License Plate Segmentation in Images Based on per-Block Contrast Analysis and CCA

Mihnea Horia VREJOIU

National Institute for Research and Development in Informatics - ICI Bucharest,
8-10 Averescu Avenue, Bucharest, 011455 Romania
mihnea.vrejoiu@ici.ro

Abstract: ALPR based applications are more and more used today. Besides the OCR part, the vehicle registration plate detection in real world images represents the main challenge in LPR. This paper presents a simple, yet quite general, fast and effective method for license plate (LP) segmentation. It is based on the evaluation of a local contrast (high gradient) measure at the level of image blocks, binarization of the downscaled contrast map obtained with these values, and analysis of connectivity between its runs, requiring modest CPU and memory resources. It provides as output not only the locations of detected LPs, but also associated bitmaps, black on white, containing only their constituent alphanumeric characters, aligned horizontally, with no slope, slant or tilt, and free of other parasitic noise. Such black on white bitmaps are directly suitable for further OCR, the correctness and completeness of final LPR strongly depending on the quality of the bitmap provided. Extended experiments carried out on own image set, as well as on other (public) data sets, showed good performance and results of the implemented method in the vast majority of situations, even on certain difficult, poor quality images. Comparison with *state-of-the-art* (based on deep neural networks, and high-end GPU parallel computing), also proved average good performance on public data sets complying with the minimal requirements of our method.

Keywords: Computer vision (CV), Segmentation, License plate detection (LPD), License plate recognition (LPR), Optical character recognition (OCR), Histogram, Threshold, Binarization, Bitmap, Run, Connected component analysis (CCA).

1. Introduction

Automatic license/number plate recognition (ALPR/ANPR) is now widely used in various Computer Vision based applications (Vrejoiu, 2010a; 2010b), mainly in the context of the Intelligent Transportation Systems (ITS), for:

- automatic control of vehicle access (in delimited areas, with selective access right);
- automatic taxation based on duration or distance (in parking lots or garages, crowded urban centers, or at toll collection points);
- statistical evaluations, traffic surveillance and management, monitoring and/or control (at border crossing points, at entries/exits on highways, on certain sections of roads);
- law-enforcement in cases of traffic rules violations (running a red light, speeding, stopping/parking in prohibited places, etc.);
- signaling the occurrence and/or monitoring the route of vehicles to be tracked, included in a “black list”, etc.

This paper addresses the most difficult part (not only in our opinion) of the processing chain involved in license plate recognition (LPR), the one responsible for detecting/localizing the LP as accurately as possible in the image.

The registration plates of the vehicles are made to be easily visible and readable under various conditions by the human eye and/or camera, being painted with good foreground-background contrast usually. In addition, the contained alphanumeric characters are written with constant thickness, have regular and reasonable shapes, sizes and spacing.

Thus, approaching the problem of license plate detection (LPD) in digital images based on these characteristics appears as a natural choice. Many solutions (see Section 2) have as starting point the properties of contrast and its distribution in the registration plate, aiming, through various image processing and analysis techniques, classical (Gonzalez & Woods, 2007), or newer (Xu et al., 2018), to detect the regions of interest (ROIs) with such properties, as candidates to contain license plates (LPs). A few less general approaches rely (also) on country specific colors of the LPs, but most of the times grayscale images are used. Some pre-processing filters/transformations are usually applied on these ones for: noise reduction (Gaussian, median, mean, etc.), contrast enhancement (histogram normalization or equalization), or edge detection (horizontal rank filtering, Prewitt, Sobel, Shen-Castan, Canny, Roberts, etc.). Various analyses of, texture (Gabor filters, etc.) horizontal and vertical projections or histograms, threshold based binarization, erosion/dilation, connected component analysis (CCA), and validation/filtering of candidate ROIs

https://doi.org/10.24846/v29i2y202005

based on various criteria, are performed for LP segmentation. In most situations, more or less drastic constraints, related to the image quality and resolution/sizes, lighting, location/framing, orientation, type, shape, aspect ratio, size and/or layout of LPs, are imposed, and customizations are required for different countries/regions and/or specific use cases. The algorithms are often quite costly in terms of computation complexity and time.

We aimed to create a method as general and robust as possible, yet also simple and fast, for segmentation/detection of various international vehicle LPs in digital images, able to directly cope, with no special customization from case to case, with a wide range of real world situations.

The developed algorithms perform the analysis of digital images acquired in either natural or infrared (IR) light, with no restriction on the image capturing equipment. Grayscale raster images (GI), thus converted in the case of color ones, with various acceptable resolutions/sizes, are used as input data. Sometimes, some pre-processing (smoothing, normalization, down-sampling, etc.) is applied on the GI, or only on certain regions in it. An iterative multiple-tries mechanism for locating the LP in the image was implemented. It is based on evaluating a local contrast (high gradient) measure along the image, on a per-blocks basis. A downscaled contrast map (CM) is thus obtained, which is further binarized, by using a histogram based computed threshold, and noise filtered. Contrast map blobs are "packed" by connectivity analysis of the runs of the binary CM. The corresponding bounding boxes of these blobs, re-mapped onto the GI, are giving the regions of interest (ROIs) that are likely to contain LPs. The final list of candidate ROIs is obtained after a sequence of adjustments and/or filtering of these ones (based on their aspect ratio, relative sizes, content, topological relationships, etc.). Some smart techniques have been used for adaptively binarizing each of the candidate ROIs, to cope with both the plates painted with dark writing on a light background and those with light writing on a dark background, as well as, to a certain extent, even affected by shadows, uneven lighting, or mud/dirt. A bitmap (BM) is thus obtained for each remained ROI most likely to contain a license plate. Also, smart algorithms have been developed and implemented to filter parasitic elements (borders, shadows, noise), as well as to compensate possible rotation, slant and/or tilt affecting the shape of characters in these BMs of the LPs.

The paper is further organized as follows. Some related works are overviewed in Section 2, while Section 3 details the requirements for the input images and the method and algorithm designed and implemented. In Section 4, the results obtained by experimenting with those described in Section 3 are presented and commented. Finally, some conclusions and possible future work are synthesized in Section 5.

2. Related Works

The segmentation method, subject of this paper, has its roots in a particular approach of 1993-94, co-authored with Dr. Eng. Dan Mugurel Onțanu, used by VEDA CAR™ engine. It has been later substantially reformed, generalized, improved and refined occasionally starting with 2006-2007, new version being used by our VEDA LPR engine (free testing demo download at *https://veda-mhv.github.io*). We concentrate below on some approaches from the respective period and of the same category as our method, as relevant comparison terms, beside other more recent techniques, including the *state-of-the-art*, based on the new technologies, as deep learning.

A project dealing with license plate location in raster images is described in (Parker & Federl, 1996). RGB color images are first converted to grayscale and enhanced by applying 5×5 median filter for noise removal, preserving sharpness. Then, Shen-Castan edge detection, adaptive thresholding based on the maximum difference between the values in a local neighborhood of each pixel, erosion-dilation and CCA are applied for getting candidate ROIs and their bounding boxes. These ones are filtered based on some dimensional, aspect ratio, and minimum number of characters criteria, and the remaining ones are assumed as possible LP's characters. A genetic algorithm (GA) is used for finding the LP, based on these bounding boxes and on the edge image.

A neural network (NN) based artificial vision system, VISICAR, is presented in (Draghici, 1997). Input 752×582 pixels grayscale image is initially enhanced by applying 2-D Gaussian filtering, and histogram stretching for contrast improvement. Enhanced image is horizontally scanned to find repeating contrast changes on a scale of 15 pixels. Areas with concentrations of high contrast gradients that can be approximated by a rectangle are kept as ROIs, and histogram stretching are performed on each one. Sobel horizontal and vertical edge detection is then applied on each ROI, the result is thresholded and smoothed using a 2-D Gaussian filter, and the horizontal and vertical projections on each binarized ROI are computed and smoothed with a Gaussian filter. This Sobel-threshold-Gaussian sequence is repeated for 9-16 cycles. Finally, the limits of the ROI are found by analyzing the lateral projections. 99% detection rate is claimed at 15 sec. per image, on 486/66MHz CPU.

In (Kwasnicka & Wawrzyniak, 2002), a combination of two independent methods is used, both relying on the contrast properties. Input RGB color images are changed to YUV space, and only the luminance channel Y is kept, as grayscale image, which is normalized, to minimize the effects of various environmental conditions and to increase the contrast. In the first method, it is then binarized as in (Parker & Federl, 1996), and CCA identifies some “spots”, as possible characters. A set of eliminating and grouping operations are applied. Neighbor spots are grouped in segments, which are grouped with their neighbors as possible rows of the LPs, segments with incorrect number of spots being discarded. A set of candidate ROIs is obtained. The second method is based on the detection of the LP’s “signature” (sequence of alternating minimum-maximum brightness). Every row of normalized grayscale image is scanned, and each part resembling to such signature is marked. Several operations of grouping and eliminating “spots”, now created this way, give an additional set of candidate ROIs. 233 640×480 color images of vehicles, with various lighting conditions, distances or angles of view, and LP from Poland, Germany, France, England, Russia and Croatia, were used in experiments. The detection rate reached 99%.

Probably the first approach based on texture for license plate detection (LPD), is proposed by (Kahraman et al., 2003). Grayscale image is convolved with 12 2-D Gabor filters (at 3 scales, 4 orientations), the resulted image being then thresholded. Neighbor regions are merged by morphological dilation, and ROIs are extracted by using CCA on 8-connectivity. Nonlinear vector quantization on the ROIs is used to eliminate the false ones. CCA is applied on the quantized binary images to get characters’ blobs. An expert system separates possible connected characters by using vertical projections of the blobs and other predefined rules, and filters out the candidate ROIs list, based on local and geometric criteria, and regular expressions of characters sequence in the LP. The output consists in the bounding boxes for the detected LP and for every segmented character within it. The experiments employed over 300 daylight and night images of vehicles from Turkey, Germany, Bulgaria, etc., of various resolutions between 512×384 - 768×576, and LPs areas of 173×37 - 211×47 pixels. Average computation time per image is 3.12 seconds on a P4/1.4GHz CPU, and depends on image sizes and contents. A 98% detection rate is reported.

The paper (Shapiro et al., 2004) presents the LP localization component of a multi-national *Car License Plate Recognition* (CLPR) system. Input grayscale images are pre-processed, being first rescaled to about 120 columns, keeping the aspect ratio, for reducing computation time. The resulted image is normalized, by substituting original intensity with its logarithm, for reducing the impact of various illumination conditions. By applying a Roberts vertical edge detector, an edge image is obtained. A horizontal rank filter is applied on the normalized image, each pixel being replaced with 80% of the average intensity in the area covered by the filter mask. Pre-processing stage ends up by obtaining a vertical projection that is smoothed with a 5-element uniform filter. A horizontal strip containing the LP is clipped from the image based on this vertical projection, on which a modified Radon transform, treated as Hough transform as in (Belean et al., 2017), is applied for detecting possible skew that is compensated, if any. The strip’s horizontal projection is convolved with a filter of estimated LP’s length, and candidate ROIs are clipped, relying also on the edge image. These are then filtered based on LPs’ geometric constraints (width, height, aspect ratio), and/or specific background/foreground. For binarization, Otsu’s thresholding algorithm is used. 120 images of Israeli and Bulgarian vehicles, with various lighting and contrast were used in experiments. Robustness in lighting, LP slope, scale and country is reported, except in cases of uneven lighting, alternating shadows and brightness. Detection rate reached 90%.

In (Anagnostopoulos et al., 2006), an algorithm for vehicle LP identification in grayscale images is proposed, using an adaptive segmentation technique called Sliding Concentric Windows (SCW), and CCA, for faster detection of ROIs in outdoor real conditions, in conjunction with a NN based OCR. LPs' areas are viewed as abrupt changes in the image texture. Local irregularities in the image are described by standard deviation and mean value. Two concentric windows are sliding over the whole image, consecutively centered on each pixel. If the ratio of the statistical measures in the two windows exceeds a threshold experimentally set, then the central pixel is considered to belong to a ROI. Optimal segmentation results are got if the aspect ratio of the concentric windows is near the one of the object to be segmented. Then, image masking, adaptive threshold based binarization by using Sauvola method, CCA on 8-connectivity, and filtering of resulted blobs based on certain measurements, are applied, resulting a final set of candidate ROIs. Experiments used 1,334 grayscale images of vehicles, from 5 sample sets, taken at various distances and angles, in real world, with various backgrounds and illumination. 96.5% of the LPs are detected, at about 111 msec. per image, on a P4/3GHz CPU.

Bachelor Thesis (Martinsky, 2007), presents LP detection in grayscale images in several steps. As in (Shapiro et al., 2004), horizontal and vertical Sobel edge detectors and rank filters are applied on the image, and horizontal and vertical projections are computed for allowing a two phases statistical analysis. In the first phase, a wider area of the LP is detected, which is deskewed, and processed in the second phase. The ROI of the LP is obtained after a "band clipping" by analyzing the vertical projection of the image, followed by a "plate clipping" by horizontal analysis of that band. Several LP candidates could be found, and are sorted based on a cost measure, determined for each one by with experimentally chosen heuristics. The best candidate is examined by a deeper heuristic analysis, which definitely accepts it, or rejects it and gets the next one. This latter type of analysis implies the LP segmentation in characters, being quite time consuming. Some collected images and software demonstrating principles from the thesis are publicly available (JavaANPR, 2006).

In (Moustafa & Jaradat, 2015) a method for LP detection based on vertical edge detection and growing window is proposed. Color RGB images are converted to YUV, and luminance channel Y is kept as grayscale image. This one is pre-processed first using a median filter for noise reduction, and histogram equalization for contrast enhancement. A Sobel vertical edge detector is then applied, and the resulted vertical edge image is binarized on a threshold basis. Row wise sum and variance are calculated, and based on them horizontal bands are iteratively identified as LP rows ROIs, and filtered using some dimensional rules. Edges are powered in each ROI based on their distance from neighbor edge(s), and location in ROI. The edges' powers are counted row wise in each ROI, and the ROI with maximum count is selected as a row bar candidate to contain the LP, its upper and lower limits being adjusted. Horizontal projection for this row bar is then calculated, for getting the horizontal limits of the LP. A growing window filter method in two stages is finally applied. 350 color images, normalized to a standard size of 704×576 pixels, with front and rear views of vehicles, were used for experiments. An average accuracy of 93.43% in LP detection is reported.

In recent years, the remarkable advances and performance attained with *deep neural networks* (DNN), particularly with *Convolutional Neural Networks* (CNN) and *deep learning* in Computer Vision (CV), using new powerful multiple GPU computing, inevitably influenced also the LP detection and recognition approaches. Some are using multiple networks, to firstly identify vehicles in images, and later to detect LPs in these ones' bounding boxes.

A *state-of-the-art* method is presented in (Massoud et al., 2017), with Sighthound's ALPR commercial product (*https://www.sighthound. com*). Three separate deep CNNs, interlaced with accurate, efficient algorithms, are dealing, one by one, with: vehicle identification & cropping, LP detection, and LP recognition respectively. The CNNs are trained and fine-tuned so that they are robust under different conditions (i.e. variations in pose, lighting, occlusion, etc.) and can work across a variety of LP templates (i.e. sizes, background, font, shape, layout etc). The system outperforms leading LP detection and recognition technology (i.e. ALPR) on several benchmarks.

New models of CNNs, SSD – *single shot* (Liu et al., 2015), YOLO – *you only look once* (Redmon et al., 2016) and its further versions 2 and 3, or Faster R-CNN (Ren et al., 2017) object detectors became lately *state-of-the-art*, being more and more employed in LPD, either as stand-alone, or combined with other image processing methods.

In (Silva & Jung, 2018) an ALPR system able to deal with LPs considerably distorted due to oblique views is proposed, the main contribution consisting in a novel CNN able to detect and rectify multiple such LPs in a single image. A *Warped Planar Object Detection Network* (WPOD-NET), using insights from YOLO, SSD and STN – *spatial transformer networks* (Jaderberg et al., 2015), is searching for LPs, regressing for each detection coefficients of an affine transformation to rectify the LP's area to a rectangle as in its frontal view. The positive rectified detections are then fed to an OCR network. The CNN architecture has 21 layers, 14 within residual blocks. The detection block has 2 parallel convolutional layers: one for inferring the probability, *softmax* activated, and one for regressing the affine parameters, with no activation. The method performs similarly with, and even outperforms, some *state-of-the-art* commercial and academic systems, without any adaptation/fine-tune for specific scenarios. An additional contribution is a massive use of synthetically warped versions of real images for augmenting the training data set.

A 3 stages *end-to-end* ALPR system is presented in (Laroca et al., 2019). It is composed by a sequence of 3 YOLO based CNNs, for: vehicle detection (adapted YOLOv2); LP detection and layout classification as: American, Brazilian, Chinese, European or Taiwanese (modified Fast-YOLOv2); LP recognition, all characters at once, using heuristic rules specific to the predicted layout class (CR-NET, based on YOLO). Various constraints, commonly found in ALPR systems, are avoided by training a single network for each task with images from several data sets, taken in various real-world conditions. Some data augmentation tricks and modification of chosen networks are performed to achieve best speed/accuracy trade-off at each stage. Thus, the approach competes, and even outperforms, some previous works and two commercial systems in several benchmark data sets, and provides real time processing of up to 4 vehicles in the scene. Another contribution is the manual labeling of the position of vehicles, LPs and characters, as well as their classes, in all data sets used that have none, or only part of such labels, the annotations being made public.

3. The Method and Algorithm for Segmentation of License Plates

We have intended to design and implement a method as general and flexible as possible, able to perform well under various real conditions. Nevertheless, it should be noted that we still presumed some reasonable, relaxed, minimal requirements regarding the target image and its quality (referring to resolution/sizes, framing, exposure, lighting, reflections, dirt, shadows, etc.). Thus, sufficient resolution is obviously required, for the area of the vehicle registration plate in the digital image to be large enough (to contain sufficient pixels) for allowing the segmentation of the component alphanumeric characters, which in their turn must be large and clear enough for OCR. Experimentally, we found that reasonable resolutions/dimensions for the digital image could be in a range of 320×240 - 1600×1200 pixels, or equivalent at different aspect ratios. It is advisable that the license plates have in such images an area about 120×25 - 480×100 pixels for one-row plates, and the height of the characters of the LPs should be of at least 15-20 pixels. It is also recommended to ensure such a framing that as few disturbing elements as possible would appear around the vehicle and LP. The plate shouldn't be framed too eccentrically, tangent to, or cut by one of the edges of the image. The camera angle is desirable to be as close to normal as possible, on horizontal and vertical, but rotations of up to 30° or 40° are accepted in both planes, respectively. The algorithms we have developed are tuned to provide optimal results on a quite wide range of "reasonable" images, meeting these minimal requirements. Another thing we had in mind was to keep a trade-off between the depth of analysis and required computational complexity, and the necessity of getting fast, even real time, yet very good results, on less powerful, cheap, hardware.

The segmentation method we have designed and implemented uses as input a raw grayscale raster image (GI), with 256 gray levels, one byte per pixel, already loaded in memory. In case of color images, these ones are first converted to grayscale, by using the NTSC formula for the brightness level of each pixel obtained from its corresponding RGB values (Vrejoiu, 2017):

$$GI = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \qquad (1)$$

If too large or too small, the raw grayscale image is automatically rescaled, with integer factors, thus as to have the small dimension ≥ 320 pixels, the large dimension ≥ 550 and ≤ 1600 pixels, and the aspect ratio to be ≤ 2.66.

Although multiple plates can be found with our method, we focus in the following on single vehicle license plate detection in such image.

Step 1. – Creating downscaled contrast map

A transformation is first applied on the input grayscale image (GI) of dimension $h_i \times w_i$, based on some local contrast analysis on rectangular blocks in this image, resulting a “contrast map” (CM) of lower dimensions $h_c \times w_c$, where:

$$\begin{aligned} h_c &= (h_i - h_b) / s_r \\ w_c &= (w_i - w_b) / s_c \end{aligned} \tag{2}$$

with $h_b < w_b$ representing the dimensions of the considered image blocks, and s_r and s_c being the strides, on rows and columns respectively, for sequential localization of these blocks in the image. These four latter values are empirically established and adjusted on experimental basis. Each element of the contrast map CM contains a per-block averaged measure of the significant local contrasts in the corresponding image block on which it is mapped. Local contrasts are calculated by scanning each row in a block, as absolute values, only significant ones, greater than a threshold *i_thr*, being used to compute the average per block. The *i_thr* value is the same for every block, and is, empirically set, either to a predefined value, or to a fraction from the maximum contrast value from the whole image.

Step 2. – Contrast map binarization

The contrast map CM generated at previous step is then binarized using a contrast threshold *c_thr* computed based on the normalized histogram CH, associated to the values in CM:

$$\mathrm{CH}(k) = \mathrm{N}(k) / (h_c \cdot w_c) \tag{3}$$

with $k = 1 \ldots 256$, and N(k) being the number of elements in CM with contrast value k. Each element i of CM will get a new value of 1 after binarization if CM(i) > *c_thr*, or 0 otherwise.

Step 3. – Filtering of binarized contrast map

Vertical runs (columns of connected values of 1 with no horizontal 1 valued neighbors), isolated and/or tangent to the lateral edges, are filtered/erased from the binarized contrast map, if any.

Step 4. – Segmentation of contrast map ROIs

Finally, horizontal runs (consecutive values of 1 in a row in the binarized CM) are assembled in blobs, based on run length coding (RLC) and relaxed connectivity analysis on the runs. Using the bounding boxes of these blobs, remapped on the GI image, a list of regions of interest (ROIs), candidates to contain the registration plate(s), is generated. Actually, coordinates of the bounding box for each such region are thus retained, while the CM, no longer needed, is discarded.

Step 5. – Refining and filtering image ROIs

Some fine adjustments of the coordinates of the retained candidate ROIs are tried, and their list is filtered based on some high gradient density criteria. Finally, one, several or no candidate might remain in the list. It worth noting that these regions might be “narrower” or “wider”, from case to case, than the LP’s area, which may be even rotated/slanted in the image. It has been experimentally verified that in most cases only one ROI is obtained, which really corresponds to the LP. However, in the other situations, the number of candidates is small (2, rarely up to a maximum of 5), and it will be decided in further steps which one will be kept as the correct one. If no ROI detected, a jump to *Step 11* is performed, and the algorithm is resumed with slightly changed parameters and/or processing chain, succeeding usually to get to a convenient result in the end.

Step 6. – Image ROIs binarization

An adaptive algorithm is applied for binarizing the GI contents for each remaining ROI.

A binarization threshold per region (*b_thr*) is first established. For this, two histograms (HL and HH), of those low and respectively high gray levels giving contrast values greater than the threshold *i_thr* as in *Step 1* at the block level are computed for the main area of the respective region of GI (excluding some border strips inside the ROI to avoid possible influences from outside the LP). The arithmetic mean between the statistical means (*m_low* and *m_high*) for each of the two histograms is considered as the global binarization threshold per region:

$$\begin{aligned} b\_thr &= (m\_low + m\_high) / 2 \\ m\_low &= \textstyle\sum k \cdot \mathrm{HL}(k) / \sum \mathrm{HL}(k) \qquad (4) \\ m\_high &= \textstyle\sum k \cdot \mathrm{HH}(k) / \sum \mathrm{HH}(k) \end{aligned}$$

with $k = 1 \ldots 256$.

The kind of LP background is determined based on this threshold, by counting the pixels from the main area of the respective image ROI with values below and above *b_thr*. The background type (dark or light) is given by that kind of pixels with greater counted number. Although this assumption is usually a valid one, there are still some exceptions, mainly due to the type and quality of LP and/or to its rotated pose in the image. Therefore, two other criteria are also considered, the final decision being taken by majority vote among the three. The second criterion is given by the largest sum of the lengths of long vertical runs of "dark" and respectively "light" pixels in the main area, and the third one by the long horizontal lines, "dark" or "light", that are possibly bordering up and/or down the considered main area.

Sometimes, a global threshold might be not so appropriate, mainly in situations of shading or gradient of illumination on the plate. Therefore, a vector of local variable thresholds for each row in the candidate image region (*v_thr*) is also calculated analogously to the global threshold. Depending on the type of background and the difference between the local value *v_thr*[*i*], thus calculated on a row, and the global threshold *b_thr*, the respective "*i*" component value may be either kept (possibly slightly adjusted empirically), or, in special situations, it may be replaced with the global value *b_thr*.

This way, the binarization of the respective candidate region is performed actually by using for each row of pixels the corresponding component of the variable local threshold vector *v_thr*, and taking into account the established kind of background, for getting always the bitmap (BM) with black foreground on white background for each candidate ROI. In the case that a region is too small, the creation of the associated bitmap is done after smoothly up-scaling it (by doubling its dimensions) first.

Step 7. – Filtering bitmaps #1

Most of the times, BMs contain, besides the useful black on white alphanumeric characters of the LP, also various other undesired black blobs on the white background: portions of the LP frame, parts of the car surface around the LP of various shapes and dimensions, fixing screws, various stickers/logos glued/painted on the car surface around the LP, etc. In order to get rid of such parasitic "noises", a first sequence of *ad-hoc* filters is applied both on individual runs, and on blobs formed by connected runs. The different types of filtering and their order resulted from successive experiments and refinements. For generality, wherever possible, dimensional comparisons are using invariant scalar proportions and not absolute values. It has been found experimentally that bitmaps thus "cleaned" may still contain, in some cases, remnants of "noises" still unfiltered. These will be eliminated in subsequent processing steps.

Step 8. – Compensation of rotation, slant or tilt

For each candidate region, possible geometric distortions (rotation, slant/tilt) of the black blobs on white background, kept after the first set of filtering, may exist.

To determine a possible rotation angle of the LP from the horizontal (θ_x), a method was devised based on the estimation of the slope of a virtual straight line that interpolates the geometric centers of the bounding boxes of the *N* blobs remained in the bitmap after filtering #1. All the $N \cdot (N - 1) / 2$ slopes determined by pairs from the *N* centers are calculated, outliers being first filtered. Finally, the slope – and implicitly the angle θ_x, as an *arctangent* function of it – is set as the arithmetic mean of all those calculated. Angles with absolute values between 0.4°-30° are accepted; otherwise it is considered that there is no rotation ($\theta_x = 0°$).

For finding a possible tilt angle from the vertical (θ_y), a method has been devised based on the estimation of the slope of a virtual straight line that interpolates the means of the pairs of longest horizontal segments comparable in length, determined by the outer contour points in the first and respectively the last 1/3 of the height of each blob remained in the bitmap (potentially representing a character), with the constraint that, for each such segment, these contour points are not simultaneously tangent to the bounding box of that blob. Obviously, not for all characters such a slope can be calculated because, due to their shape, the above segments properties may not be met, therefore, as many slopes as possible are calculated. Finally, the slope relative to the vertical axis (and the angle θ_y given by the *arctangent* function of it) is set as their arithmetic mean. If no slope can be calculated, or if at least two blobs are found having each the mentioned pair of segments of equal length with the width of their bounding boxes, then no inclination to vertical axis is considered. Absolute values of angles between 1.5°-30° are accepted; otherwise it is considered that there is no tilt ($\theta_y = 0°$).

It should be noted that, the presence of noises or possible distortions of the (shape of) characters of the LP could affect the correct estimate of the horizontal and/or vertical angles (θ_x and θ_y).

If at least one of these angles is non-zero, then the geometric deformations due to rotation and/or tilt is compensated. An affine transformation of the content of the bitmap is applied, pixel-by-pixel, using as center of rotation the geometric center of the bitmap, to finally align this content with the normal horizontal and vertical axes, based on the θ_x and θ_y angles. All possibly appeared isolated pixels, either black or white, are then filtered in the deskewed bitmap.

Compensating a slope of the LP from the horizontal axis of the image is mainly useful for applying further noise filters in the bitmap based on presumptions of alignment and regularity of characters in the license plate (see *Step 9*). Compensating a slant/tilt of the characters of the plate relative to the vertical axis of the image has beneficial effects for their further correct segmentation and OCR, since the variety of possible shapes for a certain character is thus considerably reduced.

Step 9. – Filtering bitmaps #2

To get rid of undesired blobs still existing after rotation, a second set of *ad-hoc* filters is applied twice on the bitmaps associated with the candidate ROIs, at the level of individual black runs, and of blobs formed by such connected runs. Here also, the types of filtering and their order are resulted from experiments, and for dimensional comparisons some invariant scalar proportions are used. It has been experimentally verified that, when it actually contains the LP, the bitmap obtained after the entire chain of analysis and processing up to this point is (almost) always "clean", noise-free, and correctly oriented, with any rotation, slant and/or tilt compensated, being prepared for further OCR. The extreme alternative is that all blobs from a bitmap – possibly excepting one – have been filtered, in which case the associated ROI is dropped out from the list of candidates.

Step 10. – Final refining and filtering of ROIs

The regions remained in the list of candidates are then "refined". Thus, in the case of ROIs corresponding to registration plates with two rows, not segmented yet as two separate entities, a method for rows separation is applied to the associated bitmap, based on the histogram of vertical projections of its lines. For each row thus separated, a new position is assigned in the list of candidate ROIs, to keep its coordinates in the image and its associated bitmap. The initial common ROI and bitmap are discarded. Finally, ROIs that are unlikely to contain LPs are filtered from the list of candidates, based on size, aspect ratio, number of blobs in their corresponding bitmaps, and some topological relationships between the respective ROIs.

Step 11. – Another iteration needed, or stop?

Finally, a decision is taken from case to case to either resume the algorithm in a new iteration, or to stop the processing at this point. Thus, if only one candidate ROI, or two, located one above the other (in the case of license plates on two-rows), was/were retained, or if the end of the last iteration (the 6th) was reached, the algorithm stops here. Otherwise, if no ROI was retained in the candidate list or if there are too many kept, and if a new iteration is available, the algorithm is resumed starting with *Step 1*.

This way, the algorithm described above can be run iteratively, as necessary, up to six times in different variants, with the minimum contrast threshold of *Step 1* chosen in each of the two mentioned ways, and *Steps 3* and *5* possibly skipped. Also, in some variants, prior to *Step 1*, some pre-processing operations (as smoothing with a mean filter, and/or normalization – stretching of the histogram of initial gray levels) are applied on the grayscale image (GI).

The variants for all iterations are summarized in Table 1, while the functional diagram of the algorithm is shown in Figure 1.

Table 1. Processing variants per iteration

Apply \ Iteration	1	2	3	4	5	6
Grayscale image pre-processing #1 (smoothing)	–	●	–	●	–	●
Grayscale image pre-processing #2 (normalization)	–	–	–	●	–	–
Relaxed threshold for contrast analysis	–	–	●	●	–	●
Fully relaxed zero threshold for contrast analysis	–	–	–	–	●	–
Relaxed dimensional ROIs filtering	–	●	–	●	–	–

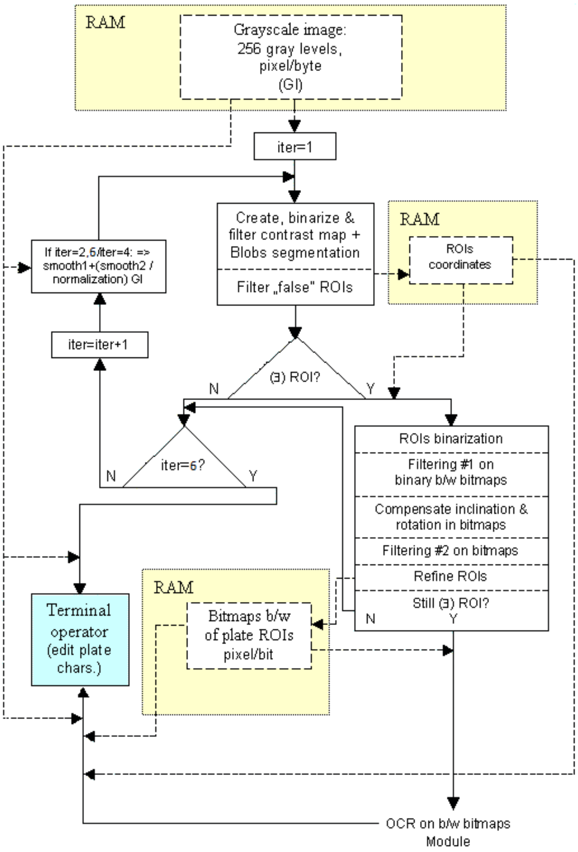

RAM
Grayscale image: 256 gray levels, pixel/byte (GI)
iter=1
Create, binarize & filter contrast map + Blobs segmentation
Filter „false" ROIs
RAM
ROIs coordinates
If iter=2,6/iter=4: => smooth1+(smooth2 / normalization) GI
iter=iter+1
(∃) ROI?
N
Y
iter=6?
N
Y
ROIs binarization
Filtering #1 on binary b/w bitmaps
Compensate inclination & rotation in bitmaps
Filtering #2 on bitmaps
Refine ROIs
Still (∃) ROI?
N
Y
Terminal operator (edit plate chars.)
RAM
Bitmaps b/w of plate ROIs pixel/bit
OCR on b/w bitmaps Module

Figure 1. The functional diagram of the proposed method/algorithm

If still no ROI found after all iterations, the algorithm could be re-run, with slightly different sizes of blocks and stride for CM generation. As an extra-improvement, everything could be run also in a parallel thread on a down-sampled grayscale image too, and results combined. LPs missed by one thread might be thus detected. The maximum number of iterations and runs is controlled by an editable parameter, giving the "analysis depth" as an integer value from 1 to 5 (greater values also involving longer duration).

4. Experimental Results and Discussion

All the software algorithms and code developed, excepting for some image file formats loaders, are original and entirely written by us, being employed in our LPR deployments over time (test demo at *https://veda-mhv.github.io/*). The C language, Win32 API, and the free command line C/C++ compiler from Inprise/Borland® (v.5.5) were utilized for development.

Extensive experiments and testing have been carried out using over 2,330 color and grayscale images, both from our own data set, captured by us and received from others, and from public data sets from the Internet (OpenALPR, 2016 including also JavaANPR, 2006; MediaLab, 2015; HTSol, 2005; Baza_Slika, 2003). The images are of better or worse quality, were taken in various real conditions, daylight (sunny, cloudy, dusky) and night (with artificial visible or IR lighting), with various cameras, at various resolutions/dimensions in the range of 320×240 – 2304×1728 pixels. They contain both front and rear, normal and inclined/rotated views of vehicles registered in over 30 countries as: Romania, Italy, France, Austria, Spain, Belgium, Netherlands, Germany, GB, Slovakia, Czech Republic, Croatia, Bosnia & Herzegovina, Slovenia, Norway, Poland, Ukraine, Swiss, Greece, Cyprus, Turkey, Israel, USA, Mexico, Brazil, Colombia, Singapore, Taiwan, Hong Kong, Australia, South Africa. Each LP contains Latin uppercase letters and/or Arabic digits, all of the same height, on one/two row(s), painted with different colors (e.g. black, blue, red – or white, silver-gray, yellow, orange), on different backgrounds (e.g. white, yellow – or black, blue), as "dark on light" or "light on dark" types.

A very good and consistent behavior of the method and algorithm deployed for LP detection was proved in the vast majority of cases, with detection rates of 97-99%, and even 100%. An overall 96.39% average was attained directly, without specific customization for any case/

situation. Detailed performance, on each data set, is shown in Table 2, from which, it worth noting also the wide variety of the images used for experiments.

Table 3 presents a comparison in terms of recall numbers with two commercial (Sighthound and OpenALPR - *http://www.openalpr.com*) and an academic (Laroca et al., 2019) *state-of-the-art*

Table 2. License plate detection results on various data sets, without any specific customization from case to case, and without removing/discarding possibly less appropriate images from these ones

LPR image data set. ***Images type and capturing conditions***	**Images resolution/size**	**No. of images**	**Missed, partial or false LPs**	**LP detection rate (%)**	**LP countries**
Own image data set Captured by us (2006): *color, daylight*	640×480	72	0	100%	Mainly RO + I, D
	768×576	98	0	100%	
	800×600	17	0	100%	
	1024×768	33	0	100%	
	1600×1200	2	0	100%	
Provided by others (1993-94): *grayscale, daylight/garage*	768×512	38	0	100%	Mainly I + F
Provided by others (2009-10): *grayscale/color, daylight/garage*	704×480	158	4	97.47%	COL
Total:		***418***	***4***	***99.04%***	
HTSol data set, (2005): *grayscale, daylight/night/IR*	768×288 (half-interlaced)	422	9	97.87%	NL, D, F, E, I, IL, GB, US, MX, BR, HK, SGP
	768×576	30	0	100%	AUS
	760×270	45	0	100%	ZA
	768×484	32	2	93.75%	TW
	640×312	30	0	100%	TR
Total:		***559***	***11***	***98.03%***	
OpenALPR benchmarks, (2016): *color, daylight/night/garage, various sizes and aspect ratios*	aprox. [640×400 - 1280×1280]	124	7	94.35%	Mainly BR + EU (D, GB, N, PL)
incl. **JavaANPR snapshots**, (2006): *color, daylight, various sizes and aspect ratios*	aprox. [320×240 - 640×220]	97	0	100%	Mainly SK + CZ
Total:		***221***	***7***	***96.83%***	
MediaLab database, (2015): *color, daylight/night*	640×480	289	11	96.19%	Mainly GR + D + CY
	800×600	119	8	93.28%	
	1792×1312	165	1	99.39%	
	2304×1728	2	0	100%	
grayscale, daylight, various sizes and aspect ratios	aprox. [300×300 - 400×400]	48	0	100%	GR
Total:		***623***	***20***	***96.79%***	
Baza_Slika, (2003): *color, daylight/night*	640×480	475	42	91.16%	Mainly HR + A, D, I, B, UA, CZ, BIH, SLO, PL, F, CH
	1024×768	27	0	100%	
	1600×1200	7	0	100%	
Total:		***509***	***42***	***91.75%***	
Overall performance:		**2330**	**84**	**96.39%**	

Table 3. Comparison with *state-of-the-art* LP detection recall rates [R = TP / (TP + FN)]

Data sets (no. of images)	**OpenALPR** *(open source)*	**OpenALPR** *(Cloud API)*	**Sighthound** *(Cloud API)*	**Laroca** (YOLO)	**Our method**
OpenALPR-EU (11+97) & **Baza_Slika** (509)	91.09%*	90.36%*	99.64%*	99.68%	≥ 93.19% (97.79%$^{***)}$
OpenALPR-US (222) & **Caltech cars_Markus** (126)	86.89%**	89.33%**	99.09%**	99.13%****	≥ 61.49% (65.24%**)

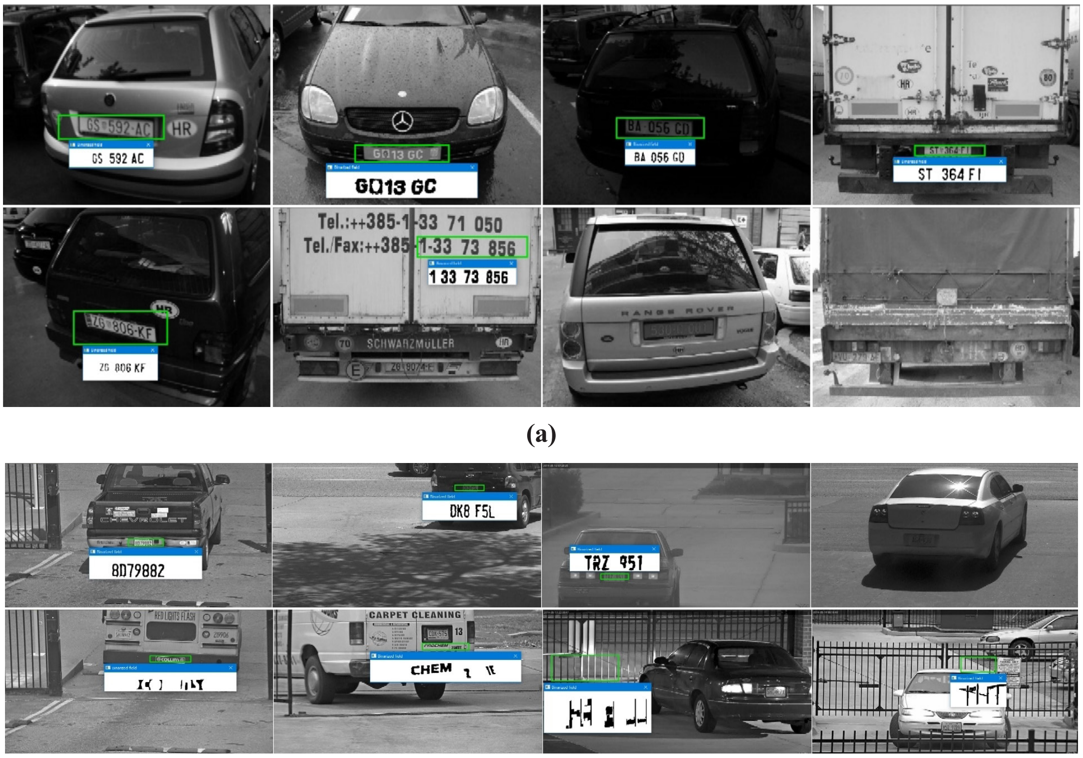
*, ** *- obtained by removing 58* (9.40%), respectively 20** (5,75%) "unreasonably hard samples" (Masood et al., 2017);*

*** *- obtained removing only 29*** (half from 58*!) less appropriate samples in our case;*

**** *- only on Caltech cars_Markus data set (126 instead of 348 images) ! (Laroca et al., 2019);*

Remark: OpenALPR requires user input for the type/region of the LP.

systems, on two data sets with vehicle images from Europe and USA (Masood et al., 2017), one more appropriate for our approach, the other one less. A brief discussion is needed here. The comparison of the results is not 100% accurate and reproducible due to the fact that (Masood et al., 2017) didn't exactly use the entire data sets, since removing 58 (9.40%), and respectively 20 (5.75%) "unreasonably hard samples" from them, and we didn't have access to their "supplementary material" indicating which were those ones. However, the recall rates are showing average good behavior of our method in the first case, and the mismatch of the kind of several images from the second data set with it. Examples of such less appropriate images, not complying with the minimum "reasonable" requirements specified for our approach, are shown in Figure 2 (b). The major issue is the wide framing of vehicle in the scene, with too small and, sometimes, low-resolution LP area, but much from the surrounding scenery. However, also in quite many such situations, our method still detects the LPs, even without cropping first the vehicles in the image. For the other cases, we manually cropped the vehicles



(a)

(b)

Figure 2. Some examples of difficult and/or less appropriate images:
(a) 640×480, from Baza_Slika data set, with vehicle/LP in poor or uneven lighting, shadowed, dirty, and/or lacking contrast, or surrounded by other texts
(b) 1280×720, from OpenALPR-US data set, with vehicle/LP framed too widely, in poor or uneven lighting, shadowed, surrounded by other texts or background with grids and alternating shadows

from such images with enough resolution, and verified that the wide framing is indeed the problem.

In most of the cases, after the first iteration of the algorithm detailed in Section 3, without the need for any filtering, the ROI containing the LP is directly obtained. On the other hand, there are situations when the initial analysis gives several candidate ROIs (usually few ones, among 2, and rarely up to 4-5 at most) in one of the necessary iterations, as in Figure 3.

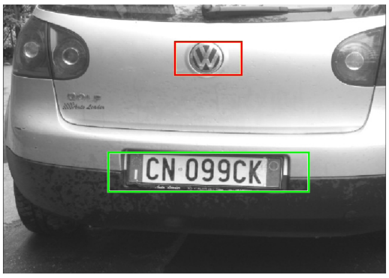


Figure 3. Initial segmentation without filtering

From these potential ROIs, usually only the correct one remains after applying the bitmap and contextual filtering described in *Steps 5, 7, 9* and *10*.

Figure 4 illustrates the outcomes at four stages: (a) - the initial bitmap obtained after the binarization of the image ROIs containing the LP; then, (b) - after filtering the long horizontal lines in the bitmap, and (c) - after the primary filtering set; (d) - after compensating possible rotations, slant and/or tilt in the bitmap, and the secondary filtering set applied on this one. In some cases, the bitmap is clean enough, and aligned from the beginning.

Still, usually, there are various noises and artifacts that must be erased, possible geometric distortions (rotation, slant and/or tilt) that must be compensated, and the remained noises could be then filtered based on the alignment and regularities of the characters composing the LP. There are also cases of two-rows plates initially segmented as a single ROI and being eventually separated as in *Step 10* of the algorithm, or cases in which the size of the ROI containing the LP is too small, and associated bitmap is created after smoothly up-scaling that ROI by doubling its dimensions.

The presented examples are illustrating the performance of our algorithms with respect to:

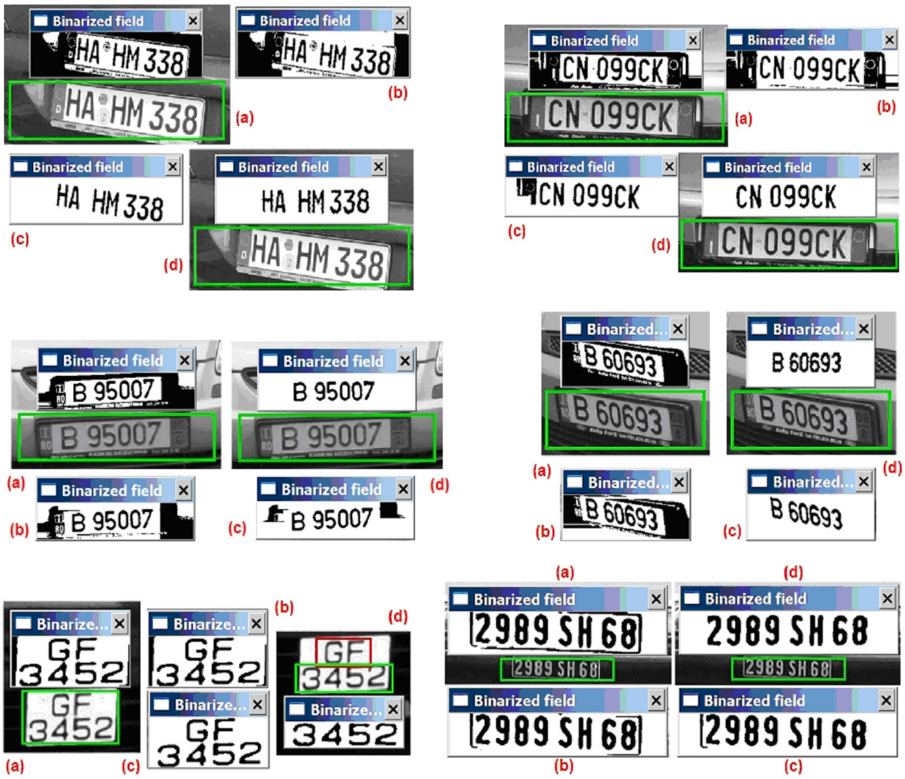


Figure 4. Segmentation stages outcomes after: (a) ROI binarization; (b) filtering long horizontal lines; (c) filtering#1; (d) eliminating possible rotation, slant and/or tilt, and filtering#2

- precise detection/localization of registration plates by bounding rectangles, regardless of LP's type, shape or orientation;

- correct binarization of ROIs thus framed;

- filtering the long horizontal "lines" given by regions of the vehicle around the plate, its bordering frame, or even shadows over it, which also ensures the "decoupling" of the characters of the license plate, possibly connected by such linking elements;

- rest of primary filtering set, which ensures the massive elimination of other parasitic blobs and noise from the (binary) bitmaps;

- compensation of geometric distortions in bitmaps due to rotation, slant and/or tilt;

- refining the list of ROIs candidate to contain a LP, by splitting in component rows where appropriate and discarding false candidates;

- secondary filtering of remained noises from the bitmaps associated with candidate ROIs, while preserving only the "useful" alphanumeric characters in them.

Bitmaps finally obtained are "clean", containing only the alphanumeric characters of the LPs, correctly aligned, with no rotation, slant or tilt (anymore), which is a favorable premise for successful segmentation and recognition of LPs' characters, as in OCR of b/w printed pages.

The processing time depends on the input image (its resolution/size, framing and content), depth of analysis, and CPU on which they are run. In our case the duration per image, also including ROIs binarization and filtering, usually lays in the range of some ten(s) to few hundreds of milliseconds on a P4/1.5GHz CPU, and much lesser on new generation multi-core CPUs. For video sequences of traffic flow, "real time" may be expressed as something about 15-30 fps., that is 67-33 milliseconds per frame. Since our method is providing comparable times for the "reasonable" usual cases, it could be also used in certain real time applications (we even tested it on moving vehicles, with good results, in video flow at 15 fps.), although it was mainly intended for static or slow motion situations, e.g. for access control at gates, barriers or toll points. Even if being essentially an iterative sequential algorithm running on cheap, modest hardware resources, still possible further optimization of its computation time could be investigated by making use of the new parallel and multi-core processing technologies available today. We are already using two parallel computing threads when also analyzing a down-sampled gray levels image. Nevertheless, more parallelization on multi(-core) systems could be also tried for some individual steps of the algorithm like: Step 1 – to compute the contrast measure per each block when generating the reduced size contrast map; Step 6 – for thresholding each proposed ROI; Steps 7, 8, 9 – to filter and deskew every bitmap associated with the ROIs at each of these steps. A processing speed optimization might be obtained this way, mainly for some images widerly framed, with high resolution/dimensions and difficult background, when deeper analysis is necessary. On the other hand, for usual reasonable images targeted by our method, it might not get relevant improvements, since the performance is already acceptable and no space up-scale of the problem is foreseen. It should be investigated whether the additional control and synchronizing code for parallel processing won't reduce/cancel possible gain in computation time.

5. Conclusion and Future Work

A fast and effective method for LP detection is proposed. It is based on estimating a measure of local contrast (high gradient) on image blocks, binarization and connectivity analysis on binary runs in the reduced size contrast map, without requiring some special computing and memory resources. The processings involved are simple and faster than in most of the other comparable approaches, mostly simple arithmetic operations with integers being implied. Some adjustments and filterings at different stages, as well as the values of some parameters used, resulted empirically from experiments.

Original contributions of our approach are:

- fast and effective generation of a reduced size contrast map (CM) associated with the image, using a per-block contrast measure;

- binarization, filtering, and segmentation of the binarized CM based on CCA on its runs;

- smart adaptive binarization of thus obtained ROIs (re)mapped on the initial image;

- smart and fast correction of rotations/slants/tilts in the (binary) bitmaps of ROIs;

- *ad-hoc* "noise" filtering in these bitmaps (to keep only those b/w blobs of alphanumerical characters in the LP, for direct further OCR on such bitmaps to complete a LPR task);

- flexible, general and effective rules for false candidates' removal from the ROIs list.

Experiments and tests performed highlighted the performances of the implemented method and algorithms. Good behavior has been observed in the vast majority of situations, even in some not among the most desired lighting and clarity cases. Usually, bitmaps finally obtained are "clean" and contain only - black on white - the alphanumeric characters composing the LPs, horizontally aligned, without any slant/tilt, which is a favorable premise for their successful segmentation, as well as for their recognition (OCR) from each such bitmap. Still, in cases of too wide framing, lack of resolution or contrast, or in extreme lighting situations (due to blinding reflections, dramatic alternations of full light with shadow, darkness), or too slanted, dirty, damaged LPs, etc., when even human eye could hardly identify correctly a LP, sometimes inconsistent results might be achieved.

Some related improvements, as well as possible optimization and parallelization of certain steps of the algorithm, might be further investigated.

Acknowledgements

Writing this paper triggered special retrospective memories of over 15 years of enthusiastic days and nights of hard work, jointly assumed, with tries and fails, frustrations and accomplishments, shared in scientific and technical creation in the Computer Vision field, but not only, with Dr. Eng. Dan Mugurel Onțanu along 1990-2005. We also thank two unknown reviewers for their helpful suggestions for improving the paper.

REFERENCES

Anagnostopoulos, C. N., Anagnostopoulos, I., Loumos, V. & Kayafas, E. (2006). A License Plate-Recognition Algorithm for Intelligent Transportation System Applications, *IEEE Transactions on Intelligent Transportation Systems, 7*, 377-392.

Baza_Slika. (2003). Student's project collected LPR image data set, coord. V. Srebric, Faculty of Electrical Engineering and Computing - Zagreb, Croatia. Available at: <http://www.zemris.fer.hr/projects/LicensePlates/english/baza_slika.zip>

Belean, B., Streza, M., Crisan, S. & Emerich, S. (2017). Dorsal Hand Vein Pattern Analysis and Neural Networks for Biometric Authentication, *Studies in Informatics and Control, 26*(3), 305-314. DOI: 10.24846/v26i3y201706

Caltech cars_Markus. (2001). Car rear images, taken in Caltech's parking lots, USA, data set by Markus Weber. Available at: <http://www.vision.caltech.edu/Image_Datasets/cars_markus/cars_markus.tar>.

Draghici, S. (1997). A Neural Network Based Artificial Vision System for Licence Plate Recognition, *International Journal of Neural Systems, 8*(1), 113-26.

Gonzalez, R. & Woods, R. (2007). *Digital Image Processing*, 3rd edition. Pearson/Prentice Hall.

HTSol data set. (2005). LPR images from Hi-Tech Solutions Ltd., Israel. Available at: <http://x.htsol.com/>.

Jaderberg, M., Simonyan, K., Zisserman, A. & Kavukcuoglu, K. (2015). Spatial Transformer Networks, *Advances in Neural Information Processing Systems, 28*, 2017-2025. Curran Associates, Inc.

JavaANPR snapshots. (2006). LPR images set from O. Martinsky, Slovakia. Available at: <http://javaanpr.sourceforge.net>.

Kahraman, F., Kurt, B. & Gokmen, M. (2003). License Plate Character Segmentation Based on the Gabor Transform and Vector Quantization, *ISCIS 2003*, *LNCS 2869*, 381-388. Springer.

Kwasnicka, H. & Wawrzyniak, K. (2002). License Plate Localization and Recognition in Camera Pictures, *AI METH 2002*, 243-246.

Laroca, R., Zanlorensi, L. A., Goncalves, G. R., Todt, E., Schwartz, W. R. & Menotti, D. (2019). An Efficient and Layout-Independent Automatic License Plate Recognition System Based on the YOLO Detector, *arXiv:1909.01754v2*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. & Berg, A. C. (2015). SSD: Single shot multiBox detector*, arXiv:1512.02325*.

Martinsky, O. (2007). Algorithmic and Mathematical Principles of Automatic Number Plate Recognition Systems, *B. Sc. Thesis*, Faculty of Information Technology, Department of Intelligent Systems, Brno University of Technology, Czech Rep.

Masood, S. Z., Shu, G., Dehghan, A. & Ortiz, E. G. (2017). License Plate Detection and Recognition Using Deeply Learned Convolutional Neural Networks, *arXiv:1703.07330v2*.

MediaLab database. (2015). LPR images set from I. E. Anagnostopoulos et al., Multimedia Technology Laboratory, National Technical University - Athens, Greece. Available at: <http://www. medialab.ntua.gr/research/LPRdatabase.html>.

Moustafa, A. A. & Jaradat, M.-I. R. M. (2015). A New Approach for License Plate Detection and Localization: Between Reality and Applicability, *International Business Research, 8*(11), 13-25.

OpenALPR benchmarks. (2016). BR, EU & US LPR images data set, uploaded by M. Hill. Available at: <https://github.com/openalpr/benchmarks/tree/master/endtoend/>.

Parker, J. R. & Federl, P. (1996). An Approach to License Plate Recognition, *Technical Report, Project 1996-591-11* at The Lab. for CV, Dpt. of Computer Science, University of Calgary, Canada. Presented at *Vision Interface '97 Conference*.

Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection, (*CVPR 2016)*, 779-788.

Ren, S., He, K., Girshick, R. & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(6), 1137-1149.

Shapiro, V., Bonchev, S., Velichkov, V. & Gluhchev, G. (2004). Adaptive Multi-National License Plate Extraction, *Cybernetics and Information Technologies, 4*(1), 76-88. Bulgarian Academy of Sciences.

Silva, S. M. & Jung, C. R. (2018). License Plate Detection and Recognition in Unconstrained Scenarios, *ECCV 2018, LNCS 11216*, 593-609. Springer.

Vrejoiu, M. H. (2010a). AL/NPR based systems. Some aspects regarding specific issues and solutions, *Romanian Journal of Information Technology and Automatic Control, 20*(1), 47-54 (in Romanian).

Vrejoiu, M. H. (2010b). AL/NPR Based systems. A synthetic analysis of some ANPR products, *Romanian Journal of Information Technology and Automatic Control, 20*(1), 55-77 (in Romanian).

Vrejoiu, M. H. (2017). A Pattern Matching Method and Algorithm for Face Detection, *Studies in Informatics and Control, 26*(1), 75-86. DOI: 10.24846/v26i1y201709

Xu, G., Wang, S., Yang, T. & Jiang, W. (2018). A Neutrosophic Approach Based on TOPSIS Method to Image Segmentation, *International Journal of Computers Communications & Control, 13*(6), 1047-1061.