Journal of Autonomous Intelligence (2023) Volume 6 Issue 2
doi: 10.32629/jai.v6i2.597

REVIEW ARTICLE

Localization of image fragments with high frequency intensity oscillation

Andrey Trubitsyn[1,*], Maksim Shadrin[2], Andrey Serezhin[1]

[1] *Electronics Department, Ryazan State Radio Engineering University, Ryazan 390000, Russia*

[2] *Kvantron Group LLC, Ryazan 390000, Russia*

*** Corresponding author:** Andrey Trubitsyn, assur@bk.ru

ABSTRACT

The problem of detecting image fragments characterized by high-frequency fluctuations in spatial intensity in the general case has not been previously considered in the literature. The article researches a sequence of known and new algorithms that allows detection and localization of such fragments. The geometric localization of the fragments is based on the Hough transform of the pixel array of the external contours of the connected components. The components connecting becomes possible due to the use of the oscillation function proposed by the authors. The oscillation function turns out to be an effective tool for highlighting intensity fluctuations zones in an image and is superior in reliability to alternative methods for detecting such zones, based, for example, on gradient methods. The article demonstrates examples of localization of the image fragments with different levels of background complexity.

Keywords: pattern recognition; salient object; integral sum image; binary image; connected component method; Hough transform

ARTICLE INFO

Received: 2 June 2023
Accepted: 19 July 2023
Available online: 15 August 2023

1. Introduction

In recent years, the rapid development of practical applications for the processing and analysis of images has clearly been noticed. In automotive industry[1], pharmaceuticals[2], in the police[3] and the armed forces[4], in automated production and measuring systems[5] various image processing systems are widely used. The subject of development in this area is efforts to convey the ability of human vision to electronic perception and understanding of the image. The purpose of computer vision and image processing is to determine whether these images contain or do not contain a specific object, function or activity. The problem of object detection is solved by methods of pattern recognition theory. The analysis of the state-of-the-art[6] in this area shows that the theory of recognition is based on models such as the Statistical Model, the Structural Model, Template Matching Model, Neural Network Based Model, Fuzzy Based Model, Hybrid Model. For a wider consumer, these models can be combined into three classes of methods: methods based on the color and intensity features of the localized fragment; methods based on identifying the shape of the localized fragment and its characteristic features; methods based on machine learning algorithms. Currently, the best algorithms for solving pattern recognition problems are based on a neural network. Their performance is close to people's abilities[7]. Nevertheless, alternative methods of pattern recognition are being developed, which are based

on the analysis of the shape and intensity features of fragments and provide a more efficient solution of the tasks in specific situations[8].

The work given is devoted to the techniques for recognizing patterns having specific visual characteristics. The main goal of the study is to ensure high reliability of detection and localization of such patterns. High frequency of intensity oscillations in fragments has been chosen as one of required specific features. This feature characterizes many artificial objects resulted from human activity: road labeling, attention-grabbing billboards, and finally, barcodes (matrix, QR-codes) being the labels of products for various purposes. The latter can be considered as the example that not only catch human attention but also provide convenient means to record and transmit information. The role of this way to deliver information for the consumers concerned is increasing. Useful and urgent information in the form of alternating geometric shapes, e.g., can be recorded on special shields or walls of buildings, etc. After automatic reading such information, e.g., city information, can be voiced for individual or group consumption, or imply a certain reaction of unmanned vehicles in a specific location. The information of such type is evident to be limited by the regions with defined and uncomplicated geometric shape (triangles, quadrangles, pentagons). Therefore, the problem to develop the algorithms of reliable detection and precise localization of image fragments with high-frequency spatial oscillations limited on image surface by the frames of simple geometric shapes is set.

We should also consider that similar problems are solved in a comparatively new direction of image processing when detecting the so-called salient objects[9,10]. Computational identification of such salient object regions is very challenging. The ability for automatic, efficient, and accurate estimation of salient object regions, however, is highly desirable given the immediate ability to isolate the object from potentially confusing background. The attributes of salient objects, allowing us to detect them visually, are color, gradient, boundaries and high contrast[11]. From the point of view of this field of pattern recognition, in our work, we study a sequence of algorithms for separating salient objects by identifying fragments of images, within the boundaries of which pronounced periodic intensity fluctuations are observed.

The main contributions of the work lie in the formulation of the problem of detecting image fragments that are distinguished by high-frequency intensity fluctuations and in the study of a sequence of algorithms that ensure such fragments selection with a high degree of reliability. As an indicator of intensity oscillations in different parts of an image, it is proposed to use a new point characteristic, which is the total number of intensity jumps in the vicinity of each pixel of the binary image. It is the calculation of this characteristic, called the oscillation function, which is the key procedure that guarantees the reliability of the problem posed solution.

2. Materials and methods

This section provides the general methodologies used. The methodology for detecting and localizing image fragments, which is considered below, is based on a strict mathematical basis and consists in the following sequence of data processing procedures, where after the arrow is the procedure output:

1) image thresholding → an image.
2) calculation of the two-dimensional oscillation function (the key procedure) and thresholding the oscillation function image → an image.
3) labeling the connected components of the thresholded oscillation function → an image of the labeled components.
4) selection of the boundary contour of each connected component → a pixel array.
5) Hough transform of the pixel array of each contour → a matrix.
6) search for the first k local maxima of the Hough transform → coordinates of *k* maxima in Hough space.

7) determination of the vertices of the localizing *k*-gon in the Cartesian coordinate system → coordinates of the *k* vertexes.

Steps 4–7 are repeated for each labeled component.

In order to increase the count rate, most of the presented algorithms use precalculated integral sum image.

2.1. Integral sum image

To accelerate several procedures of image processing the integral sum image is used[12]. Integral sum image $i(x, y)$ is matrix $I_{x,y}(i)$, coinciding in size with an original image sized $n_x + n_y$. Each element of it keeps the sum of intensities for all pixels located to the left or higher of the element given. Each element of matrix $I_{x,y}(i)$ represents the sum of pixel intensities in a rectangular from (0, 0) to (x, y), i.e., the value of intensity of each pixel (x, y) equals the sum of all pixel intensities located to the left or higher of the pixel given (x, y). The calculation of a matrix takes linear time proportional to the number of pixels in an image, thus integral image is calculated in one pass.

Matrix calculation is possible to be made according to the formula:

$$I_{x,y}(i) = i(x,y) + I_{x-1,y}(i) + I_{x,y-1}(i) - I_{x-1,y-1}(i),$$

where $x \in [0,\ n_x - 1],\ y \in [0,\ n_y - 1]$.

Integral sum image allows fast calculation of the sum for each original image intensity in a rectangular window of arbitrary size $(2m_x + 1) \times (2m_y + 1)$ with the center in pixel (x, y) by four (corner) values of integral sum:

$$D(x,y) = I_{x+m_x,y+m_y}(i) - I_{x-m_x-1,y+m_y}(i) - I_{x+m_x,y-m_y-1}(i) + I_{x-m_x-1,y-m_y-1}(i),$$

where independent m_x and $m_y = 1, 2, \ldots$

2.2. Binarization algorithms

Various approaches for image binarization can be singled out. According to main classifier the binarization methods can be divided into two groups:

1) threshold methods,
2) methods providing the equality of sum intensities for original and binary images.

Threshold methods subdivide image pixels as background and object ones in relation to a certain intensity threshold. This threshold can be global, i.e., calculated by different image features (average intensity, intensity histogram, etc.), or local, where the value of each pixel is determined by ambient pixel intensity.

Threshold methods, in contrast to the methods of second group, ignore small details and therefore are largely used when localizing objects in an image or when detecting the objects making them suitable for the solution of the task set in the work given.

Some tasks assign the indication of global binarization threshold to a user himself. However, the class for these tasks is quite small and its solution tends to be highly subjective. Thus, most widely spread algorithms nowadays are the algorithms of automatic threshold value selection.

Binarization method being widely used in practice considers global image average $t = \mu_G$ as a threshold. This method can be recommended to be applied in a large number of cases, especially since the global average can be quickly calculated with the help of integral sum image.

$$\mu_G = I_{k_x k_y}(i)/n_x n_y \qquad (1)$$

where $k_x = n_x - 1$, $k_y = n_y - 1$.

A more accurate image intensity feature compared to a global average is an image histogram. The most popular method where the choice of threshold value is made on the basis of image histogram analysis is Otsu

method[13]. The choice of a threshold in this method is based on minimizing intragroup dispersion for two groups of pixels separated by threshold *t*. Otsu showed that dispersion minimization inside the class is equal to dispersion maximization between the classes:

$$\sigma_b^2(t) = \omega_1(t)\omega_2(t)[\mu_1(t) - \mu_2(t)] \quad (2)$$

where $\omega_1(t)$—is the probability for semitone $p < t$ occurrence, $\omega_1(t)$—is the probability for semitone $p > t$ occurrence, $\mu_i(t)$—arithmetic mean of semitones in class *i*. Therefore this method searches for threshold *t*, which maximizes Equation (2). The simplest way to find the maximum is the sequential change of *t* from 0 to 255 for two byte images and selecting *t* that corresponds to maximum $\sigma_b^2(t)$ from the series of the values received.

In case of uneven illumination of the surface studied or the presence of structural defects on this surface quite a significant portion of information when binarizing image as a whole is lost. To solve this problem, methods of local binarization are used. The standard adaptive thresholding technique computes an individual threshold value for each point in the image. Threshold value is calculated as the average intensity of points under the sliding window[14]. The other commonly used techniques are Niblack[15] and Sauvola[16] ones. In work[17] the authors proposed an improved local thresholding (view Equation (3)), that requires no square root calculation for each point to calculate a standard deviation and is more stable to the selection of parameters. The modified local thresholding technique, which combines point intensity, local mean and deviation under the sliding window, calculates a local threshold value according to:

$$t(x,y) = \mu(x,y) - \frac{i(x,y)}{m_1} - \frac{s^2(x,y)}{m_2}, \quad (3)$$

where $\mu(x, y)$ is a local mean, $s^2(x, y)$ is a local variance of the pixels under the sliding window at coordinates (x, y), k_1 is a constant that is controlling penalization of bright points, k_2 is a constant that is controlling decreasing of local threshold for points in which neighbourhood intensity significantly varies (k_1 and k_2 were empirically set to 10 and 120 respectively and half-size *n* of sliding window was 17).

To speed-up threshold calculation, precalculated integral sum image should be used:

$$\mu(x,y) = \frac{\left[I_{x+n,y+n}(i) - I_{x+n,y-n-1}(i) - I_{x-n-1,y+n}(i) + I_{x-n-1,y-n-1}(i)\right]}{(2n+1)^2},$$

$$s^2(x,y) = \frac{\left[I_{x+n,y+n}(i^2) - I_{x+n,y-n-1}(i^2) - I_{x-n-1,y+n}(i^2) + I_{x-n-1,y-n-1}(i^2)\right]}{(2n+1)^2} - \mu(x,y).$$

The evaluation of recovered quality of useful information when binarizing images by the methods considered earlier is shown in **Figure 1** where the original and its corresponding binary images are presented.

(a) **(b)** **(c)** **(d)**

Figure 1. Thresholding of grayscale image: **(a)** original image; binary image resulting from **(b)** global; **(c)** Otsu and **(d)** local thresholding.

2.3. Oscillation function

In order to select regions with intensity oscillations the image gradient or some other similar differential operators are usually calculated.

Frequently gradient operators include Roberts, Prewitts, Sobel, Kirsch, Laplacian and LOG operators[18]. Considering computing time and noise squelch, the improved Sobel kernel method is recommended in works[19,20] to compute gradient and determine direction. Four Sobel kernel operators representing 0°, 45°, 90° and 135° directions are applied to each pixel (x, y) and the maximum of the four values is used as an intensity

gradient of a pixel (*x*, y). Then the gradient image should be transformed to binary image, for example, by the Otsu method.

However, direct application of this technique[19,20] to an original image gives no necessary effect (see further information) as it doesn't lead to distinctive extraction of desired region. According to work[19], the image can be reduced preliminary to a quarter of the original one with bilinear interpolation, then it must be transformed into gray image, and finally, image contrast should be enhanced. This sequence of procedures increases time costs, and it is not strictly justified, therefore, cannot be considered reliable in general. Such situation is connected with the fact that the gradient of a function in its physical sense shows the direction of maximum increase/decrease of a function, as well as quantitatively characterizes the curvature of a function at each point (in different areas). Therefore, such gradient is not quite suitable for the detection of highly oscillating functions.

As an indicator of intensity oscillation in different areas of a binary image the present work offers to use a new point characteristic $v(x, y)$, that represents the total number of p_x and p_y, "black-white" and "white-black" intensity jumps by x and y axes in the vicinity of each pixel (x, y) of a binary image. We shall call this two-dimensional function $v(x, y)$ as an "oscillation function". Pixel vicinity here is understood as a square window sized $k \times k$ with the center in its current pixel (x, y) (**Figure 2a**). Here, $v(x, y)$ estimation is admissible in two norms: $v(x, y) = max(p_x, p_y)$ or $v(x, y) = (p_x + p_y)/2$, which in practice, however, give almost identical results. At first glance, the calculation of two functions $p_x(x, y)$ and $p_y(x, y)$ with the help of sliding window along the whole original binary image may seem quite a costly procedure, but this is not true as in this case we can also make use of image integrals, to be more exact, the integrals of preliminary shaped images $i_x(x, y)$ and $i_y(x, y)$. Each pixel in these images takes 1 information bit, i.e., it can have the value of 0 or 1 depending on the presence of intensity jump in the vicinity to the left or to the right, then $i_x(x, y) = 1$, otherwise $i_x(x, y) = 0$ (**Figure 2b**); or on the presence of relative intensity jump in the vicinity above or below, then $i_y(x, y) = 1$, otherwise $i_y(x, y) = 0$ (**Figure 2c**). We should emphasize one more time that a binary image is considered to be the source for analyzing and forming the images $i_x(x, y)$ and $i_y(x, y)$. The example given in **Figure 2** shows the following number of intensity jumps $p_x(x, y) = 46$ and $p_y(x, y) = 2$, therefore the value of oscillation function in pixel (x, y) equals $v(x, y) = max(p_x, p_y) = 46$.

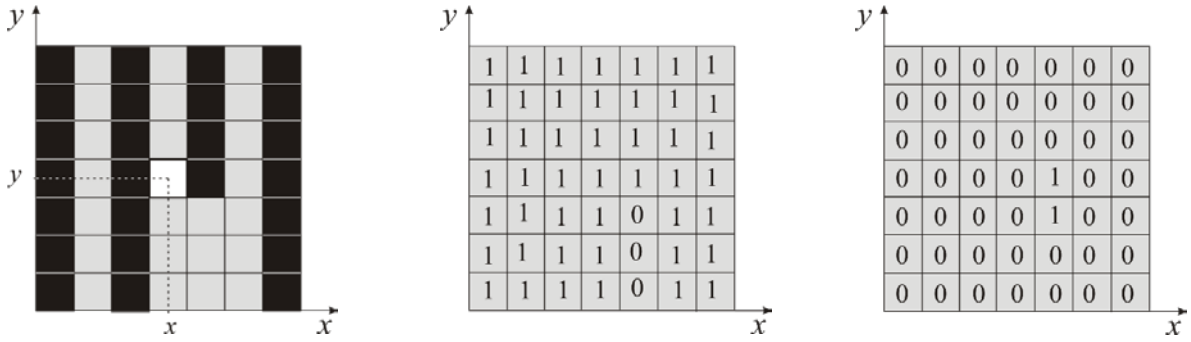
In accordance with the information given above fast calculation of functions $p_x(x, y)$ and $p_y(x, y)$ is possible to be made on the basis of the following equation:

$$p_x(x, y) = I_{x+n,y+n}(i_x) - I_{x+n,y-n-1}(i_x) - I_{x-n-1,y+n}(i_x) + I_{x-n-1,y-n-1}(i_x),$$
$$p_y(x, y) = I_{x+n,y+n}(i_y) - I_{x+n,y-n-1}(i_y) - I_{x-n-1,y+n}(i_y) + I_{x-n-1,y-n-1}(i_y),$$

where n is a semi-size of a sliding window k, i.e., $k = 2n + 1$. It's worth mentioning, that in case when $n = 7$ ($k = 15$), then the values of functions p_x and p_y will be located in the range of values between 0 and 255, its boundaries coinciding the intensity range of shades of gray for 8-bit monochrome images. This size of a sliding window can be chosen as a default parameter, but this parameter is free to be optimized in some certain cases.

The way the oscillation function operates for binary images is demonstrated in **Figure 3**. Image fragments with high intensity oscillation frequency turn into solid figures when image in **Figure 3a** is processed by oscillation function, and solid monochrome figures are left only with contours (**Figure 3b**). The binarization of oscillation function only emphasizes the fact given (**Figure 3c**).

Figure 4 demonstrates the example of oscillation function calculation with the aim to detect a fence on illuminated forest glade that has all features of the object observed with high frequency intensity oscillations.

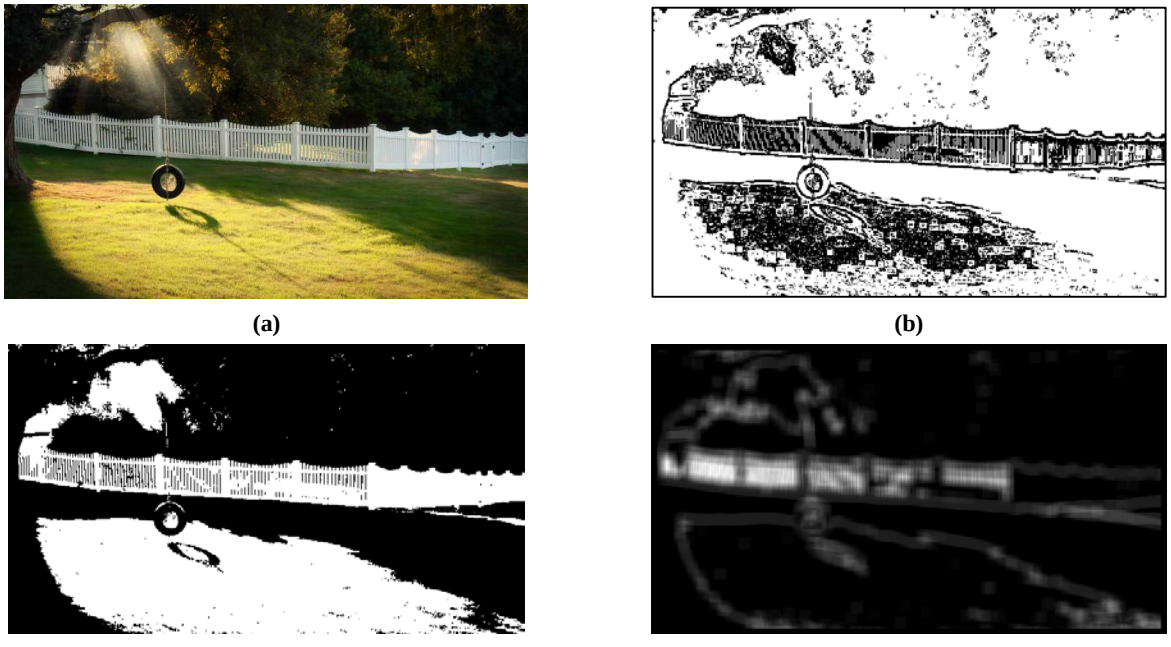


(a) **(b)** **(c)**

Figure 2. Example of oscillation function calculation: **(a)** a model of binary image fragment in a window sized 7 × 7 with a central pixel (x, y); **(b)** function $i_x(x, y)$ in the same window; **(c)** function $i_y(x, y)$ in the same window.

(a) **(b)** **(c)**

Figure 3. Calculation of oscillation function when processing binary images: **(a)** original binary image; **(b)** two-dimensional oscillation function of original image in grayscale; **(c)** oscillation function after Otsu tresholding.



(a) **(b)**

(c) **(d)**

Figure 4. Extraction of high frequency image areas when calculating an oscillation function: **(a)** original image; **(b)** image processed by Sobel operator; **(c)** image tresholding by Otsu method; **(d)** oscillation function of the binary image (c).

Processing of an original image shown in **Figure 4a** by Sobel operator (with further binarization) gives possibility to select the area of interest in the image leaving, however, those spurious well-lit regions of complex background that cannot be eliminated (**Figure 4b**). This result cannot be considered positive. Similar problems can arise in the course of local or threshold image binarization (see **Figure 4c**). All these drawbacks are not found in a method of binarized image processing by oscillation function $v(x, y)$ offered by the authors. **Figure 4d** shows in grayscale the two dimensional function $v(x, y)$ being calculated for the binary image in **Figure 4c**. The analysis of an image shown in **Figure 4d** allows making a conclusion that oscillation function

provides clear and reliable extraction of intensity oscillation regions (in our case, frontal fence links) from complex background even with strong uneven illumination.

2.4. Connected component analysis

The final aim of all processing procedures mentioned above is to select specific fragments of images as the regions being separated from each other. These regions in terms of image processing are called connected components or blobs.

To label ("color") blobs for further analysis and processing a connected component analysis is used[21]. Quite a large number of algorithms to connect the components can be singled out; the most popular of them are as follows:

1) recursive search algorithms based on undirected graph.
2) two stage method based on the algorithm of scanning and merging of equivalence classes.

Here the pixel that hasn't received its label in the process of coloring is offered to be called as black.

For each connected component the algorithms of the first type find their first "black" pixel in the process of sequential scanning of a binary image, the recursive "coloring" of the whole component found is made. When labeling each k-th component the intensity of its pixels is determined to be equal to grayscale k, its square is additionally calculated as

$$S(k) = \sum_{x,y} b(x,y),$$

where

$$b(x,y) = \begin{cases} 1, if\ i(x,y) = k \\ 0, if\ i(x,y \neq k \end{cases},$$

as well as minimum $x_{min}(k)$, $y_{min}(k)$ and its maximum boundaries $x_{max}(k)$, $y_{max}(k)$ by x- and y-axes together with the coordinates of center $x_c = [x_{min}(k) + x_{max}(k)]/2$, $y_c = [y_{min}(k) + y_{max}(k)]/2$.

2.4.1. Simplest recursive algorithm of coloring

In a simplest procedure after a current pixel is colored the same procedure is used for coloring the first "black" pixel of the four neighboring ones. If all neighbors turn out to have non-zero label, the procedure returns to the previous level where the neighboring black pixels are probed. This definition of a function is recursive. Recursion allows simplifying the recording of other algorithms. For this algorithm, however, recursion causes significant problems as recursive calls of coloring function are made for each pixel which leads to stack overflow while a computer program is executed. As practice shows, this algorithm is not applicable for the figures squared higher than one thousand pixels.

It's also worth mentioning, that similar algorithm can be built without any recursion if stack is replaced by separate arrays that store pixel coordinates during coloring procedure. In this case no stack overflow is recorded.

2.4.2. Recursive line filling algorithm

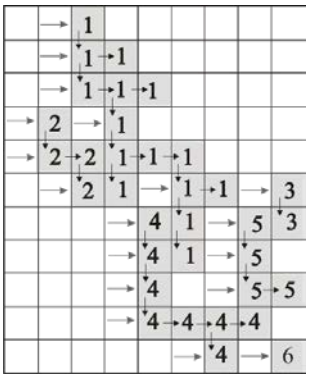
This algorithm is considered to be quite widely popular in computer graphics. It differs from the simplest recursive algorithm by a horizontal line which is placed between contour pixels and drawn on each step of coloring. The algorithm is a recursive one but here as the call of function is made for a line but not for each separate pixel the number of nested calls is reduced in proportion to the length of a line, thus decreasing the overload on computer stack memory.

2.4.3. Two-pass component labeling algorithm

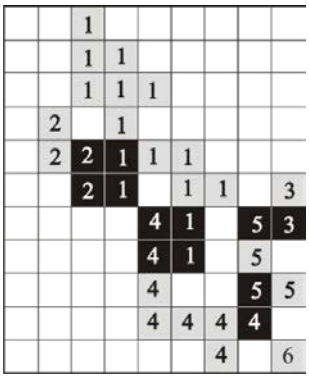
The algorithms of such type are intended to process large-sized images. Further we shall consider a classical algorithm of connected components labeling based on data structure for combining or search.

During the first pass the algorithm tries to make a new label not only for non-labeled pixels but also to distribute the labels among the neighbors that are placed to the right or below already labeled pixels. In case when two different labels can be spread for one and the same pixel the label with lower value is chosen. **Figure 5a** represents a model of a binary image, a scheme of progressive north-to-south/west-to-east scanning together with assigning new labels (large horizontal arrows) as well as the distribution of the labels assigned to a neighbor below or to the right (small arrows). During the first pass together with label assignment the algorithm determines equivalence relations of temporary labels shown in **Figure 5b** as black squares with white label numbers. Then the multitude of equivalence relations is stored as a table (**Figure 5c**, **Table 1**), where each temporary region (label) can have one or several equivalent labels with smaller numbers, in our example region 5 has equivalent labels 3 and 4. In the process of table processing that contains all equivalence relations recorded (see **Table 1**) all equivalence classes are determined (**Figure 5c**, **Table 2**). The example presented in **Figure 5** has 2 classes detected: 1 and 6, therefore, in accordance with the data in the table, labels 1, 2, 3, 4, 5 are assigned with equivalence class 1, label 6 is assigned with class 6. The reason for label 3 being also assigned with class 1 is as follows: as label 5 is equivalent to 3 and 4, and label 4 is equivalent to 1 then label 5 is also equivalent to 1 leading also to equivalence 3 and 1. Thus, of all equivalence relations we choose only one label, e.g., with minimum number, which is assigned to be a class number.

The second pass assigns to each pixel of an output image the labels of its equivalence class (**Figure 5d**). After all connected components are labeled their square S, dimensions $x_{min}(k)$, $y_{min}(k)$, $x_{max}(k)$, $y_{max}(k)$ and center x_c, y_c can be determined.



(a)



(b)

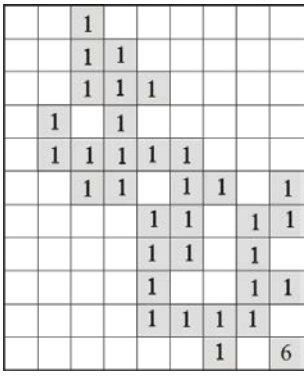
Table 1. Equivalence relations.

Label	1	2	3	4	5	6
Equivalent I	0	1	0	1	3	0
Equivalent II	-	-	-	-	4	-

Table 2. Equivalence classes.

Label	1	2	3	4	5	6
Equivalent	0	1	1	1	1	0

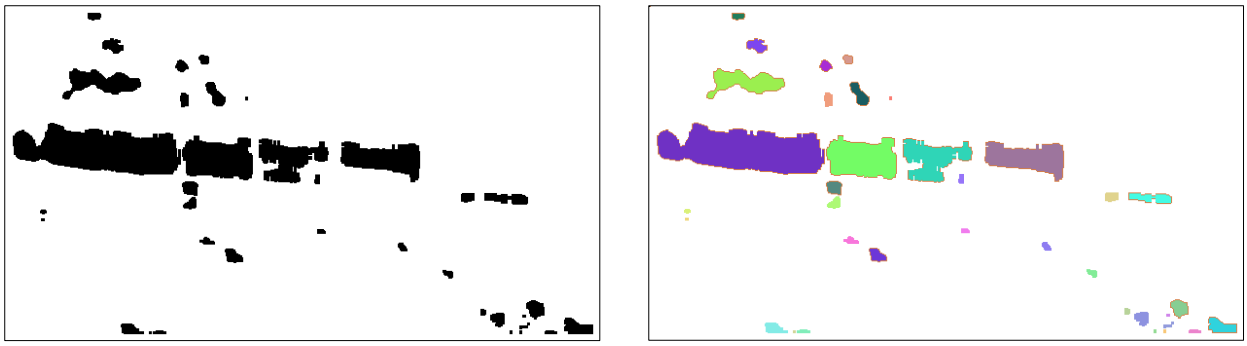
(c)



(d)

Figure 5. Application of classical algorithm based on combination-search data structure to a binary image: **(a)** binary image labeling scheme; **(b)** equivalent regions extraction on labeled image; **(c)** data structure to combine equivalent regions; **(d)** labeled image after equivalent regions combination.

Figure 6b shows the result of connection ("coloring") of binary image components (**Figure 6a**), its origin being a two-dimensional image in grayscale out of oscillation function (**Figure 4d**) that was calculated for the image of illuminated forest glade with a fence (**Figure 4a**).



(a) **(b)**

Figure 6. Connection ("coloring") of binary image components: **(a)** binary image; **(b)** connected ("colored") components.

2.5. Edge contour detection

In order to localize detected regions with specific visual features we need to detect their outer contour. To detect outer contour in connected components various methods are used.

The simplest way to obtain the edges of an object in a binary image is to blur or defocus the image and subtract the original one from it.

In fact a number of other simple algorithms of spatial filtration for edge detection even in semi-tone images can be found, such as Laplacian[22], Canny[23], etc.

The Laplace operator refers to a filtering procedure with spatial increase of sharpness that represents the sum of second private derivatives by coordinates. Discrete analog of Laplace operator is used to detect object edges in an image. The edges are formed from a various pixels where Laplace operator takes zero values as the zeros of second derivatives of a function correspond to extreme jumps in intensity.

Canny edge detector[23] is constructed on the basis of Sobel operator[24] or Scharr operator[25] which is considered to be an increased version of Sobel operator being based on calculating the first derivatives (magnitudes and directions) of pixel intensity function by means of applying these derivatives. The direction of gradient here is the maximum of gradient value in four possible directions 0°, 45°, 90°, 135°.

The disadvantage of methods is uncontrolled width or breaks of contours received, therefore, to extract a true edge closed contour of 1 pixel width we need their additional processing.

Nevertheless, the problem of contour detection is much wider than we present here, with many classes of methods that allow its successful solution[26]. Fortunately, our problem to localize the region being a polygon does not imply accurate contour construction as it is sufficient to label the main edge pixels transmitting outer region outlines and apply, e.g., Hough transform to the array of the pixels sorted. Thus, to solve the task of this kind we offer an obvious scan line algorithm by the lines from the left to the right and from the right to the left as well as by the columns from bottom to top and from top to bottom. **Figure 7** shows the scheme of image scanning containing 3 connected components 1, 2 and 3 as well as of detecting outer contour of the first component marked in gray color in **Figure 7**.

To accelerate the process of scanning we can limit the scanned area by the sizes of the component labeled 1, in our case, by axes *x* and *y*.

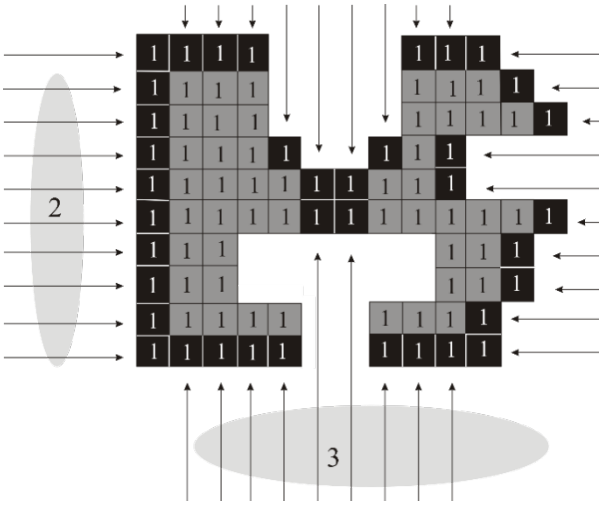


Figure 7. Scheme of image scanning containing three components 1, 2 and 3, and detecting outer contour of component 1.

A pixel is considered to be a contour one if the scanning leads to the first color (label) change of a pixel from background to the color (label) of a connected component. The pixels in the contour detected are shown as black squares in **Figure 7**. It should be noted that detected contour is not totally closed as the part of edge points is left in the shade of the front pixels relating to the direction of scanning. Such simple algorithm, however, allows constructing totally closed component contour if we remove the condition when scanning cycle along the line stops with the first change of scan color from background to the color of a connected component. In this case all shaded pixels will also be labeled. Nevertheless, our task neglects this as we need only the outmost lines in the region of our interest.

2.6. Hough transform and finding maxima

The problem to build a polygon that localizes the detected region by its contour can be solved with the help of Hough transform allowing us to detect segments of straight lines in images[27].

The main idea of Hough transform is the transition from the equation of a straight line in Cartesian coordinates to its equation in ρ and θ parameters. Parameter ρ is the length of a normal to the line drawn from the beginning of coordinates, and θ is the angle between this normal and x-axis (**Figure 8**). Such representation of a straight line doesn't cause infinite parameters as, e.g., in the equation of a straight line being parallel to y-axis in Cartesian coordinates.

Specified parameters ρ_i and θ_j provide the hit of some point with coordinates x_1 and y_1 onto the straight line under consideration:

$$\rho_i = x_1 cos\theta_j + y_1 sin\theta_j.$$

The number of points with coordinates (x_m, y_m), m = 1, 2, …, being present on a straight line (**Figure 6**), is kept in a cell $z_{ij} = z(\rho_i, \theta_j)$ of two-dimensional array called accumulator. The higher the number of these points, the higher the value z_{ij}. Therefore, the detection of k highly bright straight lines passing through relatively large number of points (pixels) on the surface is reduced to the detection of the first k local maxima of two-dimensional function $z(\rho, \theta)$. Exactly this is the essence of the algorithm that detects straight lines on the basis of Hough transform.

Nevertheless, we should note that the more accurate extraction of straight lines is possible in case when the coordinate system connected to the center (x_c, y_c) of each detected component is chosen, apart from laboratory coordinate system being used instead.

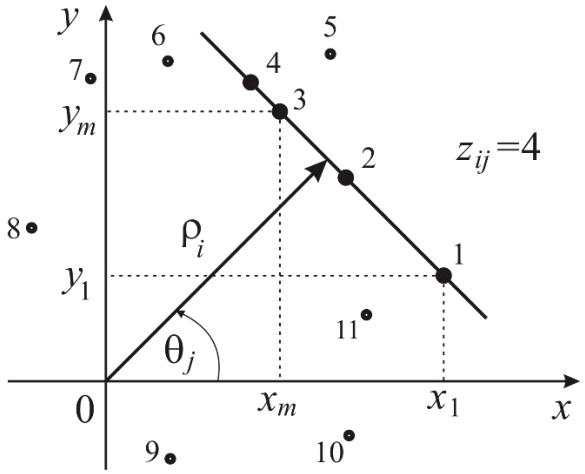


Figure 8. Demonstrating the detection of straight lines in Hough space: 1, 2, …, 11—numbers of points on the surface requiring to check the conditions for their hit on the straight lines with different ρ and θ.

To find the first *k*-local maxima the following algorithm can be recommended (**Figure 9**).

The first maximum corresponding to parameters ρ_{1max} and θ_{1max}, is identified as $z_{1max} = max(z_{ij})$ of accumulator z_{ij}. Then a computing cycle where indicator level z_{level} steps down to 1 starting from z_{1max} up to the presence of *k*-th maximum is organized. The determination of each subsequent maximum after the first one is implemented after the following conditions are checked.

1) The value of accumulator in point (ρ_i, θ_j) exceeds current level z_{level}, i.e.

$$z_{ij} > z_{level},$$

2) Value θ_j does not coincide with angles θ of *p* maxima found before, i.e.

$$\left|\theta_j - \theta_{p\max}\right| > \Delta\theta,$$

where $p = 1, 2, \ldots, k\text{-}1$, $\Delta\theta$—is a given angle range. Such condition can also be called the condition of excluding maxima with closest angles within the range of $\Delta\theta$.

3) Accumulator z_{ij} exceeds all closest neighbors in value, i.e.

$$z_{ij} > z_{i\text{-}1\,j+1},\ z_{i\text{-}1\,j},\ z_{i\text{-}1\,j\text{-}1},\ z_{i\,j+1},\ z_{i\,j\text{-}1},\ z_{i+1\,j},\ z_{i+1\,j+1}.$$

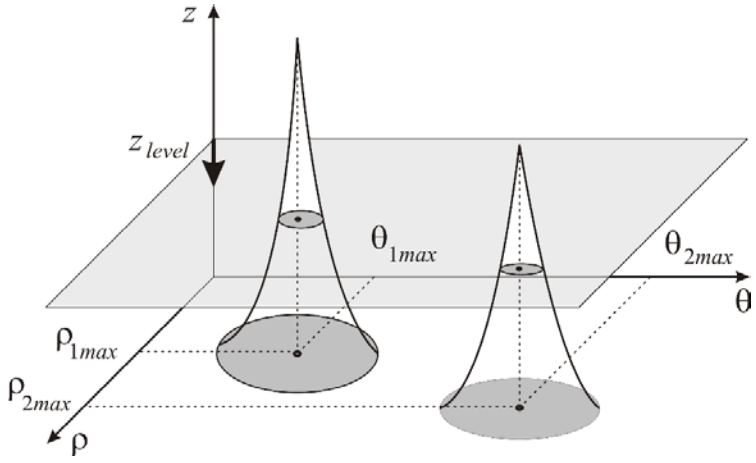


Figure 9. Search of *k* local maxima in Hough space.

The vertices of *k*-gon on the surface (*x*0*y*) are found as intersections of *k* adjacent straight lines identified in Hough space.

$$x_n = \frac{\rho_{n+1\,\max} sin\theta_{n\,\max} - \rho_{n\,\max} sin\theta_{n+1\,\max}}{sin\theta_{n\,max} cos\theta_{n+1\,\max} - cos\theta_{n\,\max} sin\theta_{n+1\,\max}},$$

$$y_n = \frac{\rho_{n\,\max} - x_n cos\theta_{n\,\max}}{sin\theta_{n\,\max}},$$

where n = 1, 2, …, k, here the values with index k + 1 coincide with the values having index 1.

3. Results

The results of applying the algorithm sequence considered are demonstrated with the example of localizing QR-codes as typical representatives of image regions with high level of intensity oscillations limited by closed quadrangles (**Figure 10**). We should mention that quite often practical implementation requires preliminary processing of original images in order to eliminate Gaussian as well as impulse noises. The least distortions when processing are introduced by adaptive methods to suppress image noise component, particularly, when the solution of the tasks under consideration may require switching median filter that eliminates both single pulse and multi-pixel ones[28].

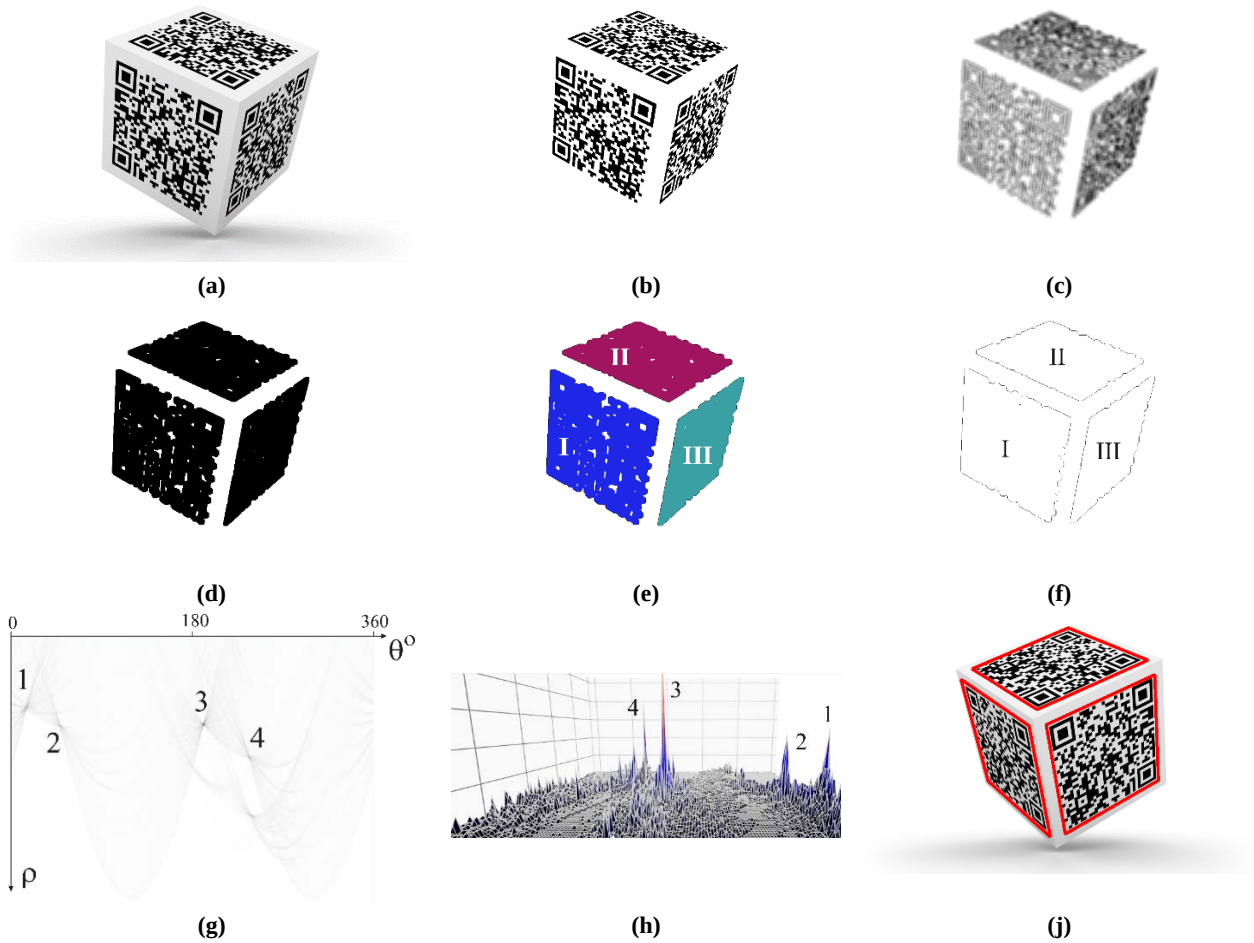
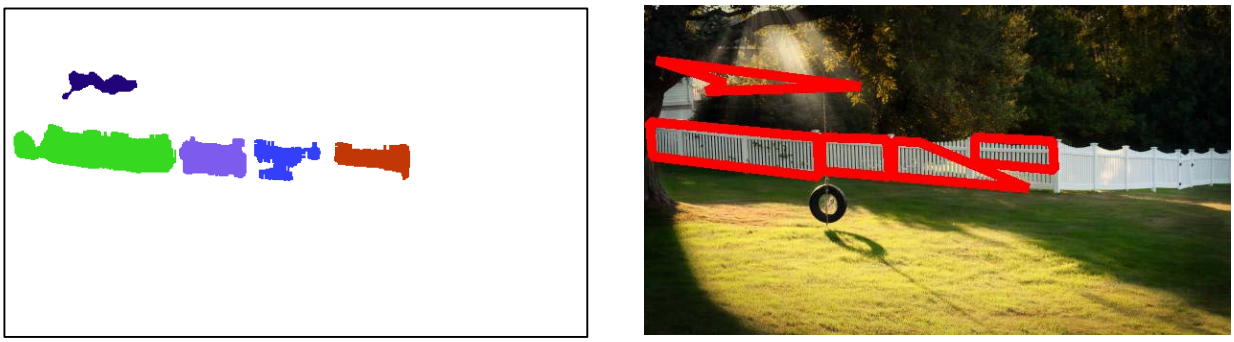


Figure 10. Detection and localization of three QR-codes: **(a)** original image; **(b)** the image after Otsu tresholding; **(c)** intensity oscillation function; **(d)** oscillation function after average level tresholding; **(e)** connected components I, II and III; **(f)** contours of connected components I, II and III; **(g)** I component contour sinogram; **(h)** Hough transform of I component outer contour; **(j)** QR-codes highlighted by red quadrangles.

Figure 10 shows the stages of detection and localization of three QR-codes placed on cube faces (**Figure 10a**): 1st step is Otsu thresholding an original image (**Figure 10b**), 2nd step is calculation of two-dimensional oscillation function (**Figure 10c**), 3rd step is average level thresholding the oscillation function (**Figure 10d**), 4th step is the detection of connected components I, II and III (**Figure 10e**), 5th step is the detection of outer contours of connected components I, II and III (**Figure 10f**), 6th step is Hough transform of each connected component, **Figure 10g**, that shows the dependence (sinogram) of intensity $z_{ij} = z(\rho_i, \theta_j)$ and the four first

maxima numbered in ascending angle θ for the contour of a connected component I, **Figure 10h** where three-dimensional intensity dependence $z_{ij} = z(\rho_i, \theta_j)$ is presented, 7th step is localization of regions I, II and III by quadrangles (**Figure 10j**).

The example in **Figure 10** demonstrates the successful solution of the task to localize regions with high level of intensity oscillations, but the original image used in the example has quite a simple background. **Figure11** is the final stage of proposed algorithm to localize such regions for the image with complex background and uneven illumination (**Figure 4a**). Before quadrangles were built with the help of Hough transform, the connected components in **Figure 6b** were filtered by the condition of their area smallness (filtering result is given in **Figure 11a**). The filtering method can vary depending on a certain situation. In our case, the fragments of a fence are successfully localized in quadrangular regions (**Figure 11b**). However, one region that is highlighted in the uppermost quadrangle turns out not to be the part of a fence. The problem of eliminating spurious regions is solved by adding and configuring the filters in accordance with preset parameters (convexity, infill percentage, angles between sides, etc.).



(a) **(b)**

Figure 11. Connected component localization by quadrangles: **(a)** connected components shown in **Figure 6b**, after filtering of small area components; **(b)** fence fragments localization by quadrangles (see **Figure 3a**) as the regions of an image with high intensity oscillation level.

At the final of this section, we conduct experiment to quantitative evaluate the performance. Since no software to solution the problem of localization of regions of our interest in the general case, we use to compare the widespread 1D and 2D barcode decoders. A comparative analysis of such decoders in terms of their ability to localize graphical barcodes has been carried out many times[8,29], so the task of comparison can be facilitated. Among the many open-source and commercial solutions used, we choose one that shows fairly high and stable performance in all tests—Dynamsoft Barcode Reader SDK[30]. Since no standard dataset is publicly available for evaluation of barcode detection algorithms, we use open-source internet images. The resolution of the images scales from 0.3 MPix to 5 MPix. An image contains at least one 2D barcode; the maximal number of codes in a scene was 13. The total number of barcodes on the all images is 47. The percentual image coverage of the codes varies from 1% to 70%. The position and size of the codes are not repeated for different images. Rotation, perspective deformation, lighting conditions, blur of barcodes and complicity of background additionally vary in each image. **Table 3** compares the performance of the two solutions in terms of the TP (True Positive), TN (True Negative), FP (False Positive), FN (False Negative), Precision = TP/(TP + FP), and Recall = TP/(TP + FN)[31]. TN is the number of barcodes, more than 20% of the area of which are incorrectly displayed in the image. FP is the number of localized image fragments that are not barcodes.

Table 3. Test results.

Method	TP	TN	FP	FN	Precision	Recall
Dynamsoft	22	11	0	14	1	0.61
Our	33	11	2	3	0.94	0.92

4. Discussion

Table 3 clearly shows that our solution achieves the highest TP, the best Recall and the lowest FN. Parameters FP and Precision have approximately the same value as Dynamsoft Barcode Reader SDK ones. That is, our solution achieves the best performance in successfully localizing the barcode regions in an image.

In general, the obtained results indicate that the problem of fragment localization is solved with a high degree of results reliability using the considered sequence of algorithms, i.e., the main goal of the study has been achieved. It can be predicted that the considered approach to object selection using the proposed oscillation function will be in demand in solving a wide range of problems where image analysis is required.

5. Conclusion

The work offers the sequence of algorithms to detect and localize images fragments being different in high frequency spatial intensity oscillations. A new procedure to analyze images called oscillation function intended for detecting and extraction of intensity oscillation regions is recommended to be used in practice. A simple algorithm to extract contours of connected components as the first step in the process of building *k*-gons on the basis of Hough transform is proposed. The algorithm to find *k*-first local maxima of Hough transform is offered and studied. Test examples demonstrate successful localization of image fragments applying the approach considered. High computational rate is provided by wide application of integral sum image. The considered pattern recognition approach, based on the application of the proposed oscillation function, can be used as an independent method for solving the tasks set, as well as a means of enhancing the confidence and correctness of decision making when used in conjunction with alternative methods of localizing areas of interest, in particular salient objects.

Author contributions

Conceptualization, AT and MS; methodology, AT; software, AT; validation, AT, MS and AS; formal analysis, AT; investigation, MS; resources, MS; data curation, AS; writing—original draft preparation, AT; writing—review and editing, MS; visualization, AS; supervision, AT; project administration, MS; funding acquisition, MS.

Conflict of interest

The authors declare no conflict of interest.

References

1. El Hachem C, Santiago R, Painvin L, et al. Brick orientation adjustment in the automotive industry using image processing techniques. In: Proceedings of 2022 8th International Conference on Control, Decision and Information Technologies (CoDIT); 17–20 May 2022; Istanbul, Turkey. pp. 729–733.
2. Dougherty G. *Digital Image Processing for Medical Applications*. Cambridge University Press; 2009. pp. 1–459.
3. Shi W. The application of image processing in the criminal investigation. In: Proceedings of the 2016 4th International Conference on Machinery, Materials and Information Technology Applications; 10–11 December 2016; Xi'an, China. pp. 139–142.
4. Schwering PBW, Kemp RAW, Schutte K. Image enhancement technology research for army applications. *Proceedings of SPIE* 2013; 8706: 2–11. doi: 10.1117/12.2017855
5. Fan L. Image processing algorithm of Hartmann method aberration automatic measurement system with tensor product model. *Journal on Image and Video Processing* 2019; 2019: 43. doi: 10.1186/S13640-019-0440-9
6. Asht S, Dass R. Pattern recognition techniques: A review. *International Journal of Computer Science and Telecommunications* 2012; 3(8): 25–29.
7. Singh C. Machine learning in pattern recognition. *European Journal of Engineering and Technology Research* 2023; 8(2): 63–68. doi: 10.24018/ejeng.2023.8.2.3025
8. Karrach L, Pivarciova E. Recognition of data matrix codes in images and their applications in production processes. *Management Systems in Production Engineering* 2020; 28(3): 154–161. doi: 10.2478/mspe-2020-0023

9. Liu T, Yuan Z, Sun J, et al. Learning to detect a salient object. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2011; 33(2): 353–367. doi: 10.1109/TPAMI.2010.70
10. Chen C, Wei J, Peng C, et al. Improved saliency detection in RGB-D images using two-phase depth estimation and selective deep fusion. *IEEE Transactions on Image Processing* 2020; 29: 4296–4307. doi: 10.1109/TIP.2020.2968250
11. Cheng MM, Mitra NJ, Huang X, et al. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2015; 37(3): 569–582. doi: 10.1109/TPAMI.2014.2345401
12. Viola P, Jones MJ. Robust real-time face detection. *International Journal of Computer Vision* 2004; 57(2): 137–154. doi: 10.1023/B:VISI.0000013087.49260.fb
13. Otsu N. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics* 1979; 9(1): 62–66.
14. Bradley D, Roth G. Adaptive thresholding using the integral image. *Journal of Graphics Tools* 2007; 12(2): 13–21. doi: 10.1080/2151237X.2007.10129236
15. Niblack W. *An Introduction to Digital Image Processing*. Prentice Hall; 1986. pp. 1–215.
16. Sauvola J, Pietikainen M. Adaptive document image binarization. *Pattern Recognition* 2000; 33(2): 225–236. doi: 10.1016/S0031-3203(99)00055-2
17. Koleda P, Hrčková M. Global and local thresholding techniques for sawdust analysis. *Acta Facultatis Technicae* 2018; XXIII(1): 33–42.
18. Burger W, Burge MJ. *Digital Image Processing: An Algorithmic Introduction Using Java*. Springer-Verlag London Ltd.; 2016. pp. 1–811.
19. Fang L, Xie C. 1-D barcode localization in complex background. In: Proceedings of 2010 International Conference on Computational Intelligence and Software Engineering; 10–12 December 2010; Wuhan, China. pp. 1–3.
20. Santosa F, Goh M. Bar code decoding in a camera-based scanner: Analysis and algorithm. *SIAM Journal on Imaging Sciences* 2022; 15(3): 1017–1040. doi: 10.1137/21M1449658
21. Shapiro LG, Stockman GC. *Computer Vision*, 1st ed. Pearson; 2001. pp. 1–580.
22. Gonzalez R, Rafael R. *Digital Image Processing*. Pearson; 2018. pp. 1–1168.
23. Canny J. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1986; PAMI-8(6): 679–698. doi: 10.1109/TPAMI.1986.4767851
24. Bradski G, Kaehler A. *Learning Opencv: Computer Vision with the Opencv Library*. O' Reilly Media; 2008. pp. 1–555.
25. Jähne B, Scharr H, Körkel S. Principles of filter design. In: Jähne B, Haussecker H, Geissler P (editors). *Handbook of Computer Vision and Applications*. Cambridge: Academic Press; 1999. pp. 125–151.
26. Papari G, Petkov N. Edge and line oriented contour detection: State of the art. *Image and Vision Computing* 2011; 29(2–3): 79–103. doi: 10.1016/j.imavis.2010.08.009
27. Hough PVC. Method and Means for Recognizing Complex Patterns. U.S. Atomic Energy Commission No. 3069654, 18 December 1962.
28. Trubitsyn A, Grachev EY. Switching median filter for suppressing multi-pixel impulse noise. *Computer Optics* 2021; 45(4): 580–588. doi: 10.18287/2412-6179-CO-841
29. Lin JA, Fuh CS. 2D barcode image decoding. *Mathematical Problems in Engineering* 2013; 2013: 848276. doi: 10.1155/2013/848276
30. Dynamsoft. Barcode reader SDK. Available online: www.dynamsoft.com/Products/Dynamic-BarcodeReader.aspx (accessed on 25 May 2023).
31. Mikolajczyk K, Schmid C. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2005; 27(10): 1615–1630. doi: 10.1109/TPAMI.2005.188