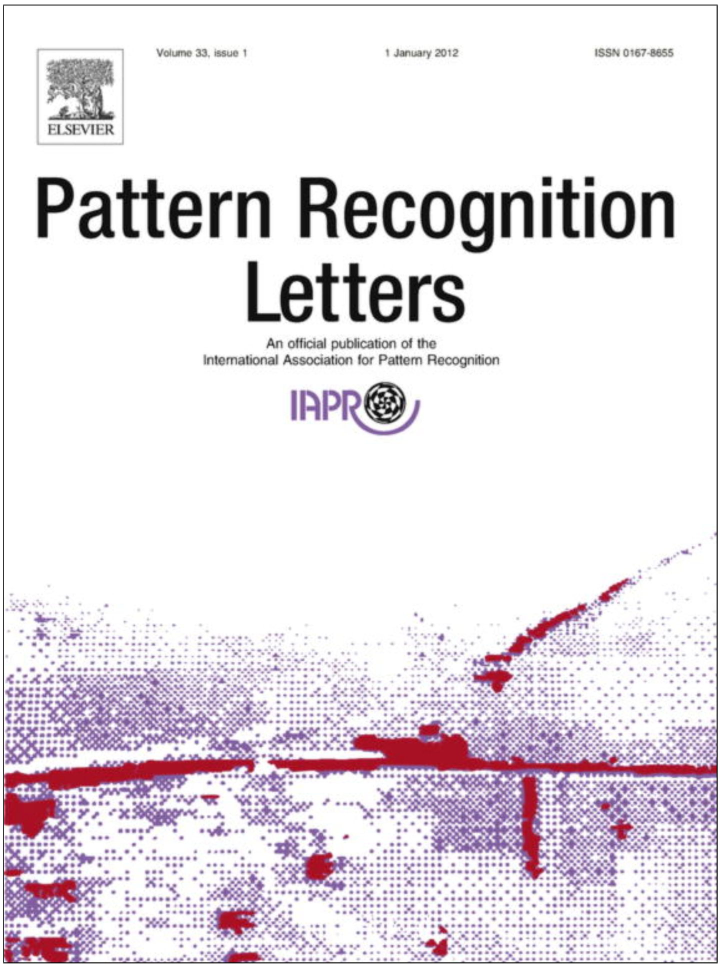

Volume 33, issue 1
1 January 2012
ISSN 0167-8655
ELSEVIER
Pattern Recognition Letters
An official publication of the
International Association for Pattern Recognition
IAPR

Pattern Recognition Letters 33 (2012) 1066–1074

Contents lists available at SciVerse ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Automatic vehicle identification for Argentinean license plates using intelligent template matching

Nicolás Fernando Gazcón [a,c,*], Carlos Iván Chesñevar [b,c], Silvia Mabel Castro [a]

[a] VyGLab, Department of Computer Science and Engineering, Universidad Nacional del Sur., Av. Alem 1253, 8000 Bahía Blanca, Argentina
[b] Artificial Intelligence Research and Development Laboratory, Department of Computer Science and Engineering, Universidad Nacional del Sur., Av. Alem 1253, 8000 Bahía Blanca, Argentina
[c] National Council of Scientific and Technical Research (CONICET) Av. Rivadavia 1906, 1033 Buenos Aires, Argentina

ARTICLE INFO

Article history:
Received 28 July 2011
Available online 19 February 2012
communicated by A. Koleshnikov

Keywords:
License plate recognition
Template recognition
Image processing

ABSTRACT

The problem of automatic number plate recognition (ANPR) has been studied from different aspects since the early 90s. Efficient approaches have been recently developed, particularly based on the features of the license plate representation used in different countries. This paper focuses on a novel approach to solving the ANPR problem for Argentinean license plates, called Intelligent Template Matching (ITM). We compare the performance obtained with other competitive approaches to robust pattern recognition (such as artificial neural networks), showing the advantages both in classification accuracy and training time. The approach can also be easily extended to other license plate representation systems different from the one used in Argentina.

1. Introduction and motivations

During the last decades, the population increase all over the world has led to a similar increase in the complexity of means of transportation, particularly vehicles such as cars and trucks. This way, the huge number of vehicles in roads and highways makes necessary to automatize a number of tasks, particularly vehicle identification. Automatic recognition of vehicles is useful in several control tasks (verification of access in bridges, tunnels; traffic levels, speed limits, border control, etc.). In fact, several technical terms associated with traffic in large cities (such as Automatic Cruise System, Intelligent Traffic System or Intelligent Highway (Alvarez et al., 2010; Parra et al., 2010)) are highly connected to automatic vehicle identification, which involves in most cases the problem of *automatic number plate recognition (ANPR)*.

ANPR has deserved attention from pattern recognition community for several decades. The first commercial systems can be traced back to the 80s, even though only in the 90s the first robust approximations are introduced, based on Optical Character Recognition (OCR) techniques (Lotufo et al., 1990). In the last years, several strategies have been proposed which attack the different stages of this problem (Bailey et al., 2002).

The problem of ANPR can be associated with techniques coming from several areas such as pattern recognition, image processing and computer vision. The ultimate goal is the identification of a plate number from pictures taken to vehicles (vehicle images). Several problems can arise from the environment conditions (illumination, deteriorated plates, etc.), which motivate the development of robust and reliable systems for ANPR. In many cases, the feasibility of solving this problem is connected with the hardware used (e.g. infrared cameras in tunnels), as well as the robustness of the algorithm being used. From image capture up to the final plate identification several stages can be identified, namely:

Capture: the image capture is strongly dependant from the hardware used and environment conditions (e.g. quality and type of camera, illumination, weather conditions, etc.). As a result, a graphic file is obtained.
Image enhancement: the captured image is normalized, eliminating noise, improving contrast, etc. A new graphic file is obtained with an enhanced image.
Plate extraction and identification: this stage involves the search, segmentation, and normalization of the plate in the previous image obtained. This can be optimized on the basis of the features of plates in some countries or regions (Parasuraman and Kumar, 2010,). The output is an image of the license plate.
Character segmentation: once the plate has been extracted, individual characters must be detected in the image, in order to proceed with the character identification. Several methods have been proposed, such as projection (Xia and Liao, 2011), connected components (Caner et al., 2008) or morphology (Dudarin and Kovaccic, 2010). As a result, every single segmented character is obtained.

* Corresponding author at: VyGLab, Department of Computer Science and Engineering, Universidad Nacional del Sur., Av. Alem 1253, 8000 Bahía Blanca, Argentina.
E-mail addresses: nfg@cs.uns.edu.ar (N.F. Gazcón), cic@cs.uns.edu.ar (C.I. Chesñevar), smc@cs.uns.edu.ar (S.M. Castro).

0167-8655/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2012.02.004

Individual identification of characters: OCR with each character is involved in this last step. Several techniques have been proposed, such as template matching (Khalil, 2010) or different machine learning classifiers (Garcia-Osorio et al., 2008), including Artificial Neural Networks (ANN) (Wang, 2009, 2010).

The ANPR problem has some features which make it country-dependable, since characters, symbols, texture, etc. may vary from one plate representation system to another. Thus, recent research (Ho and Pun, 2010,,) has focused on solving ANPR in particular countries such as Macao, India and Iran. To the best of our knowledge, no approach has been advanced yet to solve the ANPR problem in the context of Argentinean license plates. In fact, Argentina is a developing country in which the number of licensed vehicles has increased in about 200% in the last decade (INFOBAE, 2011), making more complex and difficult the organization and effective use of diverse facilities (from parking lots, automatic fee systems for highways, etc.). Consequently, providing an efficient solution to the ANPR problem for Argentinean license plates is of particular importance.

An essential requirement for effective OCR lies in the development of an accurate recognition algorithm. Besides accuracy, flexibility and speed are the main features that characterize a good OCR algorithm (Mani and Srinivasan, 1997). In particular, in the context of ANPR, robustness is also an important issue in the OCR process.

One of the simplest ways to classify characters is through the use of *templates*. Indeed, template matching is a well-known method for character classification which has been used in several research works (Ho and Pun, 2010,). This method is simple, flexible and has a reasonable accuracy rate, even though it may not be so efficient or noise resilient as other more sophisticated approaches such as ANN (Brunelli, 2009, 2002). In fact, ANN have long proved their usefulness for OCR tasks (Arnold and Miklos, 2010, 2009). ANN are robust for such tasks, and can cope with character recognition in situations which differ substantially from ideal cases (Ganapathy and Lean, 2006). However, ANN require a considerable training process, and result in a *black-box* model.

In this paper we propose a new methodology for solving ANPR called *Intelligent Template Matching (ITM)*. We show how the new proposed method can achieve higher accuracy than traditional template matching while keeping its flexibility. Experimental results show that ITM behaves as fast and robust as ANN with no need of a previous training process. A specific software was developed, oriented towards Argentinean plates, based on the stages previously described and using two approximations to OCR implementation (ITM and ANN). The results were contrasted in accuracy and applicability of the technique taking into account other research papers in the area (Almustafa et al., 2011, 2010,,,). In our experiments we considered solving the ANPR problem in two different situations: (a) in almost ideal conditions, and (b) in a real-world environment, adding variation in illumination, shadows, damaged plates, etc.

The rest of this paper is structured as follows. Section 2 details the main features of the developed system for car plate identification. Section 3 analyzes two alternative implementations of the OCR module: the first one is based on ANN, while the second one is based on ITM. Then in Section 4 we provide experimental results obtained when performing OCR tasks with both approaches, analyzing and contrasting their performance in different contexts (an ideal situation and a more realistic environment). The relation of our approach to other techniques for ANPR is presented in Section 6. Finally, in Section 7 we summarize the main conclusions that have been obtained and outline some research lines for future work.

2. License plate recognition using the FGL system

Following the different stages identified in the previous section, a particular software product was developed for license plate recognition, called *FGL*.[1] Its structure was developed and optimized for Argentinean license plates.

Argentinean car plates are composed of a reflective material, having an external white frame (Fig. 1a). This outer frame presents an inner black rectangle, which contains six white characters. These are composed of three letters, followed by an empty space (usually used for a smaller letter which identifies replicated plates) and finally three digits. At the top of the car plate the Argentinean coat of arms can be found, as well[2] as the country name (*Argentina*) written in light blue color.

An example of the steps required to perform the plate detection process is shown in Fig. 1b. From a color image provided as an input, the first step consists in its conversion to the corresponding grayscale image. This is needed because the detection algorithm will apply thresholds on the grayscale image, resulting in a binarized version of the original image, onto which a rectangle search will be performed. This search will involve identifying rectangle borders on the binarized image, which can be then translated again into rectangles in the grayscale image (located in the same positions as those found in the binarized image). Each rectangle found in the grayscale image will be extracted (removing skew and then obtaining it straightened) and a symbol search will be performed. In case of detecting six symbols in the expected positions, the detection algorithm accepts the rectangle as a license car plate (otherwise that rectangle is discarded and the search continues). Once the rectangle has been identified successfully, the characters associated with it are extracted, and the subsequent identification is carried out by the OCR module. Fig. 2 shows a flowchart identifying the main elements of our approach.

Next we summarize the details associated with each particular stage of our approach:

2.1. Preprocessing

The input to the preprocessing stage is a graphic file in a standard graphic format (TIFF, BMP, JPG, etc.). The OpenCV library (OpenCV, 2011) was used in order to convert the input file into a RGB representation. As far as resolution level is concerned, we assumed that graphic files should have a resolution of at least 300×200 pixels in order to be admissible for processing. There was no upper bound in resolution level (some of the graphic files analyzed were taken with a 5 Megapixel camera, corresponding to pictures with a resolution of 2292×1944 pixels). The RGB graphic files obtained as an output were then converted into grayscale format.

It must be remarked that no other image enhancement technique was required (e.g. removing noise), speeding up the image preprocessing stage. This choice was made on the basis of the high recognition rate obtained on our plate detection experiments. Thus, the final output of the preprocessing stage is just a grayscale version of the original graphic file, to be provided as an input for the identification and license plate extraction algorithm described in the next stage.

[1] This software was developed as part of the research project *PGI 24/N028, Visual Representations and Interactions for Visual Analysis of Large Datasets*, granted by Secretaría de Ciencia y Tecnología, Universidad Nacional del Sur, and is freely available upon request.

[2] For interpretation of color in 'Figs. 1, 4–6', the reader is referred to the web version of this article.

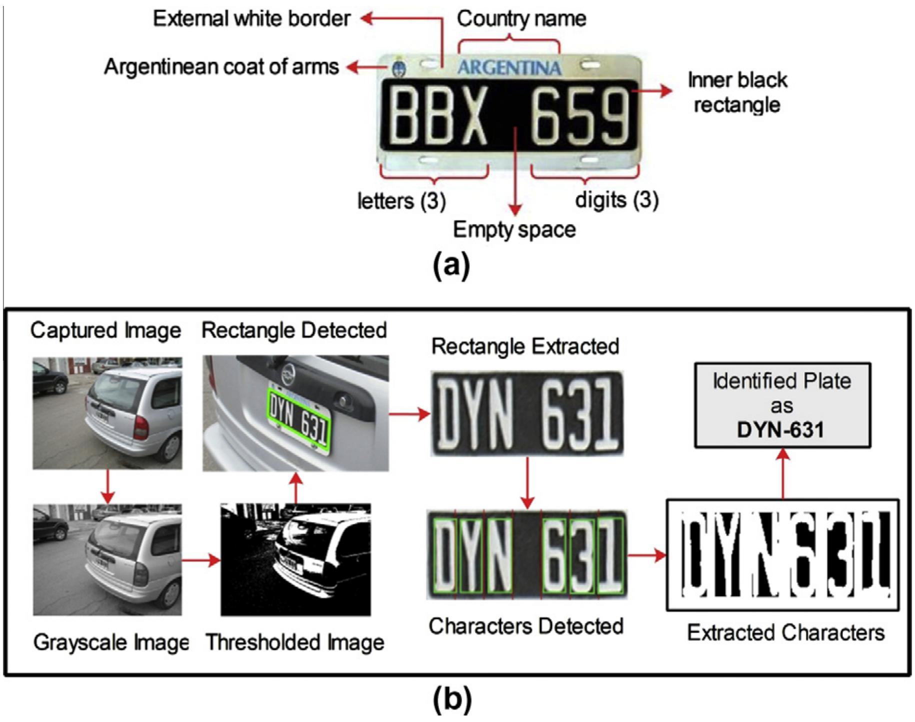


Fig. 1. An example of Argentinean car plates (a) and images of the general detection process (b).

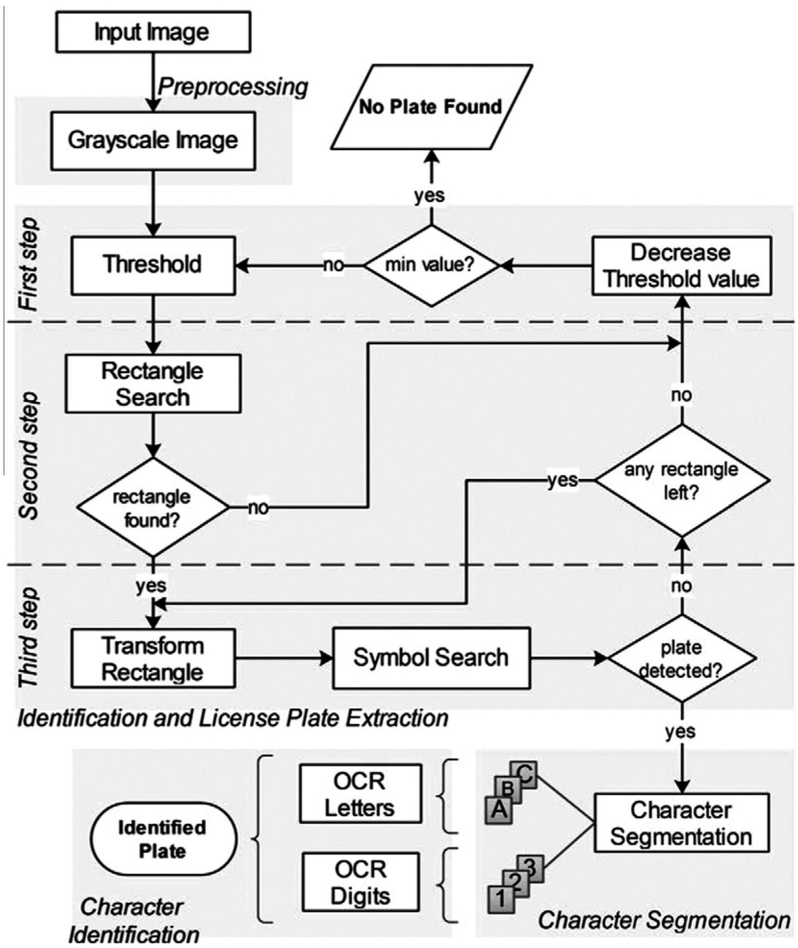


Fig. 2. The FGL approach: sketch.

2.1.1. Identification and license plate extraction

The license plate identification in an image involves first a rectangle search, which accounts for detecting a candidate area in which letters and numbers can be found. As described in the previous stage, the input at this point is a grayscale image that will undergo several transformations in order to identify rectangles on it, extract them and finally determine which one (if any) corresponds to a license car plate. In order to achieve these goals, we can describe the whole identification and extraction process as a three-step method (see Fig. 2). The first step corresponds to the threshold loop, the second step can be associated with the rectangle search, and the third step corresponds to the symbol search. The details of each step can be described as follows:

Step 1: Given the original grayscale image, a thresholding process is performed, and as a result a binarized version of the image is obtained. The thresholding process is based on using a particular grayscale value (which has a range from 0 to 255) as a cutoff to map the gray level values to black and white pixels. The first cutoff value is 250 (this cutoff value is used on the basis of experimental results since most rectangles were successfully found with high grayscale values). At this point, a contour search is performed with the binarized image (corresponding to the next step of the plate detection process). In the case that no plate is found (either because no rectangle was found, or because every rectangle found did not correspond to a car license plate), the initial threshold value is decreased, and the above process is repeated. The algorithm uses a decrease value of 10, which can be modified to get a trade-off between level of detail in the search and performance (in our case we use this small step to obtain a higher precision in the search). In the worst case the loop continues until a user-defined minimum value is reached, and a negative output is obtained (i.e. no car license plate was identified in the image). In our experiments this minimum was set to zero, allowing to search throughout the whole range of gray values.

Step 2: Once the binarized image has been obtained, a rectangle search is performed in order to find a car license plate. Searching for a rectangle involves on its turn performing a contour search. Contours are approximated to polygons, selecting those which are rectangular and with an area larger than a minimum value (in order to filter those rectangles which are too small). We consider a contour as an acceptable rectangle if the corresponding polygon obtained has four vertices and the rectangle area is larger than a minimum value. If the rectangle is acceptable, the coordinates of its four vertices are stored in memory for later use. As a result of this process, a collection R_1, $R_2, \ldots R_n$ of rectangles associated with the binarized image is obtained.

Step 3: As a last step, every rectangle found in the previous step is analyzed. This will involve a loop where every rectangle R_i is retrieved from memory, and a perspective transformation is performed in order to obtain a new grayscale image (in which the rectangle is straightened).

The symbol search process is then carried out on this new straightened image. The above transformation is just an image warping[3] from the four vertices of the original rectangle into four new vertices. We consider a warp into a new image of 246×110 pixels. This task can easily be done with OpenCV (OpenCV, 2011) by just defining two matrices (one with the original vertices and the other with the destination vertices). The OpenCV library can be used to obtain the mapping of pixels from the original rectangle into the destination image using a cubic interpolation.

Once the straightened image has been obtained, symbols are searched for within the image. Contour search is used for detecting symbols, evaluating their layout and size. Whenever six elements are found in the expected positions and with appropriate sizes according to the image size, the rectangle is identified as a license car plate. It must be noted that the symbol layout was assumed from the official layout for Argentinean license plates. In addition, it must be remarked that before searching contours the algorithm uses the morphological operation erode-dilate to deal with possible junctions between characters (which could be misleading for the recognition process). This final step can be described in more detail as a loop where every contour found is first filtered using its area and then contrasted with the expected positions of characters in the plate in order to accept (or reject) the symbol. It is important to take into account the area value in order to avoid misclassifications (e.g. blobs obtained from a damaged plate). The expected positions of characters in a plate are defined as a set of six elements (where each element corresponds to a (X, Y) vertex in 2D) along with two constants corresponding to maximum vertical and maximum horizontal tolerance (in pixels). To carry out the comparison of the contours obtained with the values associated with the expected positions of characters the bounding box of each contour was obtained, using the lower left vertex of the box for defining the position of the contour. In order to determine whether a contour is acceptable, its position is contrasted with the values associated with the expected positions of characters, considering a boundary associated with the vertical and horizontal tolerance. For example, assuming that (U, V) is the position of a candidate contour, and that the positions of the six expected characters are $\{(X_1, Y_1, X_2, Y_2), \ldots, (X_6, Y_6)\}$, and the vertical and horizontal tolerance are dh and dv, we define acceptability of a contour as follows:

$$(U,V) \text{ is accepted iff } \begin{cases} U \in (X_i - dh, X_i + dh) \\ V \in (Y_i - dv, Y_i + dv) \end{cases} \quad \text{for some } i = 1, 2 \ldots 6$$

Only if all contours are accepted as valid, the rectangle is accepted as a license car plate, proceeding with the character segmentation stage.

2.2. Character segmentation

In this stage the detected symbols in the previous step were extracted and normalized into a predefined size. To achieve this, the bounding box obtained for each character is used to perform the extraction. This simply involves creating a new image based on the bounding box and normalizing it to a specified size (15×30 pixels for the ANN and 40×80 pixels for ITM). At this point we are able to tell apart which elements are letters and which digits. As in the previous stage, the position of symbols was relevant as in Argentinean license plates the first three segmented characters correspond to letters and the next ones to digits. Finally, a new image is obtained from each character that will be used as an input for the next stage.

2.3. Character identification

This last stage correspond to the OCR process, where different alternative methods can be applied. As explained before, ANN are a usual choice for OCR on the basis of their high accuracy and noise

[3] Image warping is the process of digitally manipulating an image such that any shapes portrayed in the image have been significantly distorted. Warping may be used for correcting image distortion as well as for creative purposes (e.g., morphing) (OpenCV, 2011).

tolerance. In the current implementation of FGL, the OCR process was implemented instead of using the proposed ITM approach.

3. Contrasting ANN and ITM for performing character identification in the FGL system

As explained before, the FGL system involves several components or modules. In particular, the module for OCR can be implemented using different algorithms. In this section we will contrast a traditional ANN implementation for OCR with our proposal for intelligent template matching.

3.1. OCR using ANN

The use of ANN for solving OCR is broadly accepted as an alternative in pattern recognition, as ANNs present a strong tolerance to noise in the input data, resulting in a robust behaviour of the system (Mitchell, 1997). This favors performing character recognition in conditions which are far apart from ideal situations. This implies training time and the complex refinement of the associated parameters, requiring as well to have a large number of training examples, corresponding to different situations (i.e. pictures under different weather conditions, angles, etc.).

In our case, character recognition was carried out using two ANNs, corresponding to letters and digits, respectively. We chose to use a Layered Feed-forward Backpropagation Network composed by three layers: input layer, hidden units layer and output layer (Fig. 3a). Next we discuss the more salient details associated with each of these networks:

3.1.1. Input neurons

The input of both networks was defined by the pixels of the segmented character. In this case, we started from a 15×30 pixel image, resulting in 450 input neurons for each network.

3.1.2. Output neurons

There were 26 output units for the ANN for letter recognition, and 10 output units for digit recognition.

3.1.3. Hidden layer

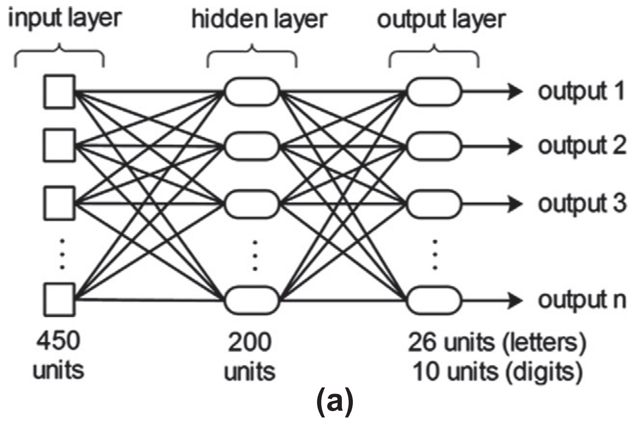
Neurons corresponding to the hidden layer were approximated in an experimental way. As a rule of thumb, the number of neurons in the hidden layer was assumed to be in a range between the number of input units and output units. In both ANNs we considered 50–450 units (with a step of 50), looking for an error less to 0.001. The best results were obtained with 200 units.

3.1.4. Activation function

For both the hidden layer as well as for the output layer the classical sigmoid function was employed.

3.1.5. Training

We decided to use a modified backpropagation algorithm (see details in Section 4) as a default algorithm. This algorithm is adaptative, not requiring to choose a particular learning rate. The used library implements the iRPROP (Igel and Husken, 2000) which is a modification of RPROP (Riedmiller and Braun, 1993). As a threshold value for the training phase an error of 0.001 was used. The training data involved characters extracted from more than 100 license plates. As not all letters appeared with the same frequency in the dataset, the training for some of them was less complex than for others (see Fig. 3b). In Section 5 we discuss some aspects related to this situation.



A	B	C	D	E	F	G	H	I	J	K	L
56	67	29	47	29	45	51	37	56	11	12	10
M	N	O	P	Q	R	S	T	U	V	W	X
5	9	23	8	13	8	26	31	15	9	14	16
Y	Z	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
9	19	66	44	35	57	78	106	98	37	46	75

(b)

Fig. 3. (a) Layers in the ANN and (b) number of training examples used.

3.2. OCR using ITM

The ITM method is based on a simplification of classic Template Matching (TM) but with some enhancements. TM in its general form consists on moving a template along an image evaluating its coincidence (Seul and O'Gorman Lawrence, 2000). In our case, it is not necessary any template displacement since we compare two character images of the same size. One of the major drawbacks of TM is its computing cost, which has two origins (Brunelli, 2009): (a) the need for using multiple templates and (b) their resolution (bigger images imply more pixels and then more processing requirements). In order to cope with the first problem, our proposal is based on the use of search trees (Fig. 4a), which use particular features of characters to prune the number of comparisons needed. Concerning the second problem, our method utilizes special templates, instead of using the whole character image as a template ITM uses only the corresponding skeleton.

In order to define search trees, salient features for letters were identified. In a first instance we analyzed whether they had inner contours or not (Fig. 4c). This way we could differentiate letters such as A, D, P from those such as C, I or T. Then, in case the letter at issue had inner contours, the size of such contours was analyzed, in order to distinguish from two possible groups (A, P, R and D, O, Q). It must be noted that it was also possible to single out a particular letter (B) in case of detecting two inner contours. In case no contours were detected, a different feature was analyzed, associated with the generation of inner contours when drawing a horizontal or vertical line through the character pixel matrix.

In the first place, a vertical segment was drawn, analyzing which characters came out with contours (Fig. 4d). For those characters not having contours, a horizontal segment was drawn

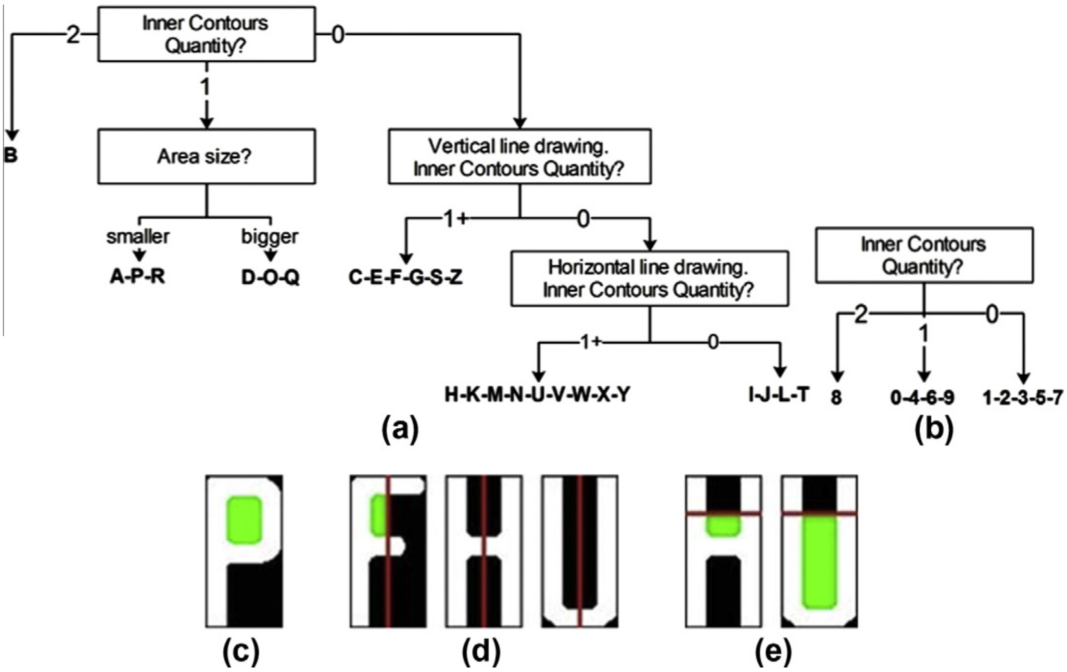


Fig. 4. Search tree for letters (a) and numbers (b). Examples of letter with inner contour (c) and cases where vertical (d) and horizontal (e) line drawing is needed to detect inner contours.

(Fig. 4e). The latter was done in the upper part with a proportion of 1/8 of the segmented image. This decision was based on the fact that for several candidate characters a contour can be found with this proportion (for example, letters H, K, M, U). In the case of search trees for digits, it was only necessary to take into account the inner contours to tell apart the different groups (Fig. 4b) before making the comparison with the appropriate skeletons.

At the last stage the templates comparison must be done. In the classical approach of TM the similarity evaluation between two templates was considered using the square distance or the cross correlation. These computations increase the processing requirements, so that the use of binary images allows some simplifications by directly making the comparison of pixels (Russ, 2007). In order to obtain better accuracy and robustness, in the present method we propose the mentioned comparison with skeletons, based on the superposition of the segmented character with those skeletons corresponding to the candidate character. To achieve this, it is necessary to have the images of the skeletons of all possible characters. In this case, we applied images of 40×80 pixels, so that the segmented character should be normalized to this window size. The symbol to identify would correspond to the one that coincides in a larger proportion between the segmented character and the base skeleton, in a similar way as performed in the classic template matching approach. The benefit of this strategy is that in general the skeleton that corresponds to the character will coincide practically in the total number of the pixels involved. Thus, for example, in the case of the letter C, the comparison would be associated with the group C, E, F, G, S, Z, where clearly the matching would be exact only for the corresponding letter (Fig. 5). In such cases it is unnecessary to carry on with the comparison, as an exact matching suffices to end the character recognition process. This also makes necessary to have some precautions when performing comparisons, as the order in which they are made is relevant based on the *inclusion* of some characters with respect to others (e.g. letter F is totally subsumed by E, so that the recognition strategy must first take F into account to find an exact match; only if this fails, then E should be considered).

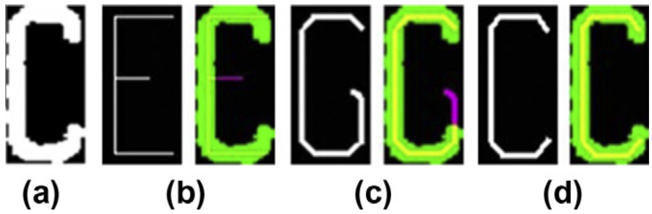


Fig. 5. Example of segmented letter (a), superimpose with incorrect skeletons (b) (c) and the correct one (d).

4. Results

Experiments have been performed to test both approximations (ANN and ITM) with input images of different sizes (from 200×400 to 2292×1944). Both methods were implemented with C, using OpenCV library for the image processing and FANN library for ANN. System run on a Pentium Dual-Core, 1.73 GHz, 2 GB RAM, Windows XP OS. Initially both methods were tested with 73 vehicle images used for the character extraction in the net training. The ANN correctly identified 99.09% characters and 91.10% for ITM method (Table 1).

Then the FGL system was evaluated with another image set having standardized environment conditions (distance, illumination, weather conditions, no damaged license plates, etc.). The global character recognition was 91.22% for ANN and 98.33% for ITM. Finally a set of pictures having all expected conditions away from ideal cases was used. About 88.50% of car plates were detected, obtaining a global character recognition of 85.88% and 91.53% for ANN and ITM respectively. The average time of whole identification process was 800 ms. Individual character recognition was 0.5 ms with neural networks and 0.75 ms with ITM.

5. Discussion

The results presented in the previous section show that ITM has an excellent detection rate for plate identification, showing as well

Table 1
Experimental results.

	From training set		Ideal environment		Real environment	
	Detect.	%	Detect.	%	Detect.	%
Plates Identified	73	–	150/150	100	177/200	88.50
ANN Letter OCR	216/219	98.63	380/450	84.44	412/531	77.59
ITM Letter OCR	197/219	89.95	438/450	97.33	464/531	87.38
ANN Digit OCR	218/219	99.54	441/450	98.00	500/531	94.16
ITM Digit OCR	202/219	92.24	447/450	99.33	508/531	95.67
ANN Global result	434/438	99.09	821/900	**91.22**	912/1062	**85.88**
ITM Global result	399/438	91.10	885/900	**98.33**	972/1062	**91.53**

Note. Number of letters and digits are based on plates found (in the first column the number of cases 219 is obtained from 73 plates) (as there are three letters/digits per plate). Global score is the sum of the two sets (438 = 219 × 2). The global recognition rate for both ANN and ITM is shown in boldface.

a robust behavior in a real-world environment. However, we can identify two kinds of images where our approach presents some limitations. In the case of badly damaged characters, with no clear separation between them, detection becomes nearly impossible. Fig. 6a and b shows how this problem affects the contour search, obtaining the union of two contiguous characters. To solve this issue, more complex segmentation methods should be used but surely at a performance cost. It must be noted that the FGL system uses the morphological operation erode–dilate to remove joints between characters that are not very severe.

The other group of images for which plate detection is difficult corresponds to car plates with black plastic protections, as this coating prevents the detection of the white outer frame and inner black rectangle. This problem could be solved by using the inverted threshold in order to detect the rectangle from the plate.

Considering character recognition, both approaches (ANN and ITM) show a high accuracy degree. However, as discussed in the previous section, the accuracy of the ANN is clearly dependant on its training. This becomes visible in tests based on images of the training set, where the recognition rate is as high as 99.09%. In the set of images that does not belong to training set, this value is considerably lower (91.22%), and even more when pictures are far from ideal conditions (85.88%). This feature highlights a great advantage of ITM, namely the fact of requiring no previous training. Indeed, our ITM approach shows recognition rates comparable to those achieved by ANN, and even better in the second set of

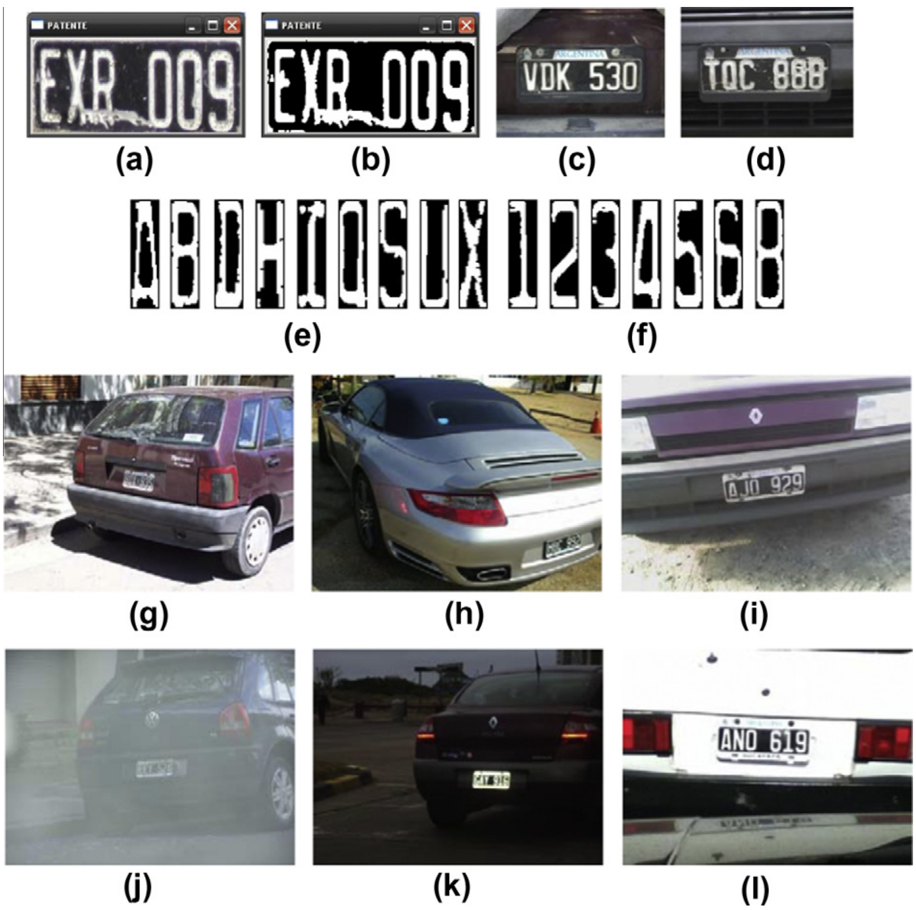


Fig. 6. Image of deteriorated extracted plate (a) and with threshold applied (b). Pictures of car plates with plastic protections (c and d). Examples of letters (e) and digits (f) quality, where identification were successfully done. Pictures correctly identified which differ substantially from ideal cases (g–l).

images (with 98.33%). In a real-world environment (last column in Table 1), the recognition rate of ITM is still above 90%. This limitation and other kinds of problems, like expected confusion of characters, i.e. E-F, U-V or O–Q, could be fairly corrected for neural networks using more training examples.

Considering execution time, ANN perform the classification process faster than ITM. However, we must consider that neural networks use 15×30 pixels character size instead of 40×80 pixels of ITM. This size difference represents about 85% higher pixel quantity, giving more processing load to skeletons comparison.

Now, taking into account the execution time of the whole car plate recognition process, our framework needs an average time of 800 ms. But we must consider the pictures used for testing, with big size in many cases (e.g. 1600×1200 pixels) and far from ideal conditions forcing the system to evaluate a lot of thresholds and rectangles search. Even more, if the plate is not recognized in bigger images, it implies nearly 1 sec. of processing time. However, our system shows a very fast performance for smaller images (800×600 pixels) which do not correspond to extreme pictures (i.e. poor illumination or shadows), detecting license plates in less than 200 ms.

Another benefit of ITM method is its flexibility and transparency, as it is totally changeable and configurable to recognize different character sets. In this way our method avoids the need of additional training as is the case of ANN.

Finally we can mention that ANN do not add so much overload to the segmentation method because of their robustness. Indeed, the high tolerance of ANN allows them to work with poor quality characters. The same result is obtained with ITM approach. As we can see in Fig. 6e and f, segmented characters are highly degraded, but this does not prevent their identification for any of the two strategies.

6. Related work

There are several papers in ANPR research which focus on particular aspects of license plate recognition, such as license plate localization, character segmentation or the final stage concerning the OCR process itself (Parasuraman and Kumar, 2010; Hamidreza Kasaei and Mohammadreza Kasaei, 2010; Khalil, 2010; Ho and Pun, 2010; Almustafa et al., 2011). Some recent research works have been focused on plate extraction, such as (Parasuraman and Kumar, 2010), reporting recognition rates as high as 98%, but with no clear information about the conditions of pictures used. Our approach is below that rate in real-world environment conditions, but in case of almost ideal pictures shows an even higher accuracy of almost 100%. It is also important to mention that the main effort in our research was not in the extraction stage. Moreover, our system performs well in pictures with extreme angles (Fig. 6g–h), shadowed plates (Fig. 6j), poor illumination conditions (Fig. 6k and l) or deteriorated characters (Fig. 6i).

In case of the OCR stage, different authors such as (Hamidreza Kasaei and Mohammadreza Kasaei, 2010,,) mention an accuracy of 96%, 90% and 92%, respectively. As in the previous analysis, there is no clear description of the quality of the images used. In some cases, some individual aspects are analyzed (such as illumination, weather or plate conditions), but in general not all of them. In our case, we have presented a method conceived to be robust in order to deal with deteriorated characters having an accuracy of 99% and 90% in an ideal and a real-world environment respectively. In (Almustafa et al., 2011) noise is added to numbers to test robustness of their system, but nevertheless they do not use real noisy extracted characters as in the case of our approach.

It must be noted that some research has been focused on analyzing plates with more than one row (see e.g. (Khalil, 2010,)). Our system is totally adaptable to such analysis, as symbol search can be performed inside of plate. In the same way, to deal with other kinds of alphabets (e.g. letters different to Latin characters) we can adapt the search trees and skeleton templates used in our system to cope with this.

7. Conclusions

In this article we have presented a novel approach to solve the OCR problem using intelligent template matching in the context of solving the ANPR problem for Argentinean license plates. As discussed before, ANN are commonly used for OCR due to their tolerance to noise on the input data, with the limitation of requiring considerable training. We introduced a new method for OCR, called ITM, with no need of prior training, obtaining a performance in time and accuracy comparable to ANN, while keeping the flexibility and simplicity of original template matching method.

Our experiments have shown that the results obtained by applying ITM yield more than 90% overall recognition accuracy. The flexibility of the proposed approach leaves several paths to be explored in a future research, such as the possibility of improving the overall performance by using smaller skeletons sizes, or testing more efficient strategies for the evaluation of character features. Other topics for future research include improving the segmentation of characters when there is no clear separation between them and the localization of car plates with plastic protections.

References

Almustafa, K., Zantout, R., Obeid, H., Sibai, F., 2011. Recognizing characters in saudi license plates using character boundaries. In: Internat Conf. on Innovations in Information Technology (IIT), pp. 415 –420.

Alvarez, S., Sotelo, M.Á., Ocaña, M., Llorca, D.F., Parra, I., Bergasa, L.M., 2010. Perception advances in outdoor vehicle detection for automatic cruise control. Robotica 28, 765–779.

Arnold, R., Miklos, P., 2010. Character recognition using neural networks. In: 11th Internat Symposium on Comput. Intell. and Informatics (CINTI), pp. 311–314.

Bailey, D., Irecki, D., Lim, B., Yang, L., 2002. Test bed for number plate recognition applications. In: Proc. of the First IEEE Internat. Workshop on Electronic Design, Test and Applications, pp. 501–503.

Brunelli, R., 2009. Template Matching in Computer Vision. Wiley.

Caner, H., Gecim, H., Alkar, A., 2008. Efficient embedded neural-network-based license plate recognition system. IEEE Trans. Veh. Technol. 57, 2675–2683.

Dudarin, A., Kovaccic, Z., 2010. Alphanumerical character recognition based on morphological analysis. In: IECON 2010 – 36th Annual Conf. on IEEE Industrial Electronics Society, pp. 1058–1063.

Ganapathy, V., Lean, C., 2006. Optical character recognition program for images of printed text using a neural network. In: IEEE Internat. Conf. on Industrial Technology, ICIT 2006, pp. 1171–1176.

Garcia-Osorio, C., Diez-Pastor, J.F., Rodríguez, J.J., Maudes, J., 2008. License plate number recognition – new heuristics and a comparative study of classifiers. In: 5th Internat. Conf. on Informatics in Control, Automation and Robotics (ICINCO).

Hamidreza Kasaei, S., Mohammadreza Kasaei, S.A.K., 2010. New morphology-based method for robust iranian car plate detection and recognition. Int. J. Comput. Theory Eng. 2, 264–268.

Ho, W.Y., Pun, C.M., 2010. A macao license plate recognition system based on edge and projection analysis. In: 8th IEEE Internat. Conf. on Industrial Informatics (INDIN), pp. 67–72.

Igel, C., Husken, M., 2000. Improving the Rprop Learning Algorithm. In: Proc. of the 2nd Internat. Symposium on Neural Comput. pp. 115–121.

INFOBAE, 2011. digital newspaper. <http://www.infobae.com/notas/585721-En-diez-anos-se-duplico-la-cantidad-de-automoviles-en-circulacion.html>.

Kasturi, R., On++Gorman, L., Govindaraju, V., 2002. Document image analysis: a primer. Sadhana 27, 3–22.

Khalil, M.I., 2010. Car plate recognition using the template matching method. Int. J. Comput. Theory Eng. 2, 683–687.

Lotufo, R., Morgan, A., Johnson, A., 1990. Automatic number-plate recognition. In: Proc. of First IEEE Internat. Workshop in Electronic Design V01.035, pp. 6/1–6/6.

Mani, N., Srinivasan, B., 1997. Application of artificial neural network model for optical character recognition. In: IEEE Internat. Conf. on Systems, Man, and Cybernet., 1997. 'Computational Cybernetics and Simulation', vol. 3. pp. 2517–2520.

Mitchell, T.M., 1997. Machine Learning. McGraw Hill.

OpenCV, 2011. Opencv project on sourceforge. <http://sourceforge.net/projects/opencvlibrary/>.

Parasuraman, K., Kumar, P., 2010. An efficient method for indian vehicle license plate extraction and character segmentation. In: 2010 IEEE Internat. Conf. on Comput. Intelligence and Comput. Research.

Parra, I., Sotelo, M.Á., Llorca, D.F., Ocaña, M., 2010. Robust visual odometry for vehicle localization in urban environments. Robotica 28, 441–452.

Riedmiller, M., Braun, H., 1993. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In: IEEE Internat. Conf. on Neural Networks, pp. 586–591.

Russ, J.C., 2007. The Image Processing Handbook. Taylor & Francis.

Seul, Michael., O'Gorman Lawrence, S.M.J., 2000. Practical Algorithms for Image Analysis: Description, Examples, and Code. Cambridge University Press.

Shah, P., Karamchandani, S., Nadkar, T., Gulechha, N., Koli, K., Lad, K., 2009. Ocr-based chassis-number recognition using artificial neural networks. In: IEEE Internat. Conf. on Vehicular Electronics and Safety (ICVES), pp. 31–34.

Wang, W., 2009. License plate recognition algorithm based on radial basis function neural networks. In: Internat. Symposium on Intelligent Ubiquitous Comput. and Education, pp. 38–41.

Wenjie, H., 2010. Automatic vehicle license plate recognition system used in expressway toll collection. In: 3rd IEEE Internat. Conf. on Comput. Science and Information Technology (ICCSIT), pp. 159–162.

Xia, H., Liao, D., 2011. The study of license plate character segmentation algorithm based on vetical projection. In: Internat. Conf. on Consumer Electronics, Communications and Networks (CECNet), pp. 4583–4586.