Buletinul Ştiinţific al Universităţii “Politehnica” din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICĂŢII

TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 56(70), Fascicola 2, 2011

A Robust Localization Method for Industrial Data Matrix Code

Ion-Cosmin Diţă [1], Marius Oteşteanu [2], Franz Quint [3]

***Abstract*—This paper provides a localization solution for Data Matrix Codes dotted on different materials in different orientations. Knowing the real world size of the Data Matrix pattern and using the parameters of the industrial camera of the recognition system, the developed method can locate the exact position and the orientation of the pattern in an image. We use an adaptive threshold method for the image binarization, in order to be independent of illumination variations or nonuniform background. While the Data Matrix pattern being composed only of dots, it is very difficult to recognize the pattern. To overcome this, we are using morphological operators to transform the pattern in a solid square. The size of the modules and the distance between them as well as the size and the orientation of the data matrix pattern are estimated out of the image. The presented algorithm was tested with very good results for Data Matrix Codes dotted on different materials and different angles.**

I. Introduction

The industrial Data Matrix Code is a two-dimensional matrix bar-code consisting of dots (modules) arranged in a square. The information to be encoded can be text or raw data. Usually data size varies from a few bytes up to 2 kilobytes and can store up to 2,335 alphanumeric characters. The length of the encoded data depends on the code dimension used. Error correction codes are added to increase the robustness of the code size: even if they are partially damaged, they can still be read.

Data Matrix Codes are made of cells: little elements that represent bits. A ”dot” module is a 1 and an ”empty” module is a 0, or vice versa. Every Data Matrix is delimitated by two dotted adjacent borders in an ”L” shape (called the ”finder pattern”) and two other borders consisting of alternating dotted ”cells” or modules (called the ”timing pattern”). Within these borders there are the rows and the columns of cells encoding information. The finder pattern is used to locate and orient the code while the timing pattern provides a count of the number of rows and columns in the code. As more data is encoded in the symbol, the number of modules (rows and columns) increases from 8×8 to 144×144.

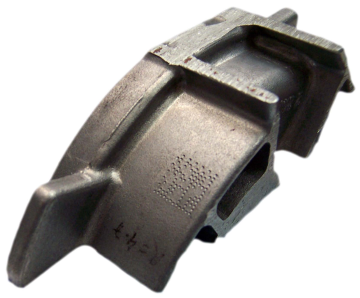
Fig. 1. Industrial Data Matrix Code

For industrial purposes, Data Matrix Codes can be marked directly onto industrial parts, ensuring that only the intended industrial part is identified with the Data Matrix encoded data. The codes can be marked onto components with various methods such as dot-marking (Fig. 1), laser marking, and electrolytic chemical etching. These methods give a permanent mark which should last the lifetime of the industrial part. [1]

II. Presentation of the Image Acquisition System for Industrial Data Matrix Code

The acquisition system is composed of a video camera, a light system and an acquisition software. Using the acquisition software, the system is parameterized with the characteristics of the camera (i.e. focal length, resolution, CCD size) and of the Data Matrix Code (real world size). The video camera is connected to the computer, which by using the *Data Matrix Localization* module processes in real time the images provided by the video camera, giving the position and orientation of the Data Matrix pattern. This module is

[1] Faculty of Electronics and Telecommunications, Politehnica University of Timisoara, Romania, *cosmin.dita@etc.upt.ro*
[2] Faculty of Electronics and Telecommunications, Politehnica University of Timisoara, Romania, *marius.otesteanu@etc.upt.ro*
[3] Faculty of Electrical Engineering and Information Technology, University of Applied Sciences Karlsruhe, Germany, *franz.quint@hs-karlsruhe.de*

connected to the *Video Interface* and the *Data Matrix Scanning*, as is showed in Fig. 2. The *light* is mounted on camera body and creates a 45^o angle with the Data Matrix Code surface. In [2] is explained why is taken a 45^o angle between the light system and the code. Next we introduce very briefly the *Video Interface* module.

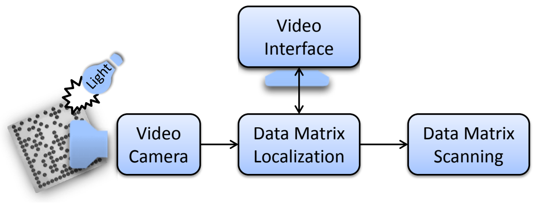

Fig. 2. The block diagram of the acquisition system

III. Video Interface

The interface is the direct connection between user and the acquisition system, displaying the result of the Data Matrix scanning. This module also allows us to parameterize the acquisition system for different types of cameras and different sizes of Data Matrix pattern. Because this system is used in industrial environment, we can choose few characteristics about video camera like: the *CCD size*, the *CCD resolution*, the *focal length of lenses* that are used, information about the real world *code size* and the approximate *distance* between camera and the code. Using these information we can compute the size in pixels of the Data Matrix Code. This computation is an estimation for the size, helping us to restrict the area of searching for the Data Matrix Code. Using equations (1) and (2), we can compute the size of the image in pixels *(I)* as:

$$B = b \cdot \frac{G}{g}, \qquad (1)$$

$$I = B^2 \cdot \frac{w \cdot h}{s}, \qquad (2)$$

where:

G	is the real world Data Matrix size (cm),
B	is the size of Data Matrix projection on CCD,
g	the distance between code and video camera (cm),
b	is the distance between sensor and optical center of the lens, approximate focal length (in case of focus to ∞),
s	is the diagonal length of the CCD (cm),
w	is the vertical number of pixels of the CCD,
h	is the horizontal number of pixels of the CCD.

IV. Data Matrix Localization

The pre-processing stage is used to locate the Data Matrix pattern, without having interest in image details. To identify the correct position of *ROI*, we have to use some information about the shape of the Data Matrix pattern.

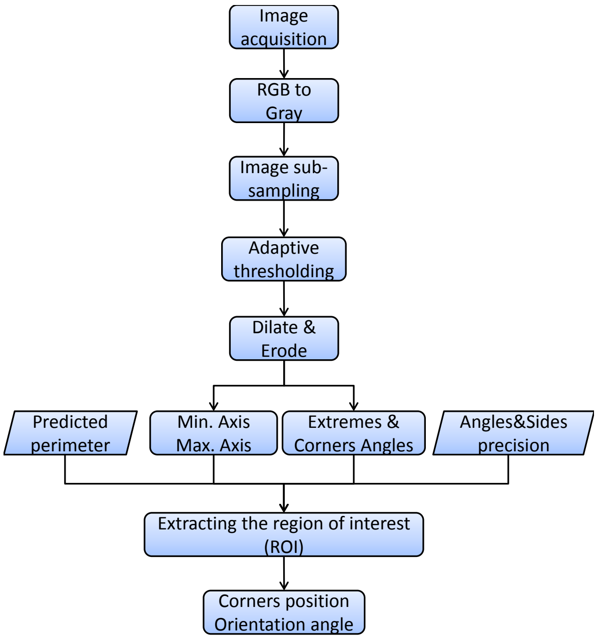

Fig. 3. Data Matrix localization process

If we follow the block diagram of the localization system (Fig. 3), we can see that an image is captured and is sub-sampled using a Sample ratio. If a high Sample ratio is chosen, the image is more decreased and the pre-processing speed is increased, that being a goal for the system. But on the other hand, the object characteristics are considerably reduced. Because of that, the Sample ratio is chosen manually by the operator, depending by the real world size of the pattern.

The image is thresholded using an adaptive threshold level [3]. Depending by the estimated pattern size in pixels, the Gray image is divided in regions, each region being equal with the estimated size in pixels of the code, Fig. 4. The number of region is $round(\frac{\textit{Image size}}{\textit{Estimated pattern size}})$. Using the Otsu method [4] for each region a local threshold level is computed. Otsus method searches for the threshold that minimizes the intra-class variance which is defined as the weighted sum of variances of the two classes, equation 3.

$$\sigma^2_W(t) = W_b(t) \cdot \sigma^2_b(t) + W_f(t) \cdot \sigma^2_f(t), \qquad (3)$$

where W_b, W_f denote the probabilities of the two classes separated by a threshold t, and σ^2_b, σ^2_f denote the variances of these classes. Otsu has proven that minimizing the intra-class variance is the same as maximizing interclass variance:

$$\sigma^2_B(t) = \sigma^2 - \sigma^2_W(t) = W_b(t) \cdot W_f(t) \cdot (\mu_b(t) - \mu_f(t))^2, \qquad (4)$$

which is expressed in terms of class probabilities W_b, W_f and class means μ_b, μ_f, which in turn can be updated iteratively. We maximize formula 4 to get the Otsu's threshold. The procedures of Otsu's method can be depicted as follows: [5]

- It computes the histogram and the probabilities of each intensity value,
- Sets the initial values of $W_b(0), W_f(0)$ and $\mu_b(0), \mu_f(0)$,
- Loops for all possible thresholds t,
- Updates $W_b(t), W_f(t)$ and $\mu_b(t), \mu_f(t)$,
- Computes $\sigma_B^2(t)$ and chooses the threshold t^* corresponding to the maximum of $\sigma_B^2(t)$.

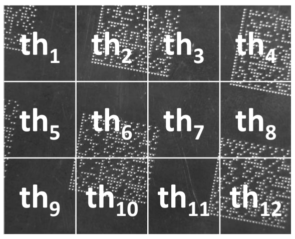

Fig. 4. Local threshold levels

Using all threshold levels is created a matrix of local threshold levels, relation 5.

$$Th = \begin{pmatrix} th_1 & th_2 & th_3 & th_4 \\ th_5 & th_6 & th_7 & th_8 \\ th_9 & th_{10} & th_{11} & th_{12} \end{pmatrix} \tag{5}$$

$$\Longrightarrow \begin{pmatrix} th_1 & \cdots & th_w \\ \vdots & \cdots & \vdots \\ th_{(w-1)\cdot h} & \cdots & th_{w\cdot h} \end{pmatrix} \tag{6}$$

Using bilinear interpolation method [6], the local threshold level matrix is extended to a matrix of threshold levels of the size of the image. One interpolated value is calculated using the weighted average of four neighbors values located on the sampling grid of input matrix, using the next equation:

$$\begin{gathered} f_{(x,y)} = (1-\alpha)\cdot(1-\beta)\cdot f_{(|x|,|y|)} + \\ +\alpha\cdot(1-\beta)\cdot f_{(|x|+1,|y|)} + (1-\alpha)\cdot\beta\cdot f_{(|x|,|y|+1)} + \\ +\alpha\cdot\beta\cdot f_{(|x|+1,|y|+1)}, \end{gathered} \tag{7}$$

α and β are the fractional part of x and y coordinates.

Each pixel from the gray image is thresholded using one threshold level from extended matrix of adaptive threshold levels, showed as in the next relation of cases [6].

$$BW = \begin{cases} 1, & \text{for Img} \geq th_x \ , \\ 0, & \text{for Img} < th_x \ . \end{cases} \tag{8}$$

Using the adaptive threshold method, a binary image is created.

For the next stages of image processing, it is desirable that the background should be black (level 0) and the foreground to be white (level 1). Since each Data Matrix pattern is surrounded by a quiet zone, there are more background than foreground pixels in a ROI. Thus, by counting the values of zeros and ones in the ROI, we can find out the current background level and, if necessary, we can negate the image to have a zero-level background [6].

In the image, there can also be noise, light spots, or other objects. Because the searched object is created only from modules it is harder to identify its position. To recognize the Data Matrix pattern, the code shape should be seen like a square not like a matrix of points. For that, using the morphological dilate operation, the image is dilated in order to fill the empty spaces between modules, Fig. 5. The structural element can be considered like a disk with a *Strel Ratio* pixels radius. This value is chosen depending on numbers of modules.

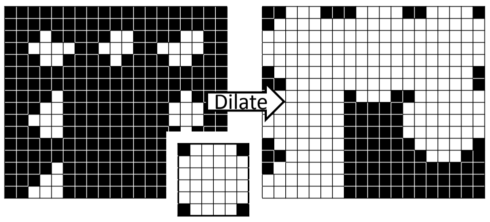

Fig. 5. Morphological Dilate Operation

The dilation process is performed by laying the structuring element B on the image A and sliding it across the image in a manner similar to convolution. If one pixel from the structuring element B coincides with a 'white' pixel in the image, then it turns the origin of the structuring element to 'white' [6]. Let be *E* an Euclidean space or an integer grid, *A* a binary image in *E*, and *B* a structuring element. The dilation of a set A using the B structural element is defined through equation:

$$A \oplus B = \{z \in E | (B^S)_z \cap A \neq \emptyset\}, \tag{9}$$

where B^s denotes the symmetric of B, that is

$$B^s = \{x \in E | -x \in B\}.$$

The generated effect by the dilation operation is to expand the objects. All white objects in the image are connected between, building a square. In the dilated image all object are expanded all around with *Strel Ratio* pixels.

Using the erode operation, the objects are resized to the initial dimension.

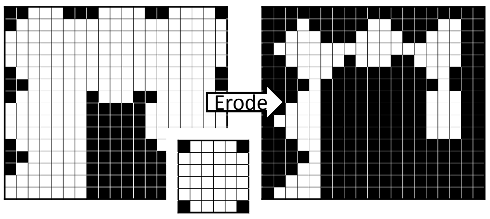

Fig. 6. Morphological Erode Operation

The erosion process is similar to dilation, but we turn pixels to 'black', not to 'white'. As before, the structuring element is sliding across the image. If at least one of the pixels from the structuring element falls over a 'black' pixel in the image, it changes the 'white' pixel in the image that coincides with the center of the structuring element to 'black' [6]. If all pixels from the structuring element falls over 'white' pixels in the image, the pixel that coincides with the origin of the structuring element is not changed and the structuring element moves to the next pixel. The erode of the set A through the structural B element is defined with:

$$A \ominus B = \{z \in E | B_z \subseteq A\}, \tag{10}$$

where B_z is the translation of B by the vector z,

$$B_z = \{b + z | b \in B\}, \forall z \in E.$$

The effect generated by eroding operation is to thin the objects. The two operations connected together are called *Image Closing*, Fig. 7.

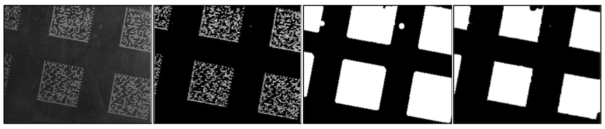

Fig. 7. The image closing process

We can calculate the major and minor axis for each object, like in Fig. 8, using equations 11 - 14.

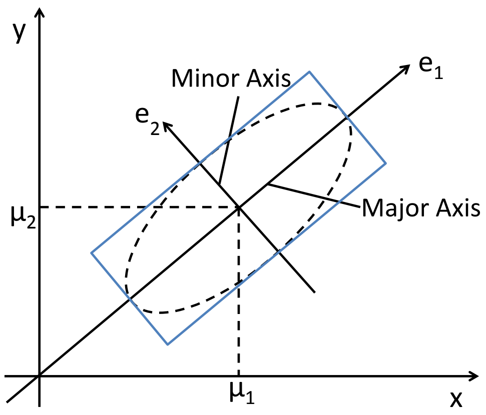

Fig. 8. Association of the object with an ellipse

$$e_1 = \sum x^2, \tag{11}$$

$$e_2 = \sum y^2, \tag{12}$$

$$e_{1,2} = \sum x \cdot y, \tag{13}$$

$$\cos \alpha = \frac{2 \cdot e_{1,2}}{\sqrt{e_1^2 + e_2^2}}. \tag{14}$$

For each object in the image, taking the maximum and the minimum objects coordinates of the points that belongs to each object, we can extract the four corners of each object. Through these points we can draw imaginary vectors, obtaining the angles between these vectors and the main axis of the object. After intersecting two by two vectors, we can compute the angles between, equations 15 - 18. We can see this in Fig. 9.

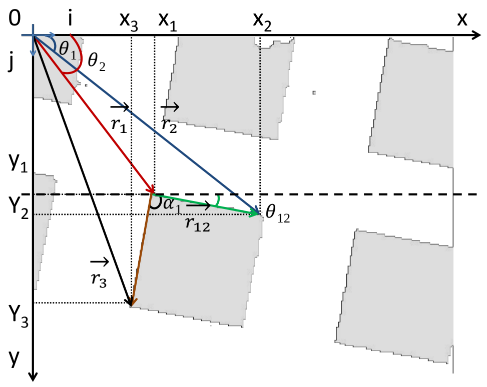

Fig. 9. Angles calculation

$$\vec{r_1} = x_1 \cdot \vec{i} + y_1 \cdot \vec{j}, \tag{15}$$

$$\vec{r_2} = x_2 \cdot \vec{i} + y_2 \cdot \vec{j}, \tag{16}$$

$$\vec{r_{12}} = \vec{r_2} - \vec{r_1}, \tag{17}$$

$$\cos \alpha_1 = \frac{\vec{r_1} \cdot \vec{r_2}}{|\vec{r_1}| \cdot |\vec{r_2}|}. \tag{18}$$

We know that the pattern is a square and also we know the estimated size of the pattern sides. The searched object must meet these conditions:

- all the sides of the pattern should be equal,
- adjacent sides must be orthogonal,
- length of the pattern side should be equal with the predicted value.

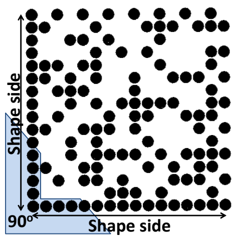

Fig. 10. Data Matrix Code

Is searching in the image just for objects which meet the conditions to be a square, but with a tolerance declared by user for object sides and for angles. Intersecting all the sets of characteristics we obtain the region of interest (*ROI* - Fig. 12) which meets the condition imposed to be a Data Matrix Code.

Because of the perspective errors, the projection on the image sensor of the Data Matrix Code can not to be a square. Perhaps it might be a convex quadrilateral as in Fig. 11 and, to overcame this we use a tolerance for angles and for sides.

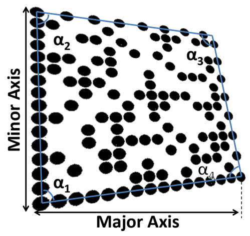

Fig. 11. Data Matrix Code - Perspective error

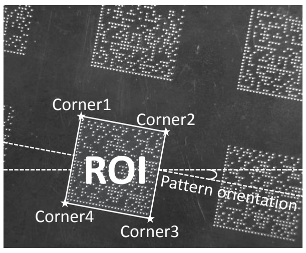

Fig. 12. Region of interest - Data Matrix Code

V. Data Matrix Scanning

The *Data Matrix scanning* block takes the information with the coordinates of the corners and the code orientation and scans inside of the code in few steps:

- computation of the distance between modules,
- the Finder pattern recognition,
- modules scanning,

For computation of the general distance between modules, it analyzes the distance between each module and 4 neighbors of it. The results are written in a matrix of distances. After all modules are queried, all data are stored and the peak of the histogram is the distance between modules.

The finder pattern is composed from two dotted adjacent borders in a "L" shape. To recognize this pattern, using the distance between dots and the code orientation, all 4 corners are queried to outside for neighbors in two direction displaced with 90^o. The corner with two neighbors is the main corner and the other two adjacent corners are the others corners of the finder pattern.

For the modules scanning, it starts to scan from the main corner using the modules distance and the orientation angle of the code. It is searching in rows and columns for each dot creating a matrix of coordinates of dotted and un-dotted modules.

VI. Conclusion

The localization of region of interest (ROI) is an important stage in operation of the image processing process for Data Matrix Reader. Using adaptive threshold level for image binarization, the differences of gray levels are reduced. Closing the image using the morphological operators dilate and erode helps to recognize the code position. Thus the recognition system being more accurate. Because of the Localization system is used for industrial Data Matrix Code recognition, the characteristics of the pattern are known. Thus it can be define a shape of the pattern and in that way the searching area is reduced. In the next figures (Figures: 13 - 16), we give four examples of tests on different types of Datas Matrix Codes.

In Fig. 13, the code is dotted on glossy copper, the size of the pattern is 2 cm and in Fig. 14, the code is dotted on shiny aluminum, the size of the pattern being 1,6 cm.

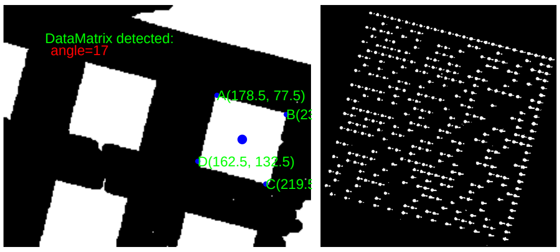

Fig. 13. Glossy copper, pattern size 2 cm

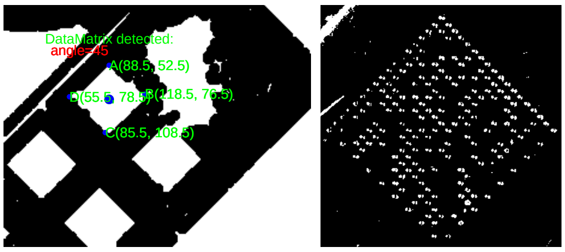

Fig. 14. Shiny aluminum, pattern size 1.6 cm

In Figures 13 and 14, we can see that the real world size of the code and the number of modules are not important for the localization. The code is successfully located in both cases.

In Fig. 15, the code is dotted on iron with rust spots, the size of the pattern is 1,6 cm and in Fig. 16, the code is dotted on metals coated with texture, the size of the pattern being 2 cm.

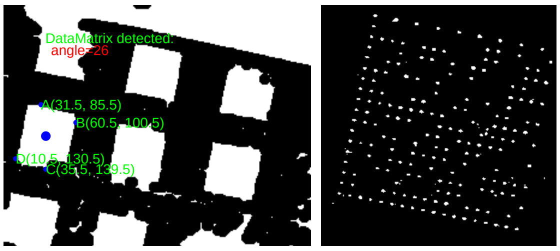

Fig. 15. Iron with rust spots, pattern size 1.6 cm

In these two cases we can see that the background of the code is not important in the image localization process. In both cases the spots of rust and paint doesn't affect the localization system.

The tests are executed in an environment with one light source mounted on 45^o to the code surface. This

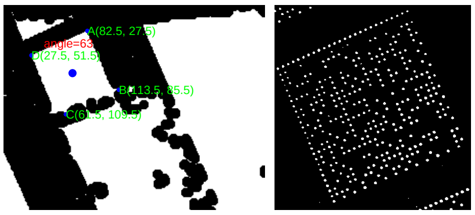

Fig. 16. Metals coated with texture, pattern size 2 cm

stage, of image pre-processing, works in real time with good results for materials that have the property of light reflection. In the cases when plastic materials were tested, it is hard to recognize the position in the image of Data Matrix pattern. Because in these cases the light reflection is to small or null especially for white color.

VII. Acknowledgment

This work was partially supported by the strategic grant POSDRU 6/1.5/S/13, (2008) of the Ministry of Labour, Family and Social Protection, Romania, co-financed by the European Social Fund Investing in People.

References

[1] INTERNATIONAL STANDARD, "Information technology — International symbology specification — Data matrix," 2000-05-01.

[2] Dita Ion-Cosmin and Otesteanu Marius, Eds., *Factors that Influence the Image Acquisition of Direct Marking Data Matrix Code*, vol. TELFOR 2009, Serbia, Belgrade, 2009.

[3] Ye Zhang, Hongsong Qu, and Yanjie Wang, "Adaptive Image Segmentation Based on Fast Thresholding and Image Merging," *Artificial Reality and Telexistence–Workshops, 2006. ICAT '06. 16th International Conference on*, pp. 308–311, 2006.

[4] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man and Cybernetics, DOI - 10.1109/TSMC.1979.4310076 (Systems, Man and Cybernetics, IEEE Transactions on)*, vol. 9, no. 1, pp. 62–66, 1979.

[5] L. W. U. Yudong ZHANG, "Fast Document Image Binarization Based on an Improved Adaptive Otsu's Method and Destination Word Accumulation," vol. JCIS 2011 Vol. 7 (6) : 1886- 1892, 2011.

[6] V Gui, D Lacrama, and D Pescaru, *Prelucrarea imaginilor*. Editura Politehnica Timisoara, 1999.