ISSN 1330-3651
UDC/UDK 004.932.75'1:629.3.066.8

CHARACTER RECOGNITION BASED ON REGION PIXEL CONCENTRATION FOR LICENSE PLATE IDENTIFICATION

Krešimir Romić, Irena Galić, Alfonzo Baumgartner

Original scientific paper

Since it has become possible to perform digital image processing in a short period of time, its usage in technical systems is getting more common. Automatic license plate recognition is one such example. By using digital image processing it is possible to automatically detect and recognize characters on vehicle license plates. The steps taken in this process are image preprocessing, plate detection, character segmentation and recognition. This process is performed by the algorithm which takes the digital image as an input and gives textual form of license plate characters as an output. There are several methods used to perform this process. These methods are explained in this paper and one of them is implemented in C# programming language. The results show that the algorithm works fine in cases without much deformation of an input image. However, there are still cases where unpredictable nature of an input image can cause unsuccessful detection or recognition.

Keywords: *character recognition, detection, digital image processing, region pixel concentration, segmentation*

Prepoznavanje znakova temeljeno na područnoj koncentraciji piksela u svrhu identifikacije registarskih oznaka

Izvorni znanstveni članak

Budući da je postalo moguće izvoditi digitalnu obradu slike u kratkom vremenu, njezina uporaba u tehničkim sustavima postaje sve češća. Automatsko prepoznavanje registarskih oznaka jedan je takav primjer. Koristeći digitalnu obradu slike moguće je automatski detektirati i prepoznati znakove s registarske oznake vozila. Koraci u ovom procesu su predprocesiranje slike, detekcija pločice, segmentacija i prepoznavanje znakova. Ovaj proces obavlja algoritam koji uzima digitalne slike kao ulaz i daje tekstualni oblik znakova s registarskih oznaka kao izlaz. Postoji nekoliko metoda koje se koriste za izvođenje ovog procesa. Ove metode su objašnjene u ovom radu i jedna od njih je implementirana u C# programskom jeziku. Rezultati pokazuju da algoritam radi dobro u slučajevima bez puno deformacija ulazne slike. Međutim, još uvijek postoje slučajevi u kojima nepredvidiva priroda ulazne slike može uzrokovati neuspješnu detekciju ili prepoznavanje.

Ključne riječi: *detekcija, digitalna obrada slike, koncentracija područja piksela, prepoznavanje znakova, segmentacija*

1 Introduction

Character recognition is a broad term and consequently its application is possible in different areas and in different ways. In license plate character recognition, the complexity of the process enables the use of different methods or even combinations of methods to solve the problem. Unpredictable nature of input data (digital images) makes it difficult to choose a universal method that would be appropriate for every input image. Often the advantages of a method in one area are also disadvantages in the other.

Automatic license plate recognition is a process with the main aim of getting textual form of license plate characters from digital image of a vehicle. Digital image as an input passes through several stages of processing to obtain the string of characters as a final result. The core of this process is algorithm for optical character recognition (OCR), [1].

Optical character recognition process has become very common in technical systems since the speed and performance of today's computers allow fast digital image processing. It also represents the main part of the automatic license plate algorithm implemented in this paper. Apart from the mentioned process, several other actions are necessary to perform the whole process of license plate identification. This primarily refers to image filtering, license plate area detection and image segmentation.

There are many different ways to solve such problems. This paper implements some new ideas in this kind of character recognition process. The ideas consider additional threshold methods for plate detection and region pixel concentration as main sources of improvement. Our main goal is to achieve successful recognition of license plate characters with less influence of input image deformation.

Algorithms like those are increasingly used in technical systems in transport. The most common examples of its usage are traffic monitoring, police investigations and identification on different kinds of entrances [2].

Since license plates of different countries vary in shapes and styles, algorithms are often tailored for the specific type of license plate. The proposed algorithm primarily works for Croatian license plates, but is applicable to similar plate shapes and styles. Furthermore, the algorithm is modifiable to work with most international plates.

1.1 Related work

Thresholding is almost inevitable in the character recognition process. It is also used in this paper, but including the version with some additional modifications.

This paper is based on methods with structural matching of characters. Structural matching means recognition of characters based on their structural features and structural differences between each other.

Initial method of grouping which depends on the number of end points is proposed earlier in [3]. This procedure requires additional time but it also shortens the entire further recognition process which is here based on differences in region pixel concentration of each character.

2 Method

This method consists of four steps. These steps are: pre-processing of an image, license plate detection, character segmentation and character recognition.

2.1 Preprocessing

The first step refers to processing and preparation of an image which is necessary for further license plate detection and character recognition. Pre-processing involves the application of digital filters to an image.

Every colour image is first converted into a grayscale image to preserve memory and speed up further processing. This does not affect the useful data from the image.

Threshold is used to accentuate license plate area. This is performed with an experimentally determined threshold value 170. After this step every pixel with a value larger than 170 becomes white (gets value 255) and every pixel with a value smaller than 170 becomes black (gets value 0). As a consequence, the grayscale image converts into a binary image with only two possible pixel values, black and white. Since the license plate is made of two contrasting colours, characters on it will remain visible [4].

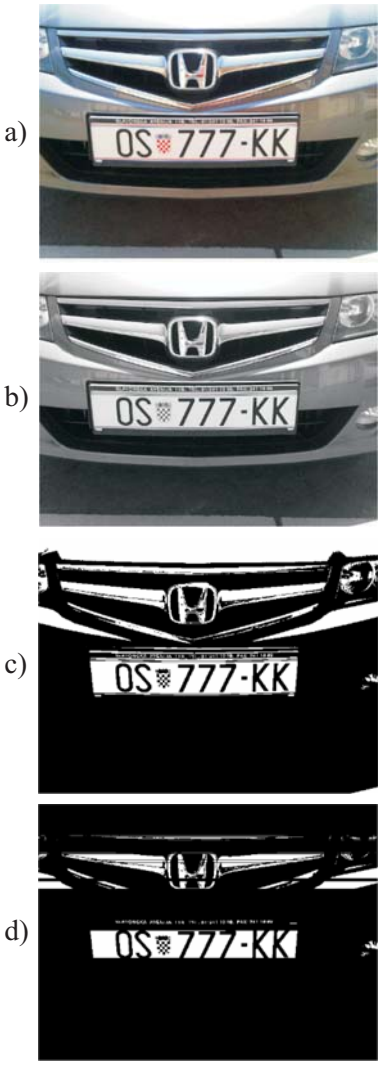

Figure 1 a) Colour image, b) Grayscale image, c) Thresholding, d) Additional thresholding

A different type of threshold is applied in order to improve license plate area detection. Our idea is to count transitions from black to white pixels and conversely in each pixel row of the image. The array of the same pixel values must be at least 3 pixels long before a transition to avoid error caused by impurity of the input image. By using the number of these transitions, it is possible to exclude unnecessary parts of the image. If every license plate has at least 7 characters, there will be at least 14 transitions in rows within the license plate area. Therefore, each row with less than 14 transitions is not part of the plate area and contains unnecessary information. The purpose of additional thresholding is to black out these rows with useless data for easier further detection. Fig. 1 shows the steps in pre-processing of an image.

2.2 Detection

According to the new idea of additional thresholding entirely black pixel rows appear repeatedly in the image after pre-processing. The white license plate area is situated somewhere between those black rows. By finding the longest vertical array of white pixels, it is possible to detect the left and the right edge of the license plate. When analyzing the image from left to right, the first longest vertical array of white pixels represents the left edge of the license plate. Accordingly, the last white column of the same size represents the right edge of the license plate. Positions of these license plate edges are sufficient to detect coordinates of the license plate, as well as the height and width as shown in Fig. 2.

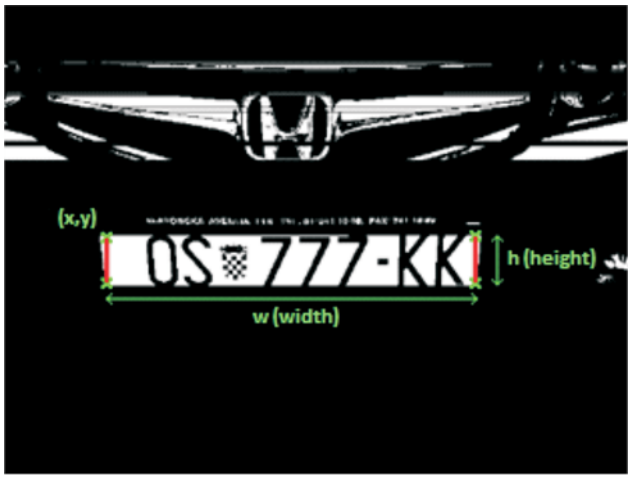

Figure 2 Coordinates of the license plate area

It is necessary to perform an additional check of the aspect ratio. It is known that the height of license plates should be about 6,5 times smaller than the width, so the aspect ratio should be 1:6,5. If the aspect ratio of the detected area does not meet this value with a certain degree of tolerance, which is in this case set to 10 %, the area is rejected and a new search begins. This additional check prevents the algorithm from detecting the wrong area of an image as a license plate area. Once the correct area is detected, the license plate image is saved and all further work is performed on it.

Before segmentation it is useful to filter the license plate image to remove isolated black pixels or smaller groups of them caused by license plate impurity or low image quality. This problem is solved with median filtering which removes spot noise and rounds off edges [1].

2.3 Segmentation

The next step is segmentation of the license plate area into smaller parts which represent each character of the

license plate. This is done using the vertical projection which is shown in Fig. 3. Vertical projection of a binary image looks like a set of black hills on a white surface. This is obtained by counting the number of black pixels in each column. Columns without black pixels represent the spacing between each character. Coordinates of each character are then determined with alternatively found left and right hill edge [5].

The whole image is divided into smaller images. These images have the same height as the license plate image, but the width varies depending on the width of each character.

Figure 3 Vertical projection (b) of a binary license plate image (a)

2.4 Recognition

The process of character recognition is repeated for each character image obtained in the last step. This process could be carried out in several steps. The output of this process should be a recognized character. The set of possible outputs are characters that appear on license plates, which are letters of the alphabet, numbers from 0 to 9 and special characters like a dash.

In order to simplify recognition, the initial step is to separate possible outputs into smaller groups counting the character end points [3]. There are 6 possible options for each character as shown in Fig. 4. It can have 0 to 5 end points and according to this fact characters are being allocated into groups. For example, character 'H' has 4 end points, but on the other hand, character '9' has only 1 end point.

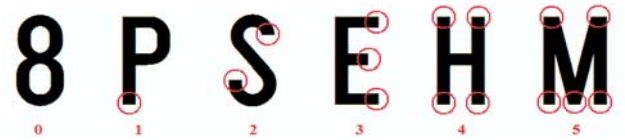

Figure 4 Characters with different number of end points

The fundamental action for counting end points is to make the characters 1 pixel thin. The algorithm which makes the character thin is called thinning. It observes the image pixel by pixel and erases outer layer of black pixels on every character. The image is observed repeatedly until every character is reduced to single pixel thickness, as if it is made of lines (Fig. 5) [7].

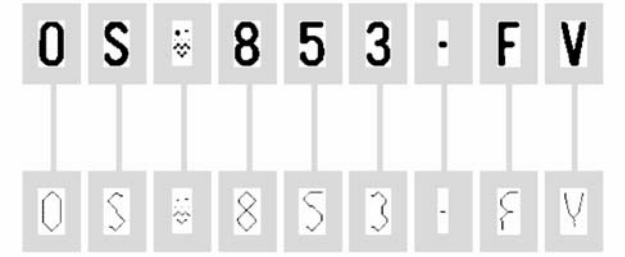

Figure 5 Characters after the application of thinning algorithm

The previous thinning algorithm allows us to use a 3×3 mask to find one of 8 possible types of end points. Every 3×3 area of an image is compared to 8 masks shown in Fig. 6. When the mask coincides with an area, the counter increases the number of endpoints. Sorting the characters by the number of end points is shown in Tab.1.

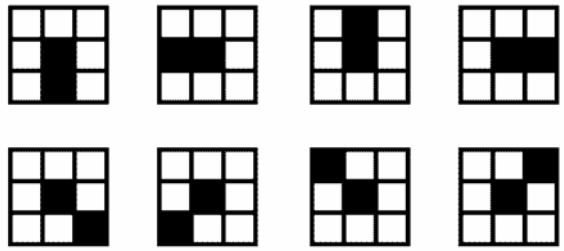

Figure 6 Possible types of end points

Table 1 Characters sorted by the number of end points

End points	Characters	End points	Characters
0	0, 8, B, D, O	3	3, A, E, F, T, V, Y
1	6, 9, P	4	H, K, N, X
2	1, 2, 4, 5, 7, C, G, I, J, L, R, S, U, Z	5	M,W

After initial group assignment, the character image is observed through characteristic areas which are shown in Fig. 7. These areas are horizontal and vertical thirds and ninths. The concentration of black pixels in these regions varies on different characters. To calculate the concentration, a function which gets the coordinates of the region, as well as its width and height as parameters, is used. These parameters are shown in Fig. 8 based on the example of character 'R'.

Figure 7 Characteristic areas: vertical thirds (a), horizontal thirds (b), and ninths (c)

The main part of our new algorithm is logic used for the purpose of comparing the regions by the concentration of black pixels. By comparing the concentration between different areas of the character image, we can make rules for further grouping. The logic varies in each initial group of characters, but it always starts by comparing the thirds and later, if needed, by comparing the ninths.

For example, letter 'P' with one end point has more black pixels in the first vertical area than in the third one. On the other hand, number '9' also with one end point, has more black pixels in the third vertical area. So, characters like those will be separated into different groups.

To get the single character from the initial group of possible characters, several rounds of comparing and subgrouping are needed. Recognizing the character from the image means reducing the initial group to only one member.

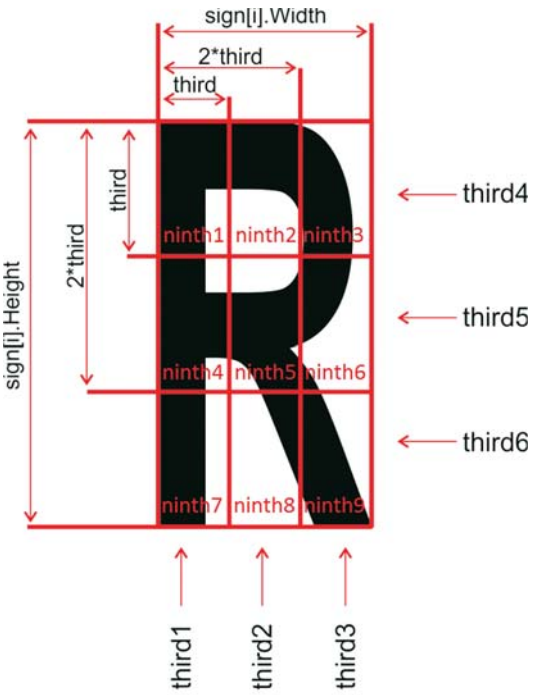

Figure 8 Parameters for character regions

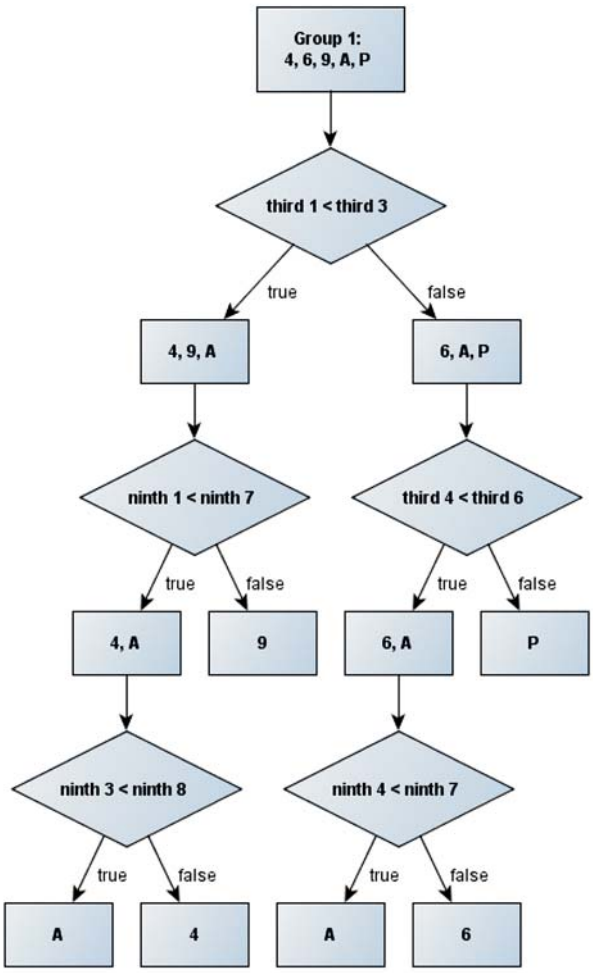

Figure 9 Flow chart for the recognition process of characters with 1 end point

An example of these procedures for the distribution of characters with one end point is shown in Fig. 9. Compared with Tab. 1, it can be noticed that the initial group of characters with one end point is extended with characters '4' and 'A'. This is done for safety reasons, because experimental results showed that in some cases the thinning algorithm can make those characters as if they have one end point. If we compare the first and the last vertical third in Fig. 9, the initial group of possible characters (4, 6, 9, A, P) is divided into two smaller groups (4, 9, A and 6, A, P). Again, due to safety reasons, if the comparison of thirds or ninths does not clearly reveal differences, the character is placed in both groups. The flow chart illustrates the whole comparing logic and the rounds of subgrouping.

After the recognition process for each segmented character, a complete string of license plate characters is pointed as a result.

3 Algorithm implementation

Implementation is carried out using C# programming language and integrated development environment Microsoft Visual Studio Express.

A graphic user interface is built for easier program testing through each phase from the previous section.
Each phase can be performed by clicking a separate button, while the whole process can be performed by clicking one single button. Images, obtained using this algorithm, are shown one after another in the image-boxes. In the end, the recognized license plate characters are presented as a string inside the label in the lower right corner.

Use of several integrated bitmap functions makes the work with images easier [6]. The functions are used for managing the pixel values and image cropping. The more complex functions described in previous section are coded and applied to image or part of image as needed.

4 Experiments

Experiments on a set of different images are performed to show the effectiveness of the implemented algorithm and used methods. The previously mentioned graphic user interface allows us to see intermediate results in each phase of the recognition process.

One example of the entire recognition process is described in detail in the previous method explanation. The following example shows the recognition process results for another loaded image. This test image shows a vehicle with a license plate in the natural environment. The proposed algorithm should be able to display license plate characters in a textual form as a final result, regardless of the bad factors (bright colours of the vehicle, impurity, background information, sun reflection).

The example shown in Fig. 10, presents the front side of a silver vehicle and some natural background. After grayscaling and first thresholding, the license plate area becomes prominent but not enough to isolate it. Additional threshold with a transition counter allows us to discard unnecessary information like white surfaces caused by sun reflection. The license plate area is then isolated correctly. Impurity between characters makes some noise problems which are solved by median filtering. Isolated license plate is segmented into individual characters. Finally, a correct string of license plate characters is obtained by the recognition algorithm.

Expected accuracy of implemented method is experimentally obtained with 50 instances of input images. Test images were taken in different conditions and show the

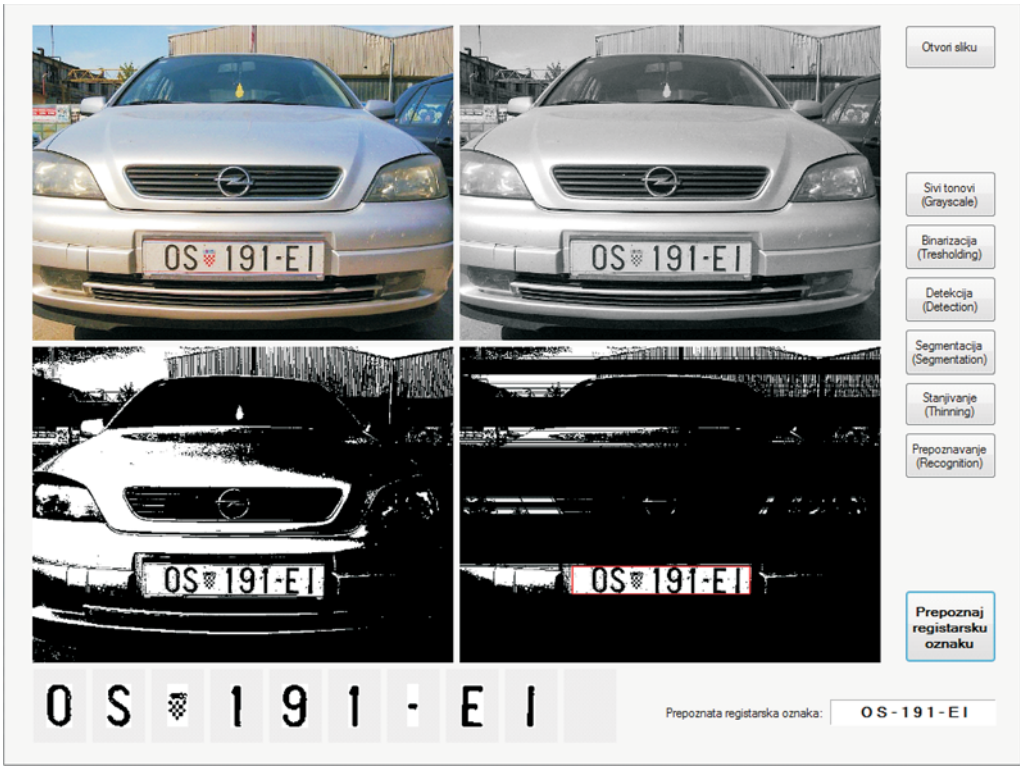

Figure 10 Example of successful recognition

different vehicles. Correct recognition of license plates appeared in approximately 80 % of cases and about half of incorrect recognitions has the lapse in only one character.

5 Conclusion

It is difficult to achieve a robust algorithm that would be good in all segments and all cases of character recognition without fault. However, a combination of different methods and maximum utilization of computer possibilities in the processing of digital images allow us to bring the performance of character recognition on a high level.

The algorithm implemented in this paper, except for a few common approaches in image processing, uses a combination of new ideas in detection and character recognition. These ideas refer to additional thresholding to make plate detection easier and use of characteristic thirds and ninths as a method for structural matching of characters. The experimental results show that the algorithm successfully recognizes characters in cases when there is no major angle distortion. Also, the algorithm avoids errors caused by noise or dirt on the license plate by applying an additional median filter. Although the recognition algorithm is based on structural characteristics of the characters, the font of characters may adversely affect the recognition result in case it varies from the font used on Croatian license plates which was the basis for writing the algorithm.

Under normal conditions, the algorithm performs its tasks successfully, but because of several shortcomings, there is still room for improvement and further development. This applies primarily to higher tolerance for input image deformation and increase of the number of recognizable font types.

6 References

[1] Gonzalez, R. C.; Woods, R. E. Digital Image Processing, Prentice Hall, New Jersey, 2008.

[2] The Automatic Number Plate Recognition Tutorial. Quercus Technologies. 2006. URL: http://www.anpr-tutorial.com (15.08.2006.)

[3] Fan, C. H.; Peng, Y. H. Vehicle License Plate Recognition System Design // Chung Hua Journal of Science and Engineering, 7, 2 (2009), 47-52.

[4] Burger, W.; Burge, M. J. Digital Image Processing, Springer, New York, 2008.

[5] Khalifa, O.; Khan, S.; Islam, R.; Suleiman, A. Malaysian Vehicle License Plate Recognition // The International Arab Journal of Information Technology, 4, 4 (2007), 359-364.

[6] Microsoft Developer Network Library - Bitmap Functions. Microsoft. 2012. URL: http://msdn.microsoft.com/en-us/library/dd183385(v=vs.85).aspx (07.03.2012.)

[7] Image processing learning resources, Morphology – Thinning. Hypermedia Image Processing Reference. 2003. URL: http://homepages.inf.ed.ac.uk/rbf/HIPR2/thin.htm (2003.)

Authors' addresses

Krešimir Romić, mag. ing. comp.
Faculty of Electrical Engineering
Josip Juraj Strossmayer University of Osijek
Kneza Trpimira 2B, HR-31000 Osijek, Croatia
e-mail: kresimir.romic@gmail.com

dr. sc. Irena Galić, prof.
Faculty of Electrical Engineering
Josip Juraj Strossmayer University of Osijek
Kneza Trpimira 2B, HR-31000 Osijek, Croatia
e-mail: irena.galic@etfos.hr

dr. sc. Alfonzo Baumgartner, dipl. ing.
Faculty of Electrical Engineering
Josip Juraj Strossmayer University of Osijek
Kneza Trpimira 2B, HR-31000 Osijek, Croatia
e-mail: baumgart@etfos.hr