[10] Industrial DataMatrix barcodes recognition with a random tilt and rotating the camera

Kruchinin A.Yu.
Orenburg State University
IntBuSoft Ltd

Abstract

The paper offers an algorithm for the recognition of two-dimensional industrial DataMatrix codes, allowing identify codes applied with shock-point or drip methods on the surface with a complex background. The algorithm can be adapted by changing the binarization mode when changing background code.

Keywords: recognition of barcodes, DataMatrix, image binarization, management of the process of recognition.

Citation: Kruchinin A.Yu. Industrial datamatrix barcode recognition for an arbitrary camera angle and rotation / Kruchinin A.Yu. // Computer Optics. – 2014. – Vol. 38(4). – P. 865-870.

Introduction

The algorithm for the recognition of industrial DataMatrix codes is actively used in the industry when applying on surfaces where it is impossible to apply/paste printed codes for example, on metal pipes. Such codes are applied either with shock-point or drip methods. Industrial solutions existed for the recognition of industrial DataMatrix codes, including those used by SICK AG Company (Germany), do not allow to recognize the codes under conditions of high noise background which occurs when there is insufficient difference between code points and the noise.

Notwithstanding the fact that there are algorithms for the recognition of industrial codes, for example [1, 2], these algorithms have some disadvantages, i.e. the unstable behavior at a random tilt and rotating the camera and greater sensitivity to noises. Therefore, we have developed our own algorithm for the recognition of industrial DataMatrix codes using the results obtained in paper [3]. The following conditions have been considered as algorithm requirements:

- radius sizes of code points vary within 3 to 6 pixels;
- code sizes are from 10×10 to 32×32;
- radius sizes of code points vary within 3 to 6 pixels;
- tilting the camera with regard to the code plain surface is within *90* to *45* degrees;
- rotating the camera may be of any direction (the code can be rotated in the plane at a random angle);
- the surface on which the code is applied and lighting conditions may vary.

The requirements have been laid down based on operation conditions of industrial robots manufactured by Intelmet Technologies Company.

1. Recognition algorithm

As in most pattern recognition methods it is necessary first to detect the original code. For this purpose we may analyze the brightness differences and use the morphological gradient or the method of the detection of binary objects presented in paper [4]. In most cases the code detection is applied to increase the recognition performance when analyzing large images. Since the paper suggests the analysis of images 640×480 pixels in format incoming as a video stream from an industrial robot machine, the detection stage shall be ignored (the code takes a lot of space in the image, so the detection is meaningless). Further execution of the algorithm may be divided into several stages shown in Fig. 1.

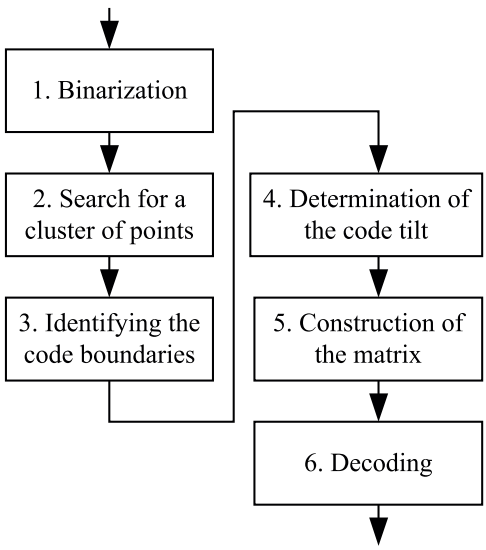


Fig. 1. Stages of industrial barcode recognition algorithm

1. Binarization. The binarization stage is of great importance due to focusing on required information. The full-color or grayscale image I is to be applied to the input, and the dual-color image I_b is to be applied to the output:

$$I_b(x,y) = f_b(I(x,y)), \quad (1)$$

where f_b – is the bunalization function.

Incorrect binarization shall lead either to data loss or to insufficient separation of point features. It is very difficult to select proper binarization optimal for all possible cases. Fig.2 shows an example of the industrial DataMatrix code and processing results of the threshold binarization.

As it can be seen from the figure, it is not always possible to use this binarization to obtain necessary data. Thus, Fig. 2*b* shows that not all data have been selected, and in Fig. 2*c* there is too much noise. For the time being many binary (bimodal) criteria are known, for example, Otsu, Bernsen, Equal, Niblack criteria, etc. These criteria might be used to calculate the optimal threshold; however this approach is not suitable in cases with variable light intensity in various image areas. In this case we can use an adaptive method of binarization in the neighborhood of each point (a similar procedure is given in OpenCV library). However, in the case when the background surface is not a one-color surface, the similar method will generally give a considerable amount of noises and poor binarization quality. Therefore, the use of the above Otsu criterion will provide better results not globally but locally. For example, the paper [5] offers to locally use the Niblack criterion for barcodes binarization. There are also different combinations of local binarization approach with the edge detection methods [6], such as Sobel, Kenny, morphological gradient methods, etc.

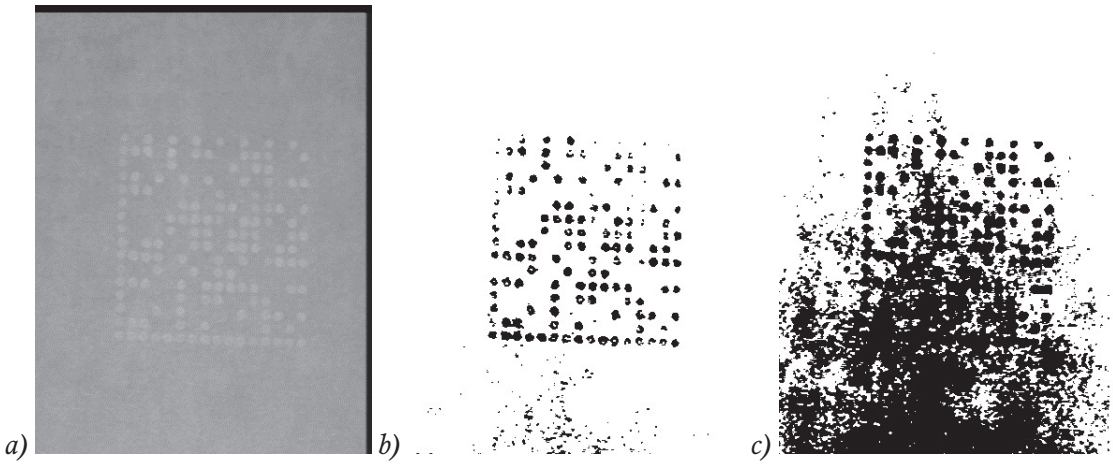


Fig. 2. Industrial DataMatrix code (a), threshold binarization (b), threshold binarization with a lower threshold (c)

A specific feature of the recognition of industrial DataMatrix codes applied on metal surfaces, e.g. on tubes, is a wide spread of recognition conditions (Fig. 3) stipulated by specificity of markers application, metal properties and lighting.

In order to select suitable binarization methods the computational experiment was carried out which aimed at determining how much codes would be recognized as a result. The experiment included the images involving the most of possible lighting conditions, options of code angles, the quality of code application, and the surface quality and color. Based on this features several binarization methods have been selected which are given in Table 1.

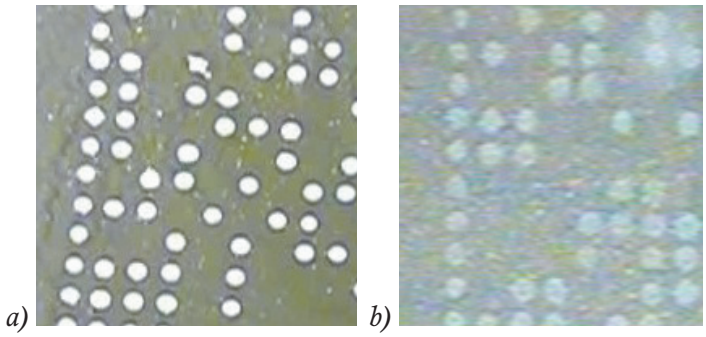


Fig. 3. Examples of recognition conditions for industrial DataMatrix codes

Table 1. Binarization methods and image processing adjustments

	Pre-binarization processing	Binarization	After-binarization processing
1	-	Block binarization	-
2	-	Block binarization	Dilatation
3	Blurring	Block binarization	Dilatation
4	Blurring	Threshold binarization (170)	-
5	Blurring	Threshold binarization (185)	-

Pre-binarization blurring was performed by means of a median filter with a window 3×3 in size. Dilation was used after binarization with the window 3×3 in size. The threshold binarization was fixedly set by the brightness level of 170 and 185 (due to the fact that the lighting is within a certain range, and we may carry out high-quality binarization for a large number of images with codes, the brightness of white points should not fall below 170). Block binarization was performed while operating with image blocks 40×40 pixels in size. The brightness threshold value in the block was calculated based on the following algorithm (in C language syntax):

$$(M + k > p \,\&\&\, M + k < 255)\,?\,t = M + k : t = p \qquad (2)$$

where M – is the mean brightness value in the block, k – is the differentiation factor (the value 30 used), p – is the minimum threshold value (the value 170 used), t – is the resulting threshold brightness value in the block.

However, even all previously described binarization methods didn't allow to process the image with high quality in complex background images. Therefore, one more applied method was the comparison with a shaded circle pattern which characterized the code point. This procedure used a sliding window with w×h pixels in size supplied with an inscribed circle of r radius gradually comparing all pixels of the test image I to compute the correlation by the following formula:

$$R(x,y) = \frac{\sum_{x',y'} \left(T(x',y') I(x+x',y+y') \right)}{\sqrt{\sum_{x',y'} T(x',y')^2 \sum_{x',y'} I(x+x',y+y')^2}} \qquad (3)$$

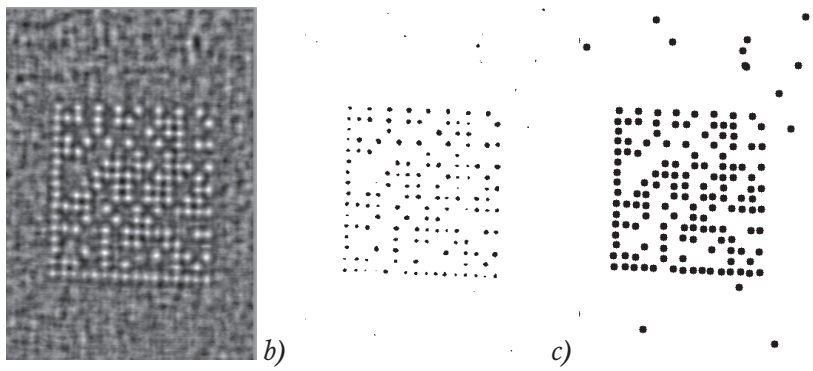
where R – is the comparison result image, x and y – are the resultant image coordinates, $0 \le x' < w$ and $0 \le y' < h$ – are the circle pattern window coordinates, T – is the pattern image. Figure 4*a* shows the 2*a* image processing result. Figure 4*b* shows the threshold binarization result. After that the contour search function was applied for localization of the points, and the point with the specified radius was restored in the center of each found contour (Fig. 4c).

Under the change of recognition conditions, when none of the above methods has been experimentally tested in such conditions (e.g., lower lighting and the code point color starts with the brightness value 150), it may happen that none of the above methods (including that one indicated in Table 1) would be applicable. In such situation it is necessary to perform an additional computational experiment, i.e. the set of test images shall be formed for this lighting, and possible binarization modes and performance evaluation tests are to be sorted out; after that a list of used binarizations shall be developed for these hardware settings. The specified binarization modes happened to be enough under actual operating conditions, and even after a year of industrial operation it was no need to add new modes.

*2. Search for a cluster of point*s. From all specified image areas which can be considered as the points it is necessary to identify the cluster of points which refers to the following code:

$$B = f_c(Contours(I_b)) \qquad (4)$$

where the cluster of B-points is represented as the set $\{p_1, p_2, ..., p_i, ..., p_N\}$, N – is a number of points in the cluster, p_i – is the point presented in the form of (x, y), f_c – is the function selecting a proper cluster of points from the found (recognized) contours on the image I_b.



a) b) c)

Fig. 4. The image with DataMatrix code processed by the correlation benchmarking method with the circle pattern (a), the binarized image (b), the image with restored-size points (c)

The selection algorithm for the cluster of points after binarization:

- search of contours matched in size for the code points (a proper range is to be predetermined); we may compare at this stage the selected contour with the circle (no need if compared with the pattern);
- additional noise filtering – deleting of points that are significantly smaller than the mean value;
- for each of the found points we determine a proper number of neighbors, i.e. the points which are located at a distance less than a certain threshold value; the threshold value is determined depending on the mode of applying the points on the surface, and it must be slightly greater than the maximum distance between the centers of applied points;
- partitioning of points in clusters, i.e. if all points might be sorted out (i.e. all of them shall stay within the threshold distance between the points), all of the points would belong to the same cluster;
- a group with the maximum number of points is to be established and identified (if there are several codes on the image it is possible to identify several groups).

3. Identifying the code boundaries. It is necessary at this stage to identify the exact code boundary:

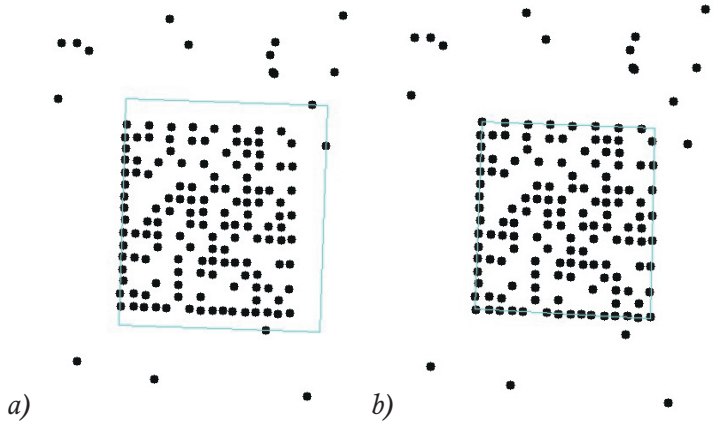
$$P = f_p(B), P = (p'_1, p'_2, p'_3, p'_4) \quad (5)$$

where f_p – is the function which restores the code boundary in the way of four endpoints.

The algorithm consists of two steps:

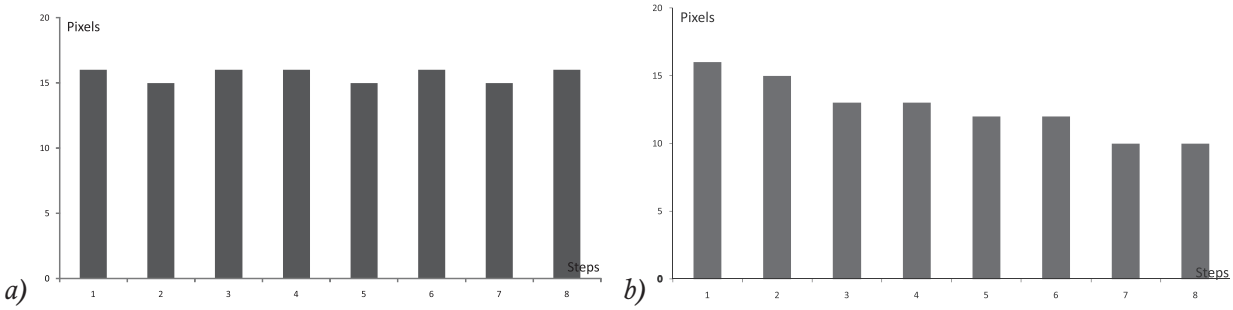
- Identifying a minimum bounding rectangle by any of the known methods (different libraries including the used OpenCV library might support this function); the result is shown in Fig. 5*a*;
- updating each of 4 lines: the nearest points are identified for each line (max 5-8); the lines are generated for each combination of point pairs, and the line crossing the maximum number of points is considered to be the optimal boundary (Fig. 5*b*);
- the edge points are determined by line crossing.

4. Determination of the code tilt. Since the camera axis may not be exactly perpendicular to the code plane, dimensions of code points (cells) will change. In this case the code will not take the form of a square but the form of a rhomb– or trapezoid-like quadrangle [3]. In order to determine dimensional changes of the cells it is necessary to analyze sparse DataMatrix code edges (Fig. 5*b* – the top and right edges). Figure 6*a* shows dimensional changes of the cell pairs in case when the camera axis is almost perpendicular to the plane where the DataMatrix code 16×16 is located. Figure 6*b* shows dimensional changes of the cell pairs in case when the camera is significantly tilted about the axis.



a) *b)*

Fig. 5. Minimum bounding rectangle (a); updated rectangle (b)



a) *b)*

Fig. 6. Dimensional changes in pixels (cells pairs along the sparse edge of the code) with the camera axis perpendicular to the plane (a); with the significantly tilted camera (b)

Any tilt changes should be recorded to be used at the next stage. Therefore, based on the data obtained (Fig. 6) we may calculate linear trends of dimensional changes for both sides:

$$C = (L_1, L_2), \quad (6)$$

where L_1 and L_2 – are the line formulas in the following way:

$$s_i = a * i + b, \quad (7)$$

where s – is the dimension of the cell pair, i – is the pair number, a and b – are the line parameters.

The code sizes, e.g. 16×16 or 32×32, etc., are also to be determined at this stage. This is done by computing the same pairs of cells (Fig. 6).

5. Construction of the matrix. In order to restore the code cell matrix it is necessary to know the matrix boundaries (stage 3), as well as the proportions of dimensional changes of the cells and the number of cells (stage 4).

Optimal sizes of the cell pairs shall be determined for each code side according to changes in formula lines.

$$\sum_{i=1}^{m} s_i = len \quad (8)$$

where m – is the total number of pairs (1/2 of the number of cells), len – is a lateral length.

Despite the fact that the line formulas are defined for two sides, the relative dimensional changes for the cell pairs are also applied to the opposite sides.

Therefore, each side is marked out across the width of the cells, i.e. the points are to be defined on each side. The respective pairs of points from the code opposite sides allow us to build the lines which will form a grid to determine the matrix (Fig. 7*a*).

If the DataMatrix code is detected and if it is well separated after binarization, the matrix will be constructed correctly (Fig. 7*a*). If any errors occurred in binarization or the image area is not considered to be the code, the matrix will be constructed incorrectly (Fig. 7*b*). Information should be read out by determining how the cells have been filled with black cell pixels [3].

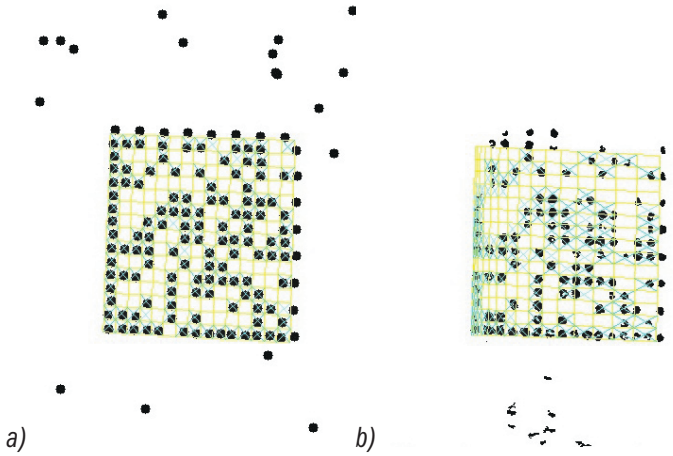


Fig. 7. Correct construction of the matrix after binarization using the template method (a); incorrect construction of the matrix after binarization (item 4, Table 1) (b)

6. Decoding. The stage of information recovery deals with calling the function that is responsible for decoding the Reed-Solomon codes. When the code is severely damaged (or if there is a false candidate for the DataMatrix code), the error or otherwise the text will be returned. There are many free libraries, e.g. libfec, allowing decode the Reed-Solomon codes.

Work results

The industrial DataMatrix codes are used in complicated conditions; it means a variable recognition environment and a limited time required to make a decision. The developed algorithm has been tested and implemented in tube making automation system. The software was developed in accordance with the requirements described in paper [7].

Since recognition conditions may change (Fig. 3), some proper types of binarization might be applicable for some proper conditions. It would be logical to assume that it is necessary to try to identify the code calling in turns the recognition algorithm with different types of binarization until the code is recognized. However, it's not the kind of experience for real-time systems – the operation should be performed within certain time limits. If consider that in addition to the recognition the computer system should perform also some other functions, the recognition time should be minimized. Therefore, the turn-based use of several binarization modes for one code should be strictly limited. For example, two or three methods with regard to CAS performance.

We have used in our experiments five binarization modes given in Table 1 and four benchmarking binarizations with the shaded-circle pattern with 3, 4, 5 and 6 pixels in radius. Three different code groups have been used which differed in their background surface, lighting and sizes of the code points. Totally 500 different codes were used. Table 2 shows the results of recognition. Group 1 – includes very complicated cases which can hardly be detected (they cannot be detected even judging from their appearance). Group 2 – includes the medium complicated cases. Group 3 – includes the simplest cases to be detected.

Table 2. Recognition accuracy for different settings using binarization methods

Binarization	Group 1	Group 2	Group 3
Table 1 – No. 1	0.020	0.008	1.000
Table 1 – No. 2	0.039	0.047	0.853
Table 1 – No. 3	0.176	0.047	0.938
Table 1 – No. 4	0.039	0.333	0.665
Table 1 – No. 5	0.020	0.109	0.684
Pattern – 3 pixels	0.412	0.667	0.000
Pattern – 4 pixels	0.196	0.775	0.059
Pattern – 5 pixels	0.098	0.698	0.327
Pattern – 6 pixels	0.000	0.047	0.004

As you can see in Table 2, one of the binarization methods provides 100% recognition only for Group 3. We won't discuss any false operations here, since they are missing due to the Reed-Solomon coding. In order to increase the number of recognized codes, we can combine various binarization methods, i.e. the recognition of only one type of binarization can be performed first, and if the code cannot be found the other type is to be recognized. Fig. 8 shows, taking as example the subset from 130 elements of Group 2, what methods can be used and how much correct results of recognition may be achieved. If we combine the methods 6 and 7 the accuracy may be increased up to 0.883 if compared to 0.775.

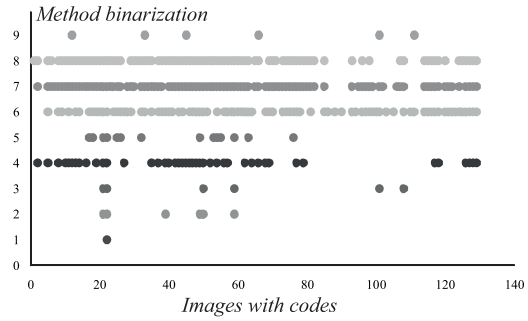


Fig. 8. Recognition results of the test subset using different binarization methods (the circles show the correctly recognized images)

The problem requirements changed gradually – first, there was one lot of tubes in the conveyor, then another. Therefore, it is required to connect with each of the used methods a variable value which will evaluate the method performance based on the number of correctly recognized codes within a certain time interval (or frames). For example, to use three binarization methods, and the method which provides low results by the recognized images needs to be changed by a random approach selected from the available methods.

The developed algorithm and software are used in IntBuSoft Company Ltd (http://intbusoft.ru) and are provided for companies specializing in production automation systems including tube making.

References

1. Dita, I. A scanning method for industrial data matrix codes marked on spherical surfaces / I. Dita, V. Gui, M. Otesteanu, F. Quint // SITE'12 Proceedings of the 11th international conference on Telecommunications and Informatics, Proceedings of the 11th international conference on Signal Processing. – USA, World Scientific and Engineering Academy and Society (WSEAS) Stevens Point, Wisconsin. – 2012. – P. 38-42.

2. Dita, I. Using mean shift algorithm in the recognition of industrial data matrix codes / I. Dita, V. Gui, M. Otesteanu, F. Quint // SITE'12 Proceedings of the 11th international conference on Telecommunications and Informatics, Proceedings of the 11th international conference on Signal Processing. – USA, World Scientific and Engineering Academy and Society (WSEAS) Stevens Point, Wisconsin. – 2012. – P. 174-179.

3. Kruchinin, A.Yu. Algorithm of recognition of two-dimensional graphic codes with any angle of rotation and a chamber inclination: materials of the seventh All-Russia scientifically-practical conference (with the international participation) "the Modern inform technology in a science, formation and practice". – Orenburg: OSU. – 2008. – P. 193-196. (In Russian).

4. Kruchinin, A.Yu. Detection of synthetic binary objects in the scenes observations on the basis of a block image segmentation using histogram features // Herald of computer and information technology. – 2011. – Vol. 11. – P. 9-12. (In Russian).

5. Yang, H. Binarization of low-quality barcode images captured by mobile phones using local window of adaptive location and size / H. Yang, A.C. Kot, X. Jiang // IEEE Transactions on Image Processing. – 2012. – Vol. 21(1). P. 418–425.

6. Satish, M. Edge Assisted Fast Binarization Scheme for Improved Vehicle License Plate Recognition / M. Satish, V.L. Lajish, S.K. Kopparapu // Communications (NCC) 2011 National Conference on. – 28-30 Jan. 2011. – P. 1-5.

7. Kruchinin, A. Yu. Specifics of development of software systems for real-time pattern recognition // Automation and Remote Control. – 2013. – Vol. 74(9). – P. 1599-1605.