Efficiency of the Electronic License Plate Recognition Systems

P. Slapsinskas[1], M. Zilys[1]
[1]*Department of Electronics Engineering, Kaunas University of Technology, Studentu St. 50–144, LT-51368 Kaunas, Lithuania*
mindaugas.zilys@ktu.lt

***Abstract*—The precise recognition of the vehicle determines the efficiency of intelligent transportation systems. Intelligent transportation systems become more effective if it can identify the vehicle. Electronic license plate recognition systems used to solve this problem. Proposed electronic license plate recognition systems model adapted for simple internet protocol video surveillance cameras. An analysis of the internet protocol video surveillance camera's processors, operating principle and software created guidelines for the algorithm. According to performance of the internet protocol video surveillance camera formed four efficiency criteria for license plate recognition algorithms. It's performed the experiment with four algorithms to search for a license number plate by different attributes of the license number. The most efficient algorithm was selected by criteria from experimental results. The algorithm is further analyzed and proposed the application model of the internet protocol video surveillance camera.**

***Index Terms*—Intelligent transportation systems, image analysis, image processing, video surveillance.**

I. INTRODUCTION

With constant growth of transport flows and increase in traffic jams, air pollution and noise level in the city increase, fuel is used without any need. In order to settle this problem the Intelligent Transportation Systems (ITS) are created and installed. The purpose of these systems is to collect information on traffic conditions and transport flows and transfer this information to transportation systems. ITS quality is mostly influenced by data collection and processing subsystems [1]. ITS is considered more efficient, if such subsystems are capable of identifying a vehicle, i.e. ITS can trace vehicle, speed and direction of its movement and direct it to a faster route. Electronic Vehicle License Plate (VLP) recognition systems help to achieve this objective. The Electronic License Plate Recognition System (ELPRS) is a closed system, which is able to detect VLP from the received data.

At this moment the most effective measure of ITS for the data collection is video surveillance and processing systems [1]. In such way ELPRS is implemented. In order to get lower system's resources expenditure and more accurate results in such ELPRS, the up-search of VLP is made in the image [2]. At first a car is searched in the image. After finding vehicle, the VLP location search is performed. In the location image VLP symbols are cut and the plate is read according to the optical recognition of symbols. The following is used for implementation of these tasks: intelligent video cameras and video digital signal processing (DSP).

ELPRS, implemented in the following methods, are sufficiently accurate and effective; however the disadvantages of such systems are high price and large energy consumption. The system uses a powerful processor, with integrated image processing and analysis functions.

This article proposes another way of realization of the ELPRS. A simple IP camera is monitoring traffic flow, capturing frames, where the most visible VLP. Of the frames found VLP and sent VLP picture of the network.

Such a realization will perform: analysis of system of IP video surveillance cameras; VLP location recognition analysis; application and research of VLP location recognition in the IP video surveillance camera.

II. ANALYSIS OF SYSTEM OF IP VIDEO SURVEILLANCE CAMERAS

IP video surveillance camera consists of: image sensor, video processing and codec subsystem, ethernet card, additional input and output interfaces. Video processing subsystem consists of video processing unit, read-only memory and random-access memory.

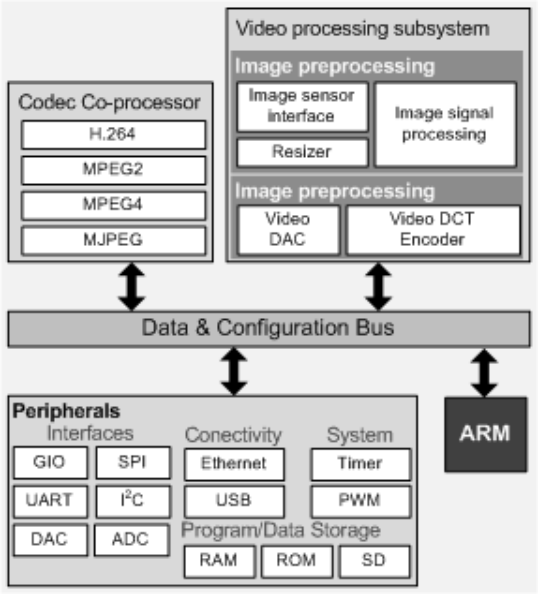

Fig. 1. Block diagram of video SoC processor.

Manuscript received December 6, 2012; accepted May 24, 2013.

Let's consider the video processing systems more. SoC (System on Chip) processors are used for IP video surveillance cameras (Fig. 1). Video processing subsystem performs video pre-processing (image scanning from image sensor, image conversion from RGB to YUV, image size changing, image split to blocks, etc.) and video post-processing (image conversion to analogue signal, image pre-codec, etc.). Integrated co-processors are used for encoding. Peripherals consists of different processor inputs/outputs (USB, Ethernet interface, GIO, etc.), timers needed for systems, data transfer bus. The kernel of this processor is ARM processor, which surveys status of subsystems and changes their parameters. All subsystems communicate with processor via data bus [3]–[6].

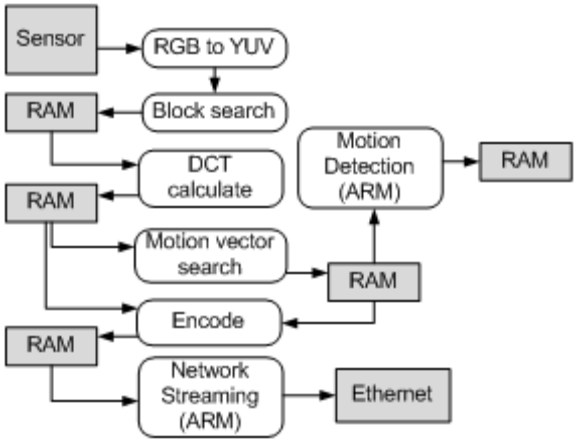

Fig. 2. Action diagram of IP video surveillance camera.

IP video surveillance camera with such processor codes the video, as shown in the diagram (Fig. 2). Depending on type of image sensor, processor scans the image in blocks or portions of rows. Video subsystems immediately convert the scanned video shot from RGB to YUV colour system; small video blocks are formed and stored in RAM. When the necessary number of shots is collected, encoding of the video is started: DCT coefficients and movement vectors of small video blocks are found. This data is encoded. ARM processor puts the encoded information to the network (RTSP, HTTP) packages, from which video data transfer flow is formed. Since the search of movement vectors is necessary during video encoding, movement recognition function is implemented according to movement vectors, which is performed by ARM processor [4].

Due to constantly improving video codec and processing algorithms, IP video surveillance camera software is constantly renewed. Due to this criterion and easier allocation and management of system resources, IP video cameras use Linux operating system, which has all interfaces with video processing processor's subsystems and periphery. A program is created in this system, which, according to system's status, sends instructions to ARM processor on how to control IP video surveillance camera. In order to improve or create new function for such type of IP video camera, we must create the program namely for Linux system [4], [5].

The main difference between DSP for video analysis (e.g. DM6467) [7] and video IP surveillance cameras processors (e.g. DM365) [6] is that one of kernels of DM6467 is video DSP, which is used for video analysis, and DM365 have only a minimum video subsystem, which is intended only for video codec. In other words, DM365 only ensure good quality video coding.

After analysis we formed the model of electronic ELPRS (Fig. 3) according to the analysed diagram of IP video surveillance camera (Fig. 2). As we can see, VLP location recognition will be performed by ARM processor. ELPRS will take video and information on moving objects from RAM. In order to implement such system, the following criteria must be taken into consideration:

1) Processor is not particularly fast and it uses 50-60% of its resources for video processing, therefore the method of VLP location recognition must be found with as few calculations as possible;
2) VLP recognition method can be performed only by ARM processor, because video processing subsystem only generates data for video codec, and coding processors only encode it;
3) VLP recognition method may scan the image and information on moving object from RAM;
4) Use as few resources of RAM for interim ELPRS results as possible.

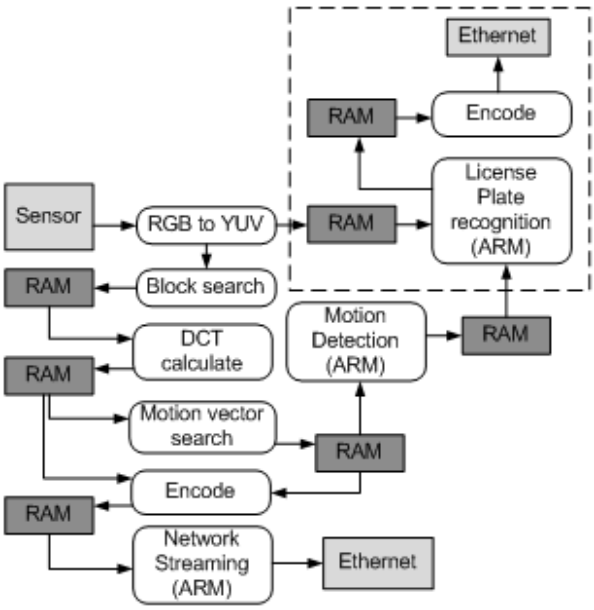

Fig. 3. Proposed method of ELPRS on the video cameras.

III. VEHICLE LICENSE PLATE LOCATION RECOGNITION ANALYSIS

Human eye detects VLP number from the whole entirety according to VLP form and location on the car, its symbols and background colours, which brightness differs a lot. A person may read VLP from symbols only after processing of information. Image is stored in colour pixels in the video processing systems, and the algorithms of typical pixels search are used for recognition of VLP number. The same as human eye, VLP recognition methods search for pixels or entirety of pixels, which define VLP number's colour, outlines and colour of symbols, its form and place in the image.

Looking for the most efficient VLP recognition algorithm for IP video surveillance camera, we made the experiment with 129 photos with the size of 256×256 and vehicle in them with the visible license plates. During the experiment we observed reliability of algorithm i.e. how many numbers

the algorithm has correctly recognized and evaluated it in percentage; we calculated the average duration of algorithm processing. We also calculated how many operations must be made for one pixel in order it could define VLP feature. In order to find out how much RAM the algorithm needs, during the experiment we observed the areas of pixels of data necessary in each step S_{avg}. If in a specific step data is the entire picture, then we multiplied length and width. If only specific pixels are analysed, we add them all up. After that we calculated the average of all pixels of tests.

We carried out the experiment with the following algorithms: according to VLP colour [8]; according to VLP form [9]; according to outlines of VLP symbols [10]; according to corners of VLP symbols [11].

All algorithms were implemented with MATLAB standard functions and examined on computer with processor Intel Dual Core, 2×2.4 GHz. The received results P, t_{avg}, c_{op}, S_{avg} of the experiment we filled in the Table I.

In order to select the most effective algorithm of IP video surveillance camera, we evaluated each algorithm from 1 to 4 according to the analysed criteria.

TABLE I. COMPARISON OF SEARCH ALGORITHMS OF VLP LOCATION.

Algorithm / Criteria	By Color	By License Plate Form	By Character Contours	By Character corners
Accuracy P, %	93,02	97,67	96,12	96,9
Average time of operation t_{avg}, ms	583	403	97	85
Operation count for 1 pixel c_{op},	358	195	174	112
Average operating system memory S_{avg}, KB	288227	212992	87818	70908
Efficiency, E	1	2,5	2,75	3,75

If algorithm $P$is the highest, its evaluation is $E_p = 4$; the second highest according to reliability is evaluated in $E_p = 3$, the third – $E_p = 2$, the last $E_p = 1$. If algorithm t_{avg} is the lowest $E_t = 4$, the remaining higher are evaluated in sequence 3, 2, 1. Also, if c_{op} and S_{avg} are the lowest, the algorithm is respectively evaluated in $E_{op} = 4$ and $E_S = 4$.

After such evaluation of all algorithms, we calculate its suitability for ELPRS system according to the following formula

$$E = \frac{E_p + E_t + E_c + E_s}{4}. \quad (1)$$

The calculated $E$is shown in Table I, last row. As we can see, the highest E is LP recognition according to corners; therefore we will apply it in IP video surveillance systems as ELPRS.

IV. Application of Vehicle License Plate Location Recognition According to Corners in Electronic License Plate Recognition System

VLP location recognition according to corners consists of the following steps: search of outlines; search of corners according to minimum proper value; search of VLP possible places; VLP selection from possible places and its cutting from picture [11].

We will use 5 cycles for these four steps (2 cycles will be needed for the second step) in order to select the most typical pixels. During each cycle we will narrow the data processing interval and index the typical pixels. In such way we will lose RAM resources, but accelerate the process itself. We will reserve static matrixes in RAM, in which we will store indexes of typical pixels and interim results.

From the proposed ELPRS model (Fig. 3) we know that ELPRS system surveys the flow of video and fixes moving objects. If the object crosses the line in the road, the shot is fixed, which is immediately converted to YUV colour system and each component of pixel colour is saved in the memory in the matrixes M_Y, M_U, M_V:

$$M_Y = [y_{ij}]_{n\times m}, \quad (2)$$

$$M_U = [u_{ij}]_{n\times m}, \quad (3)$$

$$M_V = [v_{ij}]_{n\times m}, \quad (4)$$

where n – video height, m – video width. Since video in YUV system is stored in 8 bits, therefore in ELPRS RAM we reserve s_I bits

$$s_I = 24 \cdot m \cdot n. \quad (5)$$

Movement recognition algorithm stores the area of moving object in the matrix M_O

$$M_O = \begin{bmatrix} l_0 & b_0 & e_0 \\ \vdots & \vdots & \vdots \\ l_i & b_i & e_i \\ \vdots & \vdots & \vdots \\ l_h & b_h & e_h \end{bmatrix}, \quad (6)$$

where l_i – moving object row in video; b_i – the first pixel of moving object in the row l_i; e_i – the last pixel of moving object in the row l_i; h – number of rows of moving object. We plant that the object may occupy the entire height of video and $m > 255$, therefore in RAM we will reserve static matrix of 16 bits 3 columns and height m. This matrix will need s_O bits

$$s_O = 16 \cdot 3m = 48m. \quad (7)$$

According to [12]–[17], Sobel edge detection algorithm is four times faster than the Canny edge detection algorithm. The edges detection algorithm is first filter to finding VLP,

so this step will process most of the data. Therefore, we must choose the easier and the faster algorithm. Applying Sobel filter in the first step, we find all sides of in the area of video M_Y, defined by M_O. We apply Sobel filter both, horizontally and vertically, and store the results in matrixes G_x, G_y of the same size as M_Y:

$$G_x = [g_x]_{n\times m}, \tag{8}$$

$$G_y = [g_y]_{n\times m}, \tag{9}$$

where:

$$g_x = f(i,j) = M_Y * \begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}, \tag{10}$$

$$g_y = f(i,j) = M_Y * \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}. \tag{11}$$

As we can see from formulas 10 and 11, g_x, g_y may acquire negative values and negative result may exceed 128, therefore 8 bits are not enough for storage of values. However, knowing that in further acts these numbers will be squared, we straight multiply the negative values by -1 and allocate in RAM for these two matrixes s_G bits

$$s_G = 2\cdot 8\cdot m\cdot n = 16\cdot m\cdot n. \tag{12}$$

We also apply the filter in the first cycle, which would pass only those pixels, which satisfy the following condition

$$\min(g_x, g_y)_{ij} > T_E. \tag{13}$$

If the condition is satisfied, we write in the matrix M_E, which height is equal to n, in the row i column j_e the pixel index j and increase j_e by one. In the new row we equate j_e to 0. Considering that $m > 255$ and outlines are separated by interval of at least one pixel, we allocate in RAM for matrix M_E s_E bits

$$s_E = 16n\frac{m}{2} = 8\cdot m\cdot n. \tag{14}$$

After indexing pixels, which define possible places of corners, the second step may be performed in these pixels – to find out proper values of the pixel matrix M and if its minimum proper value is higher than T_C, then the pixel is the corner. Therefore in the first step of the second step we will find the minimum proper values of the matrix M. Pixel matrix M is defined

$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}, \tag{15}$$

where

$$A = g_x{}^2 * G, \tag{16}$$

here G – Gaussian filter.

We calculate the proper value $\lambda_{\min}$:

$$\lambda_{\min} = \frac{A + B - \sqrt{(A-B)^2 + 4C^2}}{2}, \tag{17}$$

$$M_\lambda = [\lambda_{\min}]_{m\times n}. \tag{18}$$

We store the received result in the matrix M_λ. Since during calculation of proper value of the matrix Gaussian filter is used, which numbers are real, then for storage of the matrix M_λ values 32 bits will be needed, therefore this matrix will need s_λ bits

$$s_\lambda = 32\cdot m\cdot n. \tag{19}$$

In the second cycle of this step we will look for corners, which indexes of columns we will store in the matrix M_C and the array A_C. Using M_E indexes we start the search in the matrix M_λ with the scrolling window, which centre is the analysed pixel and the length and width is equal to odd number R_C. This window scrolls from top to bottom starting from the left. If window value is higher than T_C, it is checked if there are higher values in the window environment than the central pixel. If this condition is satisfied, we consider that the central pixel is the corner. Consequently:

$$M_C(i, j_c) = 1, \tag{20}$$

$$A_C(j_c) = j, \tag{21}$$

$$j_c = j_c + 1. \tag{22}$$

16 bits are needed for storage of array A_C value and only 1 or 0 are stored in the matrix M_C, therefore s_c bits will be needed for storage of corner indexes, which depend on the width of window

$$s_C = \frac{32m}{R_c - 1} + \frac{2mn}{R_c - 1} = \frac{2(16m + mn)}{R_c - 1}. \tag{23}$$

After finding the corners and storing their indexes in the matrix M_C and the array A_C, we can start the search of VLP location. We have stored the possible locations in the matrix M_L, which is made of 100 rows and 5 columns. The first element of the row is number of corners; the second and the third column store the coordinate of the left bottom VLP location corner, and the fourth and the first – the coordinate of the top right corner. This matrix will need $s_L = 5\cdot 100\cdot 16 = 8000$ bits.

Using the indexes of corners, we start the cycle of the third step from the first found bottom corner. We consider this corner the left corner of VLP. Using the scrolling square window, which side is equal to $R_L = 2r + 1$; its movement direction is shown in Fig.4. At first it moves to direction 1

and looks for the corner. If the corner is found, 1 is added to C and the new corner becomes the window centre.

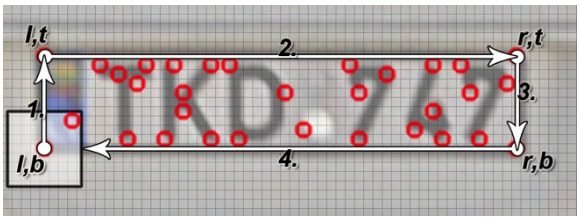

Fig. 4. The Scheme of possible VLP location's search.

After that the next corner is searches. If there are no new corners in the window, the window moves through 5 more pixels to the last movement direction 1. If the corner is still not found, it is fixed that the last found corner is the top left VLP corner and we start to move in the direction 2 and look for new corners. If window cannot find new corners also moving to direction 2, it is fixed that this is not VLP. In such way the bottom right corner of number is searched moving to direction 3. Finally the window returns to the left bottom corner in the direction 4 and all corners C are counted. If $C > C_{\min}$, we fix that this is the location of VLP and save in the matrix the number of M_L corners and the coordinate of the left bottom and the right top corners of VLP. We delete the corners found in case of successful or unsuccessful location search from the matrix M_C, writing instead of them 0.

1) After founding all possible locations of VLP, we select from them the most suitable according to the following criteria:

2) The actual VLP location must have one of the most corners;

3) The actual VLP location must occupy the area, which length is as close as possible to w and the height – as close as possible to h;

4) The actual LP location must be as close as possible to the video bottom.

At first the possible locations are analysed according to the first 2 criteria. If several locations are found that meet the criteria, then they are evaluated according to the criterion 3. The found VLP location is cut from the matrixes M_Y, M_U, M_V. JPG image is formed from the data of these matrixes and is sent by Internet.

TABLE II. MEMORY ALLOCATION.

Matrix size	S_U, Used RAM, KB	S_R, Reserved RAM, KB	S, %
s_I	833,203	833,203	100
s_O	2,343	2,343	100
s_G	296,133	555,469	53,3
s_E	17,117	277,344	6,2
s_λ	68,469	1110,938	6,2
s_C	0,975	36,105	2,7
s_L	0,219	0,97	22,4
s	1218,439	2816,77	43,3

$$s = s_I + s_O + s_G + s_E + s_\lambda + s_C + s_L. \quad (25)$$

The quantity of RAM needed for interim results is calculated according to the (25).

V. EXPERIMENT RESULTS

We were played during an experiment video (711x400 pixels) with traffic on a motorway. Motion detection algorithm cut moving vehicle in the image from the video frame. The ELPRS algorithm is tested on three systems: PCs with Intel Dual Core, 2 x 2.4 GHz on the DM365 board leopard and Freescale i.MX27 Evaluation other. Select these optimal parameters [13]: $T_E = 45$, $T_C = 95$, $R_C = 3$, $R_L = 5$, $C_{\min} = 15$, $w = 100$, $h = 23$. Of the 969 images with the vehicles the ELPRS algorithm correctly cut 941 numbers. The experiment was monitored in the intermediate results: area of the vehicle in the pixels, the vehicle's contour pixels area and the vehicle's corner's pixels area. The results are sorted from the lowest to the highest according to the moving vehicle area and drawing graphs (Fig. 5–Fig. 7).

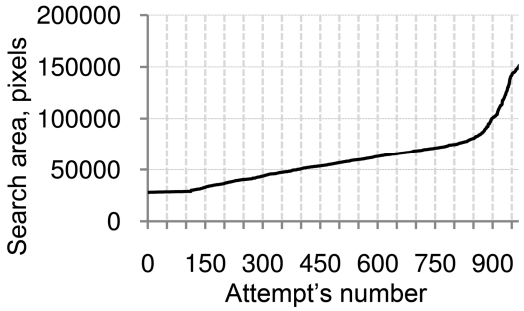

Fig. 5. Filled pixels in the image of the object.

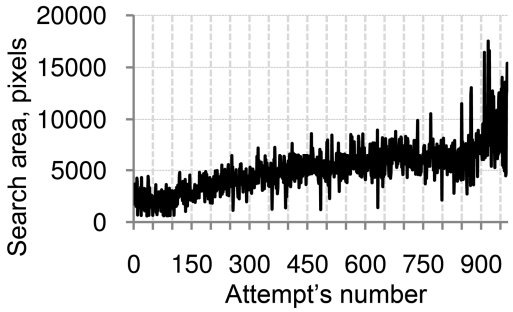

Fig. 6. Filled pixels in the image of the corner.

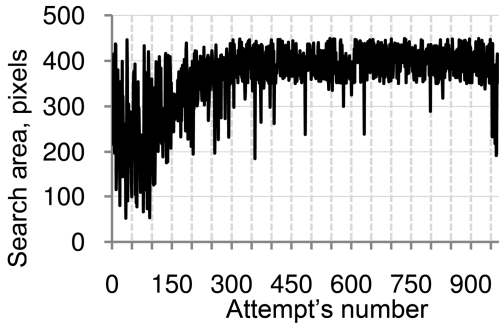

Fig. 7. Filled pixels in the image of the corner.

From these results, we have chosen the maximum and calculated according to the memory used, which is shown in the Table II. In addition, this table includes reserved memory to operating memory, and calculated the ratio between used and reserved memory. The experiment was recorded license plate search time: we begin to calculate a time, when loaded M_Y, M_U, M_V and M_O arrays in the

memory and stop a time, when excluded the license plate. We have arranged the obtained results according to the moving object area, and featured the graph (Fig. 8).

In addition, we have compared the same algorithm, when the license plate search is performed with the proposed indexing and without it (Fig. 9).

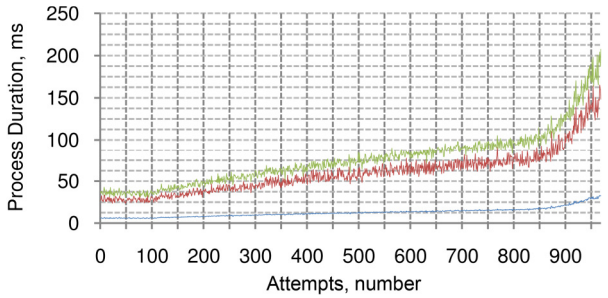

Fig. 8. Process duration of proposed VLP location search on the: first of the top (green) – DM365, second (red) – i.MX27, last (blue) – PC.

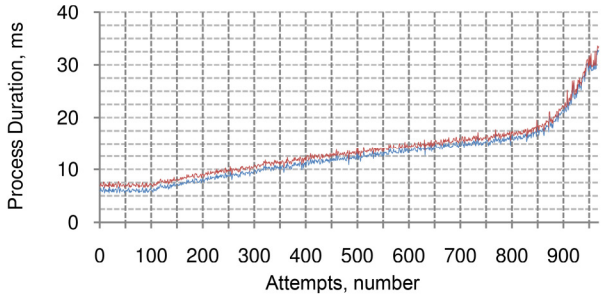

Fig. 9. Process duration of the VLP location search with (red, first from the top) and without (blue, second) area's indexing.

VI. CONCLUSIONS

The electronic license plate recognition system can be offered for intelligent transportation solutions, parking systems, as the clipped image of the license plate is about 20KB.Therefore, the character recognition we believe the central intelligent transportation system server. The proposed algorithm is suitable for image processors, as it is reliable (97.11%) and fast: search of the license plate took 12ms on a PC, DM365 - 207 ms i.MX27 - 165 ms.

A graph shows, that the processing time increases of algorithms exponentially, when increasing the area of the moving object. Compared with the time graphs of the algorithm with proposed area indexing and without it, the PC saves time for 1ms, so the image SoC will save approximately 8ms. The disadvantage of the proposed algorithm is a demanding operating memory(2.75 MB), although they are actually using only 43%.

As we see from Table II, it is possible to reduce a size of s_E and s_C, because the M_E, M_C matrixes and A_C array save index of the searching area. The used corner search method of the minimum eigenvalue is not the best, because the eigenvalue need to save the most of the memory and complex calculation. Therefore, the proposed vehicle license plate search, according to the methodology of the corners, should be adapted to different corners detection algorithm.

REFERENCES

[1] A. Marma, D. Eidukas, M. Žilys, A. Valinevičius, "Intelektualiųjų transporto valdymo sistemų efektyvumas", *Elektronika ir Elektrotechnika (Electronics and Electrical Engineering*), no. 6, pp. 61–66, 2005.

[2] B. Chen, W. Cao, H. Zhang, "An Efficient Algorithm on Vehicle License Plate Location", in *Proc. of the IEEE International Conference on Automation and Logistics,* 2008, pp. 1386–1389.

[3] N. Kehtarnavaz, M. Gamadia, *Real-Time Image and Video Processing: From Research to Reality,* Morgan & Claypool, 2006, pp. 41–43.

[4] *IPNetCam Reference Design on DM365: Application Design Guide*, Texas Instruments, 2009.

[5] B. I. Pawate, *Developing Embedded Software using DaVinci& OMAP Technology,* Morgan & Claypool, 2009, pp. 6–8

[6] *TMS320DM365 Digital Media System-on-Chip(DMSoC),* Texas Instruments, 2009, pp. 1–4.

[7] *TMS320DM6467 Digital Media System-on-Chip*, Texas Instruments, 2010, pp. 1–5.

[8] R. Zhang, Y. Zhang, "Car Number Plate Detection Using Multi-layer Weak Filter", In *Proc. of the Business Intelligence and Financial Engineering,* 2009, pp. 228–232.

[9] X. Zhai, F. Benssali, S. Ramalingam, "License plate localization based on morphological operations", in *Proc. of the Control Automation Robotics & Vision,* 2010, pp. 1128–1132.

[10] J. Kim, Y. Han, H. Hahn, "License Plate Detection Using Topology of Characters and Outer Contour", in *Proc. of the Future Generation Communication and Networking Symposia,* 2008, vol. 3, pp. 171–174.

[11] Z. Qin, S. Shi, J. Xu, H. Fu, "Method of License Plate Location Based on Corner Feature", in *Proc. of the Intelligent Control and Automation,* 2006, vol. 2, pp. 8645–8649.

[12] P. Slapšinskas, "Elektroninės miesto parkavimo sistemos", M.S. thesis, KTU, 2010. (in Lithuanian).

[13] M. G. Roque, R. M. Musmanno, A. Montenegro, "Adapting the Sobel Edge Detector and Canny Edge Extractor for iPhone 3GS architecture", in *Proc. of the 17th International Conference on Systems, Signals and Image Processing (IWSSIP),* 2010, pp.486–489

[14] K. Okarma, P. Mazurek, "Application of Shape Analysis Techniques for the Classification of Vehicles", in *Proc. of the Communications in Computer and Information Science,* Springer, 2010, vol. 104, pp. 218–225.

[15] J. Mikulski, "Using Telematics in Transport", in *Proc. of the Communications in Computer and Information Science,* Springer, 2010, vol. 104, pp. 175–182.

[16] P. Mazurek, K. Okarma, "Background Suppression for Video Vehicle Tracking Systems with Moving Cameras Using Camera Motion Estimation ",in *Proc. of the Communications in Computer and Information Science,* Springer, 2012, vol. 329, pp. 372–379.

[17] K. Okarma, P. Lech, "Application of the Monte Carlo preliminary image analysis and classification method for the automatic reservation of parking space", *Machine Graphics and Vision*, vol. 18, no. 4, pp. 439-452, 2009.