontal. This process is known as skew correction. The main task in this process is to determine the angle, at which the LP is skewed. The Radon transform is used to determine the skew angle in the current work. The skew correction process is performed by substituting the specific skew angle θ in (1). The results of the skew correction process are shown in Table 1.

Table 1. Examples of skew correction.

Obtained LP regions	Skew angle	Regions after correction
	$\theta = -1$	
	$\theta = 2$	
	$\theta = 4$	
	$\theta = -7$	

$$\begin{pmatrix} i' \\ j' \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix}. \qquad (1)$$

5. Unnecessary Area Removal

Unnecessary areas, that is, areas other than the plate characters can be observed in the roughly obtained rectangular LP region image (shown in Fig. 10). To separate the characters from each other, unnecessary fields must be removed. Otherwise, unsuccessful results are inevitable at the character segmentation stage. Therefore, we should focus on the narrowest possible frame that contains only the plate characters (see Fig. 11).

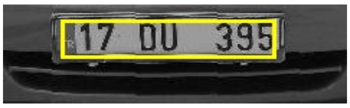


Fig. 11. Example of the narrowest frame that contains the plate characters.

To perform these operations, mostly horizontal (vertical) edges and horizontal (vertical) projections are used. A horizontal projection is obtained by calculating the number of bright pixels of each row on the horizontal edge image and vertical projection is obtained by calculating the number of bright pixels of each column on the vertical edge image. These operations consist of two steps. In the first step, unnecessary fields on the left and right sides of the frame are discarded. In the second step, unnecessary fields on the top and bottom sides of the frame are discarded. The same logic is used to perform the operations after the projections have been calculated using the horizontal (vertical) edges from the LP borders. Adjacent indices that have at least one white pixel in the projection as a whole will point to a piece of plate area. All pieces inside the projection are determined and the piece that has the longest range is selected (this region belongs to the LP). After these steps are performed successfully, the LP region is free of unwanted areas. The final narrowed LP region contains only plate characters, as shown in Fig. 12.

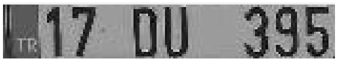


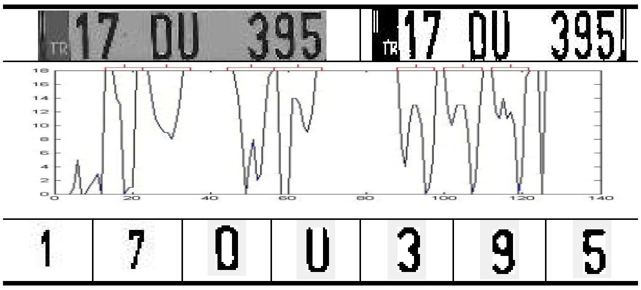
Fig. 12. Narrowed LP region.

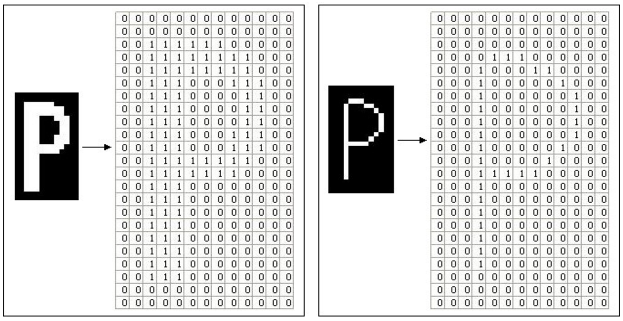
IV. Character Segmentation

Because the designed system will recognize each character individually, characters in the rectangular LP region must be separated from each other. Character segmentation is one of the most important steps in the automatic LPRS because all further steps rely upon its success. If the segmentation fails, the LPRS immediately gives the wrong results. We used a vertical projection method for the character segmentation in this work. These operations are performed in the following way.

First, the narrowed gray scale LP region is reduced to black and white (“0” and “1”) using a threshold value. The thresholding operation is performed as follows. If the value of a certain pixel is below the threshold value, the new value of the pixel will be zero. Otherwise, the new value will be one for pixels with values above the threshold value. The method is very simple, but the success of the method depends on the selection of an appropriate threshold for each LP image. In our study, we used Otsu’s method [25] to find the optimal threshold value for each LP image. After binarization, the binary LP image is improved using mean filtering. This process,

Table 2. Segmentation of characters in a LP.





0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	0	0	0	1	1	1	0	0
0	0	1	1	1	0	0	0	0	1	1	0	0
0	0	1	1	1	0	0	0	0	1	1	0	0
0	0	1	1	1	0	0	0	0	1	1	0	0
0	0	1	1	1	0	0	0	1	1	1	0	0
0	0	1	1	1	0	0	0	1	1	1	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

(a)

0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	1	0	0	0	1	1	0	0	0	0
0	0	0	1	0	0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0

(b)

Fig. 13. Binary character (a) before and (b) after the thinning process.

especially for dirty plates, ensures that the black dots appearing in the form of salt-and-pepper-type noise disappear in the white background. Additionally, this improvement helps separate the characters from each other. Finally, the coordinates of each character are determined in the improved LP image and the characters are separated using these coordinates (see the figures in Table 2).

V. Character Thinning

In order to analyze the complex structural features of each character such as connection points, edges, and loops and to implement the character recognition ANN, each segmented character needs to be thinned. For this process, a modified NWG thinning algorithm [26], which is a fast thinning algorithm, is used in this work. The result of the algorithm is a skeleton of the thinned character, as shown in Fig. 13.

VI. Character Recognition

The processing of input patterns by machine to produce meaningful outcomes is referred to as character recognition. For character recognition, there are simple and effective methods developed using character features such as their center of gravity, four-way cross-section structures, and skeletal structures. A character recognition procedure contains the steps of preprocessing, feature extraction, and classification. Considering the input pattern to be a point in feature space after feature extraction, the recognition problem becomes one of classical classification [27]. In the current work, we prefer to use an ANN that was been widely applied to pattern recognition.

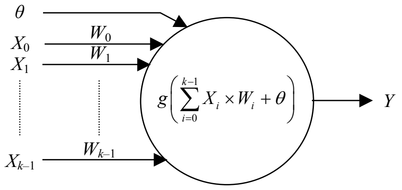


Fig. 14. Nerve cell (processing unit) used in the ANN.

1. ANNs

ANNs, in general, are formed by processing units, or, in other words, the processor elements (artificial nerve cells) are associated with each other. The structure of an artificial nerve cell is shown in Fig. 14. The structure of the connections between each nerve cell determines the structure of the ANN. How the connections are changed to achieve the desired goal is determined by the learning algorithm in the ANNs [28], [10]. Until the error rate between system result and expected result is reduced to a value sufficiently close to zero, the network weights are changed according to a learning rule. Because there is no rule of thumb to determine how many hidden layers are needed, we use one hidden layer, which gives good results in the current work. However, many hidden layers can be fruitful for difficult objects such as handwritten characters and face recognition problems. Therefore, in this work, a three-layer feedforward artificial neural network using a backpropagation learning algorithm is constructed and the characters are determined using this ANN in the character recognition stage.

2. Feedforward ANN

In feedforward ANNs, nerve cells (neurons) are organized into layers and the outputs of the cells in a layer are given as input to the next layer through weights. The input layer receives information from the external environment and transmits it to the nodes (processing units) in the hidden layer without any alteration. Network outputs are determined by processing information in the hidden and output layers. The most well-known backpropagation learning algorithms are used effectively in the training stage of this ANN. The architecture of a three-layer feedforward ANN used the character recognition phase of the current system is shown in Fig. 15.

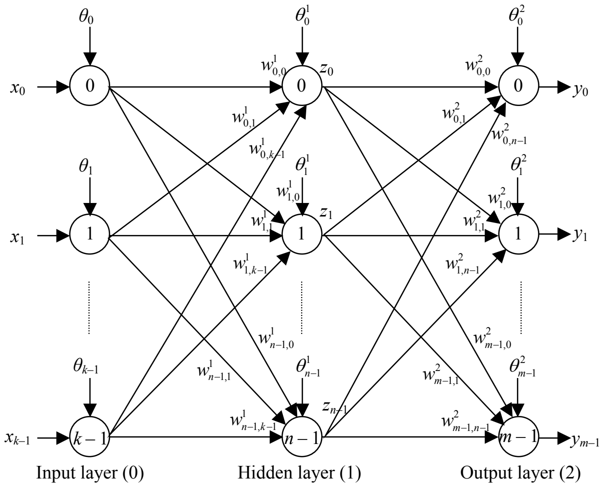


Fig. 15. Three-layer feedforward ANN model.

In the ANN model, *k* input, *n* hidden, and *m* output neurons are given. Input values are stored in $x[k]$, hidden layer outputs are stored in $z[n]$, and system outputs are stored in $y[m]$. Further, $\theta^1[i]$ and $\theta^2[i]$ are the bias vectors in the hidden and output layers, respectively, and $w^1[i, j]$ and $w^2[i, j]$ are the weight matrices in the hidden and output layers, respectively.

3. Training of Feedforward ANNs

The training of the ANN model consists of five steps:

1) *Activation step*: The network is activated (the weight and bias values are set up) by filling $\theta^1[i]$, $\theta^2[j]$, $w^1[i,j]$, and $w^2[i,j]$ with random numbers in the range [0,1].

2) *Feedforward step*: Values for $z[i]$ and $y[i]$ are generated using the following formula in the hidden and output layers.

$$z[i] = g\left(\sum_{j=0}^{k-1} w^1[i,j] \times x[j] + \theta^1[i]\right), \qquad i = 0,1, \dots ,n-1,$$

$$y[i] = g\left(\sum_{j=0}^{n-1} w^2[i,j] \times z[j] + \theta^2[i]\right), \qquad i = 0,1, \dots ,m-1.$$

The generated values ($z[i]$, $y[i]$) need to be converted into a format that is appropriate for the solution of a particular problem. A sigmoid saturation function is used as an activation function in this ANN model. The sigmoid function reduces the value to a range in [0,1] and thus helps simplify to the solution of nonlinear problems. The sigmoid function, mathematically, may be written as

$$g(\xi) = \frac{1}{1+e^{-\xi}}. \tag{2}$$

3) *Backpropagation step*: The error rates are calculated by comparing the system outputs with the expected values. Errors are spread backwards for the calculation of the new weights and are calculated using the following formula in the hidden and output layers.

$$E^2[i] = \left(B[i] - y[i]\right) g'\left(y[i]\right), \; i = 0,1, \dots ,m-1$$

$$E^1[i] = g'\left(z[i]\right)\left(\sum_{j=0}^{m-1} E^2[j] w^2[j,i]\right), \; i = 0,1, \dots ,n-1.$$

The derivative of the sigmoid function is used to calculate the errors and is given by

$$g'(\xi) = g(\xi)\left(1-g(\xi)\right). \tag{3}$$

4) *Calculation of new weights*: The improved weights in the hidden and output layers are calculated using the following formula.

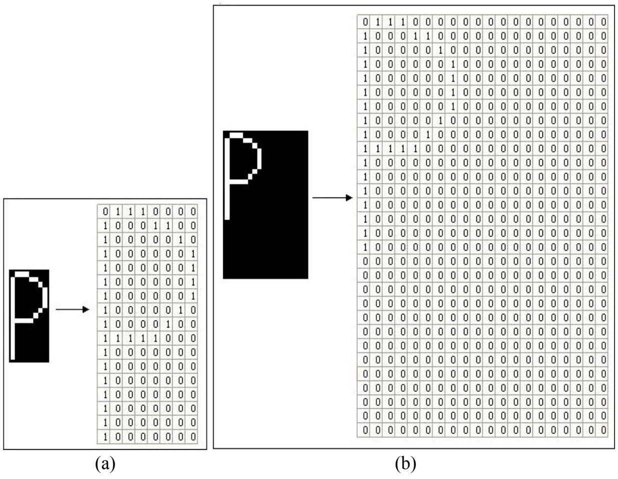
$$\begin{aligned} w^1[i,j] &= \sum_{j=0}^{k-1} \lambda E^1[i] x[j], \; i = 0,1, \dots ,n-1, \\ w^2[i,j] &= \sum_{j=0}^{n-1} \lambda E^2[i] z[j], \; i = 0,1, \dots ,m-1. \end{aligned} \tag{4}$$

5) *Calculation of Mean Square Error (MSE) of the system*: The *MSE* for each character is calculated using the following formula.

$$MSE = \frac{1}{m} \sum_{i=0}^{m-1} \left(B[i] - y[i]\right)^2. \tag{5}$$

Until the *MSE* is reduced to a value sufficiently close to zero, the calculations in each step (except the first step) are repeated. When the *MSE* reaches a predetermined threshold (a value close to zero) for each character, training of the network is stopped. As a result, the most appropriate weight values for each character are saved to a file for character recognition.

The character matrix given to the ANN as an input value must be prepared. A character matrix is a two-dimensional array of black and white pixels of a thinned character image.



0	1	1	1	0	0	0	0
1	0	0	0	1	1	0	0
1	0	0	0	0	0	1	0
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1
1	0	0	0	0	0	1	0
1	0	0	0	0	1	0	0
1	1	1	1	1	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0

(a)

0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(b)

Fig. 16. Matrices of (a) thinned character and (b) character transferred to the template.

Table 3. Parameters of the ANN model.

Input layer	$k = 600$ Number of neurons = 600 Number of inputs = 600 x[600]
Hidden layer	$n = 300$ Number of neurons = 300 Number of outputs = 300 z[300], θ[300], w^1[300, 600]
Output layer	$m = 33$ Number of neurons = 33 Number of outputs = 33 y[33], B[33], w^2[33, 300], B[33]

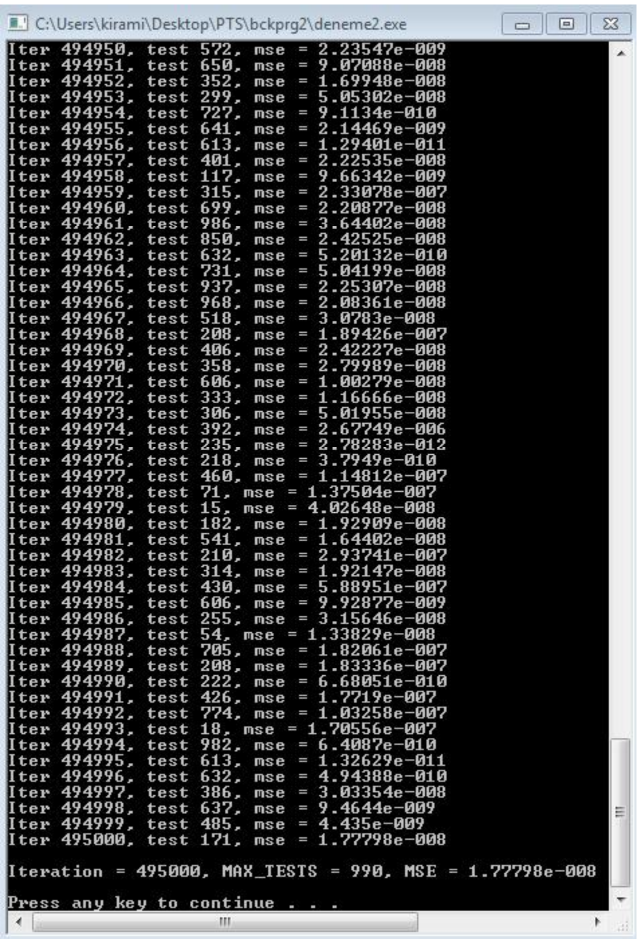
Table 4. Performance of the LPRS.

LPRS stage	Number of images/ success	Performance (%)
LP region determination	357/357	100
Character segmentation	357/357	100
Character recognition	357/346	96.92

The sizes of the characters obtained from the LP region after thinning operations can vary for different LPs. However, all characters supplied to the ANN must be a standard size. A template matrix that has a maximum size to fit all of the characters under consideration is used. The size of this template matrix is (30 × 20), which was determined by an analysis of the different test characters in our current image set. The pixel values of the thinned characters are placed into a template matrix starting from the top left corner of the template matrix, as shown in Fig. 16.

Elements of this template matrix are stored in a vector $x[k]$ that holds the input values. The size of input vector $x[k]$ is equal to the number of elements of the template matrix (30 × 20 = 600). Hence, for 600 input values, 600 neurons are used in the input layer of the ANN model. The optimal size of the hidden layer is usually between the size of the input and size of the output layer, or should be 2/3 the size of the input layer plus the size of the output layer [29]. Therefore, we use 300 neurons in the hidden layer. Because 33 characters are used in standard Turkish LPs, the size of the vector that holds the expected values is also 33 and we have used 33 neurons in the output layer. Each row of the matrix corresponds to the expected value for each character in Turkish LPs and can easily be modified to recognize the LPs of other countries (see Table 3).

During the ANN training stage, 30 samples of each character are used. Hence, a total of 990 test data are used for different characters. The proposed LPRS performs 500 iterations for each test data set considering the ANN parameters for each layer, as given in Table 4. Hence, the iterations are repeated 495,000 times for 990 test data. The *MSE* for each character is about 1.0 × 10^{-8}, as shown in Fig. 17.



```
Iter 494950, test 572, mse = 2.23547e-009
Iter 494951, test 650, mse = 9.07088e-008
Iter 494952, test 352, mse = 1.69948e-008
Iter 494953, test 299, mse = 5.05302e-008
Iter 494954, test 727, mse = 9.1134e-010
Iter 494955, test 641, mse = 2.14469e-009
Iter 494956, test 613, mse = 1.29401e-011
Iter 494957, test 401, mse = 2.22535e-008
Iter 494958, test 117, mse = 9.66342e-009
Iter 494959, test 315, mse = 2.33078e-007
Iter 494960, test 699, mse = 2.20877e-008
Iter 494961, test 986, mse = 3.64402e-008
Iter 494962, test 850, mse = 2.42525e-008
Iter 494963, test 632, mse = 5.20132e-010
Iter 494964, test 731, mse = 5.04199e-008
Iter 494965, test 937, mse = 2.25307e-008
Iter 494966, test 968, mse = 2.08361e-008
Iter 494967, test 518, mse = 3.0783e-008
Iter 494968, test 208, mse = 1.89426e-007
Iter 494969, test 406, mse = 2.42227e-008
Iter 494970, test 358, mse = 2.79989e-008
Iter 494971, test 606, mse = 1.00279e-008
Iter 494972, test 333, mse = 1.16666e-008
Iter 494973, test 306, mse = 5.01955e-008
Iter 494974, test 392, mse = 2.67749e-006
Iter 494975, test 235, mse = 2.78283e-012
Iter 494976, test 218, mse = 3.7949e-010
Iter 494977, test 460, mse = 1.14812e-007
Iter 494978, test 71, mse = 1.37504e-007
Iter 494979, test 15, mse = 4.02648e-008
Iter 494980, test 182, mse = 1.92909e-008
Iter 494981, test 541, mse = 1.64402e-008
Iter 494982, test 210, mse = 2.93741e-007
Iter 494983, test 314, mse = 1.92147e-008
Iter 494984, test 430, mse = 5.88951e-007
Iter 494985, test 606, mse = 9.92877e-009
Iter 494986, test 255, mse = 3.15646e-008
Iter 494987, test 54, mse = 1.33829e-008
Iter 494988, test 705, mse = 1.82061e-007
Iter 494989, test 208, mse = 1.83336e-007
Iter 494990, test 222, mse = 6.68051e-010
Iter 494991, test 426, mse = 1.7719e-007
Iter 494992, test 774, mse = 1.03258e-007
Iter 494993, test 18, mse = 1.70556e-007
Iter 494994, test 982, mse = 6.4087e-010
Iter 494995, test 613, mse = 1.32629e-011
Iter 494996, test 632, mse = 4.94388e-010
Iter 494997, test 386, mse = 3.03354e-008
Iter 494998, test 637, mse = 9.4644e-009
Iter 494999, test 485, mse = 4.435e-009
Iter 495000, test 171, mse = 1.77798e-008

Iteration = 495000, MAX_TESTS = 990, MSE = 1.77798e-008

Press any key to continue . . .
```

Fig. 17. Results of the ANN training stage.

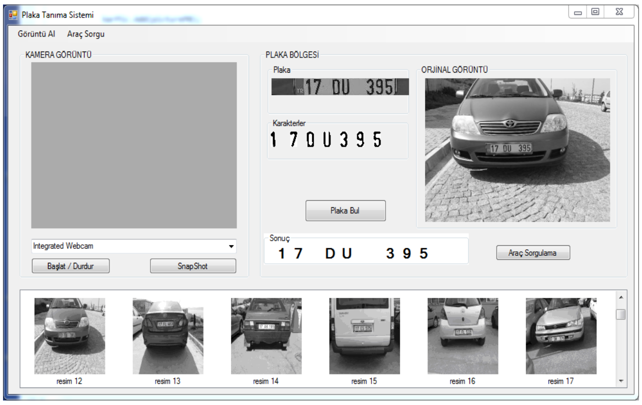
4. Recognition of New Characters

After network training is complete and the appropriate weight values have been obtained, the network runs the feedforward step using the obtained weights to recognize characters and produces system outputs $y[m]$. If the system is properly trained, as in this work, one of the vector $y[m]$ values must be one or close to one, and others must be zero or close to zero. Thus, the current can successfully recognize a character that has not previously been encountered in the test data set.

VII. GUI of the Proposed LPRS

We used 357 car images acquired from real scenes for this study. We developed the Graphical User Interface (GUI) for the LPRS using C# in Microsoft Visual Studio with a laptop

https://doi.org/10.4218/etrij.17.0115.0766



Integrated Webcam

Sonuç: 17 DU 395

Fig. 18. GUI of the proposed LPRS.

equipped with an Intel core i3 processor with 2.13 GHz and 4 GB of RAM. The GUI of the developed system is shown in Fig. 18. On the left of the screenshot, an original car image taken from the image database, a button used to find the LP in the car image, and the processing time are shown. In the middle, the image of the rectangular LP region, images of the characters segmented from the LP region, and textual LP characters are shown. The processing time in the current developed system is very low (58 ms on average) and it may be considered a real-time system.

VIII. Conclusion

An LPRS that spends less processing time, less computing power and has better recognition rates under fewer restrictions was developed using ANNs and various improved image processing techniques in this study. In the LP region determination stage, to increase system performance, the original image is enhanced by image processing. The rectangular plate region is obtained by edge-based image processing methods on the binary image. Using skew correction, the plate region obtained in the plate region determination stage is prepared for the character segmentation stage. Characters are separated from each other by a vertical projection method on the plate region. The segmented characters are prepared for the character recognition stage by thinning. In the character recognition stage, a three-layer feedforward ANN using a backpropagation learning algorithm is constructed and individual characters are determined using this ANN.

In the performance metric used for the proposed system, it is assumed that an LP is not recognized correctly if one or more characters segmented in the LP are incorrect or cannot be recognized.

The results of this study show that the range of distances and viewpoints between the vehicle and camera are limited to an appropriate range of values. The results of experiments performed on 37 images that were obtained from different distances and perspectives shows that our system gives the best results on images taken from 1.5 m to 3 m away from the camera and for perspectives ranging between −20° and +20°. In the other cases, the sizes of the LPs are either too small (13 images) or too big (15 images) to detect. The LPs could not be determined for nine images outside of the angle range specified above. These are the major restrictions of the developed system. Hence, these pictures were not taken into account in the test stage for the current work.

The system performance on the 357 test images taken under the specified conditions is shown in Table 4. The developed system has a success rate of about 97%. The processing time of the system is about 60 ms in all images. Thus, we believe that this system can be used to solve problems in real time. In the near future, we think that more advanced image processing techniques will arise, so an LPRS with a higher performance under fewer restrictions could be developed.

Acknowledgments

This work was supported by the Canakkale Onsekiz Mart University Research Fund (Grant No. 2011/134).

References

[1] C.N.E. Anagnostopoulos et al., “License Plate Recognition from Still Images and Video Sequence: a Survey,” *IEEE Trans. Intell. Trans. Syst.*, vol. 9, no. 3, Sept. 2008, pp. 377–391.

[2] C.N.E. Anagnostopoulos et al., “A License Plate-Recognition Algorithm for Intelligent Transportation System Applications,” *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no 3, Sept. 2006, pp. 377–392.

[3] X. Shi, W. Zhao, and Y. Shen, “Automatic License Plate Recognition System Based on Color Image Processing,” *Comput. Sci. Its Applicat.*, Singapore, May 9–12, 2005, pp. 1159–1168.

[4] N. Zimic et al., “The Fuzzy Logic Approach to the Car Number Plate Locating Problem,” *Proc. Intell. Inform. Syst.*, Bahamas, Dec. 8–10, 1997, pp. 227–230.

[5] R. Zunino and S. Rovetta, “Vector Quantization for License-Plate Location and Image Coding,” *IEEE Trans. Ind. Electron.*, vol. 47, no. 1, Feb. 2000, pp. 159–167.

[6] T.D. Duan et al., “Building an Automatic Vehicle License-Plate Recognition System,” *Proc. Int. Conf. Comput. Sci.*, Cantho, Vietnam, Feb. 2005, pp. 59–63.

[7] F. Kahraman, B. Kurt, and M. Gökmen, “License Plate Character Segmentation Based on the Gabor Transform and Vector Quantization,” *ISCIS Comput. Inform. Sci.*, vol. 2869, 2003, pp. 381–388.

[8] C.T. Hsieh, Y.S. Juan, and K.M. Hung, "Multiple License Plate Detection for Complex Background," *Int Conf. Adv. Inform. Netw. Aapplcat.*, Taipei, Taiwan, Mar. 28–30, 2005, pp. 389–392.

[9] L. Dlagnekovin, *Video-Based Car Surveillance: License Plate, Make, and Model Recognition*, M.S. thesis, Comput. Sci. Eng. Dept., Univ. California, San Diego, USA, 2004.

[10] L. Gang, Z. Ruili, and L. Ling, "Research on Vehicle License Plate Location Based on Neural Networks," *Int. Conf. Innovative Comput., Inform. Contr.*, Beijing, China, Aug. 30–Sept. 1, 2006, pp. 174–177.

[11] J.A.G. Nijhuis et al., "Car License Plate Recognition with Neural Networks and Fuzzy Logic," *Proc. IEEE Int. Conf. Neural Netw.*, Perth, Australia, Nov. 27–Dec. 1, 1995, pp. 2232–2236.

[12] S.K. Kim, D.W. Kim, and H.J. Kim, "A Recognition of Vehicle License Plate Using a Genetic Algorithm Based Segmentation," *Proc. Int. Conf. Image Process.*, Sept. 1996, pp. 661–664.

[13] J. Xiong et al., "Locating Car License Plate Under Various Illumination Conditions Using Genetic Algorithm," *Proc. Int. Conf. Signal Process.*, Beijing, China, Aug. 31–Sept. 4, 2004, pp. 2502–2505.

[14] B. Hongliang and L. Changping, "A Hybrid License Plate Extraction Method Based on Edge Statistics and Morphology," *Proc. Int. Conf. Pattern Recogn.*, Cambridge, UK, Aug. 23–26, 2004, pp. 831–834.

[15] D. Zheng, Y. Zhao, and J. Wang, "An Efficient Method of License Plate Location," *Pattern Recogn. Lett.*, vol. 26, no. 15, Nov. 2005, pp. 2431–2438.

[16] S. Nomura et al., "A Novel Adaptive Morphological Approach for Degraded Character Image Segmentation," *Pattern Recogn.*, vol. 38, no. 11, Nov. 2005, pp. 1961–1975.

[17] A. Capar and M. Gokmen, "Concurrent Segmentation and Recognition with Shape-Driven Fast Marching Methods," *Proc. Int. Conf. Pattern Recogn.*, Hong Kong, China, Aug. 20–24, 2006, pp. 155–158.

[18] B.G. Han et al., "Real-Time License Plate Detection in High-Resolution Videos Using Fastest Available Cascade Classifier and Core Patterns," *ETRI J.*, vol. 37, no. 2, Apr. 2015, pp. 251–261.

[19] Y. Yoon et al., "Best Combination of Binarization Methods for License Plate Character Segmentation," *ETRI J.*, vol. 35, no. 3, June 2013, pp. 491–500.

[20] K. Sonavane, B. Soni, and U. Majhi, "Survey on Automatic Number Plate Recognition," *Int. J. Comput. Applicat.*, vol. 125, no. 6, 2015, pp. 1–4.

[21] K.K. Kim et al., "Learning-Based Approach, for License Plate Recognition," *Proc. IEEE Signal Process. Soc. Workshop, Neural Netw. Signal Process.*, Dec. 11–13, 2000, pp. 614–623.

[22] P. Comelli et al., "Optical Recognition of Motor Vehicle License Plates," *IEEE Trans. Veh. Technol.*, vol. 44, no. 4, Nov. 1995, pp. 790–799.

[23] Y.P. Huang, S.Y. Lai, and W.P. Chuang, "A Template-Based Model for License Plate Recognition," *IEEE Int. Conf. Netw., Sensing Contr.*, Taipei, Taiwan, Mar. 21–23, 2004, pp. 737–742.

[24] R. Gonzalez and R. Woods, *Digital Image Processing*, Englewood Cliffs, NJ, USA: Prentice Hall, 2002.

[25] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. 9, no. 1, Jan. 1979, pp. 62–66.

[26] M.V. Nagendraprasad, P.S.P. Wang, and A. Gupta, "Algorithms for Thinning and Rethickening Binary Digital Patterns," *Digital Signal Process.*, vol. 3, no. 2, Apr. 1993, pp. 97–102.

[27] M. Cheriet et al., *Character Recognition Systems: a Guide for Students and Practitioner*, Hoboken, NJ, USA: John Wiley & Sons, 2007.

[28] J.A. Anderson, "Introduction to Neural Networks," *Handbook of Brain Theory and Neural Networks*, Cambridge, MA, USA: MIT Press, 1995.

[29] J. Heaton, *Introduction to Neural Networks for Java, 2nd Edition*, Heaton Research, 2008, pp. 158–159.

İbrahim Türkyılmaz received his MS and PhD degrees in applied mathematics from the Department of Mathematics, University of Manchester, UK, in 1997 and 2001, respectively. He is currently a faculty member in the Department of Computer Engineering, Çanakkale Onsekiz Mart University, Turkey. His major research interests include scientific computing and image processing.

Kirami Kaçan received his BS and MS degrees in computer engineering from the Department of Computer Engineering, Çanakkale Onsekiz Mart University, Turkey, in 2008 and 2010, respectively. He worked between 2009 and 2011 as a research assistant at the Computer Engineering Department of Çanakkale Onsekiz Mart University. He is currently working as a project manager at EDU Software Company in Ankara, Turkey.

https://doi.org/10.4218/etrij.17.0115.0766