Operators for Edge Detection in an Image Based on Technologies of Cellular Automata

Stepan Bilan

Taras Shevchenko National University of Kyiv, Volodymyrska Street, 60, Kyiv, 01033, Ukraine

Abstract

The paper considers the operators for detecting characteristic features on complex color images. Such operators are used to highlight characteristic features in the image, from which a vector of quantitative values is formed for further recognition. The existing operators of Roberts, Sobel and Prewitt are considered, as well as on the basis of cellular automata using the averaging value in the neighborhood of cells. Edge selection operators based on cellular automata technologies are proposed. The color image is divided into binary layers, to each of which local transition functions are applied, allowing to select edge cells. Based on the results obtained, an updated image is formed, on which the main information elements are highlighted, which display the edges of the brightness and color differences. The application of edge selection operators based on the von Neumann and Moore neighborhoods was studied. Differences in the results of applying the operators used are presented. The results showed that the use of edge selection operators allows you to preserve color and highlight the most informative edges in the image. The use of thresholding allows you to select only information elements that form color and brightness differences between image elements.

Keywords [1]

Edge detection operator, cellular automata, local transition function, color image, binary layer, cell neighborhood

1. Introduction

Modern computer vision systems are characterized by the use of a wide range of image processing and recognition methods. [1-5]. Preliminary image processing is aimed at preparing and converting real images received at the input of the system in order to extract the specified characteristic features. From the quantitative characteristics of the selected characteristic features, a vector of features is formed, which enters the decision block. The decision block, based on the existing base of reference vectors, decides whether the image belongs to a particular class. A very important task for building computer vision systems is the task of forming an optimal set of characteristic features. The structure of the system and the method, which is aimed at their selection and determination of quantitative characteristics, depends on the set of characteristic features. The structure of the vector of characteristic features (the sequence of numbers) depends on the vision system and image processing and recognition methods. In fact, for different images (text, geometric figure, human face image, fingerprint, etc.), different sets of characteristic features and different methods are used to form a vector of characteristic features that fully describes the analyzed image. Almost all modern computer vision systems use the edge detection operation in the input image preprocessing [5-7]. The edges carry the most information about the image and often completely describe its geometric structure, which makes it possible to recognize the image with high accuracy. However, different methods allow you to select a different number of pixels belonging to the edges and do not always give the desired result [5-7]. Almost all existing edge detection operators require additional thresholding. This is especially true for complex color images with a large number of brightness changes.

Information Technology and Implementation (IT&I-2022), November 30 - December 02, 2022, Kyiv, Ukraine
EMAIL:bstepan@ukr.net
ORCID: 0000-0002-2978-5556

2. Statement of the Problem

In this paper, we solve the problem of using cellular technologies to detect edges in a color image, which reduces the time spent on the implementation of the operator, and also does not use complex mathematical calculations inherent in known edge detection operators. The problem is solved by splitting a raster image into binary slices, which form a set of binary images and to which the rules of two-dimensional cellular automata (CA) are applied to detect edge pixels.

3. Relative Works

Operators for detecting edges in an image are described in many literature sources [1, 5 - 7]. The various operators for selecting edges on the image are described in particular detail in the works [5, 6]. In [5], edge detection operators of the first and second orders are considered, and their detailed comparison is also carried out. Specific examples show the positive aspects of some operators and the disadvantages of other operators. Almost all edge selection operators use a mask of coefficients of different dimensions. The most commonly used coefficient masks are 2×2 and 3×3. In this case, the masks contain different values of the coefficients, both positive and negative. Masks are applied for each pixel of the image, and the resulting value is calculated for each pixel. One of the earliest such operators is the Roberts operator [8], which uses 2 × 2 coefficient masks. However, the Roberts operator is characterized by the fact that the edges are displayed as double selected cells. The Sobel [9] and Prewitt [10, 11] operators use 3×3 coefficient masks. At the same time, they highlight the edges in dark colors. These operators are quite sensitive to brightness changes and can highlight a large number of pixels in complex color images. Second-order edge detection operators are more complex than first-order operators. Such operators include, for example, the Laplacian operator [5, 12] and the Marr-Hildreth operator [5, 13]. These operators use more complex calculations, and, accordingly, are implemented by algorithms that are more complex. Compared to first-order operators, second-order operators select fewer pixels, since they pay more attention to the analysis of image pixel properties.

Another approach to edge detection in an image is to use CA technologies, which give good results for binary images. [14]. The papers [14, 15] consider the use of CA technologies for edge detection in color and grayscale images. To do this, a neighborhood is selected and the average value in the control cell is determined, which is compared with a pre-selected threshold value. After thresholding, pixels belonging to the edges remain on the image. Different number of pixels for different thresholds are allocated. In fact, this technology uses a mask, the shape of which corresponds to the shape of the selected neighborhood of cells, and also uses the calculation of the average value for each cell.

In practically almost all described operators, after their application, do not make it possible to determine the chromaticity of pixels inside the edges. In practically almost all of them use quantitative values obtained as a result of implementing a certain formula for each pixel, which requires many passes of all image pixels to implement the operator. The time it takes to implement the operators depends on the time it takes to analyze the states of the neighborhood cells and on the implementation of the computational algorithm for each pixel. The execution time of the computational algorithm also depends on the dimension of the mask, which is used to implement the corresponding edge detection operator. To solve this problem, this paper uses CA technologies that simplify calculations and implement the operation of detecting edges in one cycle of scanning the image field for computer systems. The most successful use of CA is provided by their hardware implementation. A single-bit memory element and a control combination circuit represent each cell. For hardware implementation, edge detection is carried out in one cycle, since each binary layer is implemented by hardware, and all cells perform calculations in parallel and transmit signals to the cells of each subsequent layer also in parallel. In such vision systems, scanning of the entire image field is not required, one binary layer performs operations in one cycle, which depends on the response time of the combinational circuit and the time it takes to switch one memory element (trigger).

4. Implementation of Transition Rules in CA Cells for Detection of Edge Pixels

To implement the edge detection operator based on CA technologies, the initial color image is represented in RGB coding, which is currently the most promising (among all existing ones) in their

representation in computer vision systems. Each pixel is encoded with 24 bits, which are divided into three bytes. In this sequence, from left to right, bytes are arranged, which, respectively, encode blue, green and red colors. In this encoding, the properties of each pixel can be represented from zero (black) to 16777215 (white). Since each pixel in this encoding is represented by 24 bits, and all image pixels represent a matrix of binary codes of a certain size, the corresponding weight bits in the codes of each pixel can be separated into a separate bit weight matrix (binary layer). In this case, in RGB encoding, the image can be divided into 24 binary layers. Grayscale images are represented by 8 bit code for each pixel. Grayscale images can be split into 8 binary layers (8 binary images). In the early stages of development, images with a color encoding depth of 4 bits were used. Figure 1 shows an example of splitting a code fragment represented by four bits into four binary layers.

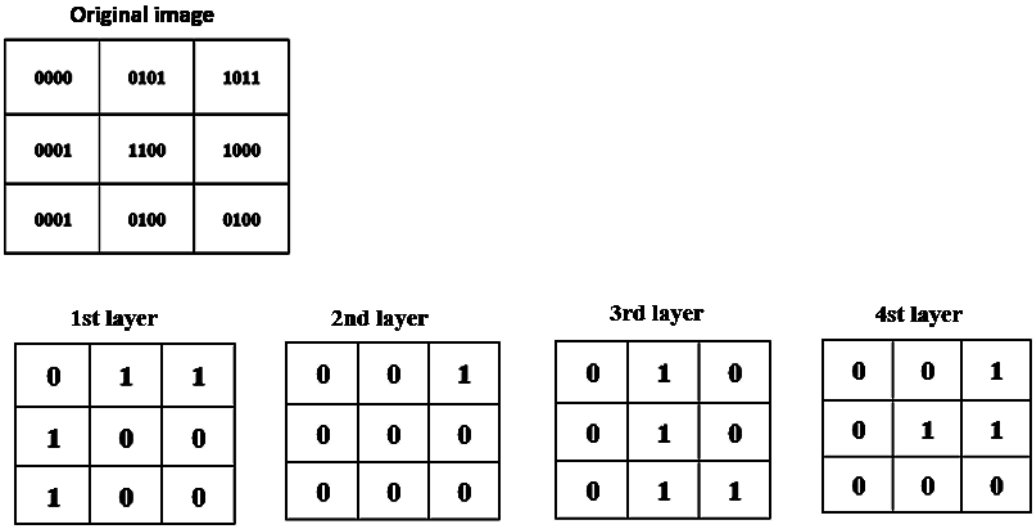

Figure 1: An example of splitting an image into binary layers

The image fragment by 3×3 pixel on Figure 1 is presented. Each pixel is encoded with a 4-bit code. With 4-bit encoding, 2^4 colors of an image can be represented. The least significant bits of the code of each pixel form the first layer (in Figure 1 this layer is represented by a matrix of 3 × 3 pixels on the left in the bottom row), each pixel of which displays the value of the least significant bit from the pixel codes of the original image. Similarly, from left to right, the remaining layers and the corresponding bit values are represented for each pixel. Figure 1 shows four binary layers, which can be considered as cellular automata with different neighborhood shapes. Each such binary layer and each cell (pixel) in it can perform a local transition function, according to the given rules. For two-dimensional CA, a different set of rules can be applied, the maximum number of which depends on the shape of the neighborhood. For example, if the von Neumann neighborhood is used (the neighborhood is four cells), then 65536 rules can be applied, and if we also take into account our own control cell (a total of five cells), then 4294967296 rules can be applied. This is a large number, which shows the impossibility of investigating all local transition functions at the present time. However, you can use some rules that give the desired result for solving a particular problem. Thus, in [16], the rules 1000, 7000 and 55555 for the von Neumann neighborhood are considered, which allow obtaining various color images from the initial binary forms. Rule 1000 allows you to implement an image shift to the right. In this case, the image at each time step is formed by codes obtained in the evolution of the initial binary layer (CA). It does not use the transformation of each current layer of a previously generated image. In [17], rules 43690, 65280, 52428, and 61680 were implemented and studied for the von Neumann neighborhood. These rules implement the image shift down, right, left and down, and combinations of these rules allow you to build any movement trajectory. At the initial moment, a binary image was formed on the initial (zero) binary layer. As a result of each subsequent time evolutionary step, each subsequent binary layer was formed, the bits of which formed each subsequent bit for each pixel. After 24 time steps of evolution, a color image was formed that showed the shift of the original image or other operations.

However, these rules were considered as the evolution of two-dimensional CA, and the results of evolution were considered as a color image. Practically no studies were considered that were based on the application of such rules to the transformation of already formed color images. The transformation

of each binary layer based on CA technologies gives different results, which allow for better image pre-processing compared to other methods.

5. Implementation of the Edge Detection Operator on the Image Based on the Rules of Two-Dimensional CA

To avoid various calculations to determine the quantitative value of each pixel, logical transformations are used in the work, which are implemented using local transition functions in two-dimensional CA. Such local transition functions are used for binary images for different types of neighborhoods [14]. For color images, neighborhood mean calculations and additional thresholding were used. The full use of CA technologies is based on splitting a color image into binary layers and the consistent application of local transition functions (rules) for two-dimensional CA. To describe the rules for two-dimensional CA, the coding shown in Fig. 2 is used.

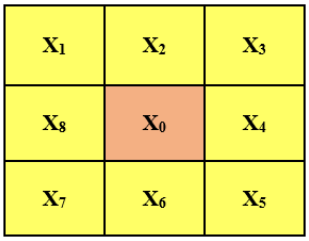

Figure 2: Encoding neighborhood cells for a cell X_0

With this encoding of the nearest cells, for the von Neumann neighborhood, the rule 715827882 is used, which is described by the following table 1. This rule allows you to select cells that form the contours of a binary image. If all cells (X_2, X_4, X_6, X_8) that form the von Neumann neighborhood of the analyzed cell (X_0) have the state of logical "1", then the state of the analyzed cell (X_0) goes into the state of logical "0". If at least one of the cells in the neighborhood has a state of logical "0", then the analyzed cell remains in the same state. Thus, in each binary layer (CA), single cells are selected that form the edges of the binary image.

Table 1
Rule coding table 715827882

Binary set number	State of neighborhood cells at time t					The value of the cell $X_0(t+1)$ at the next time (t+1) according to the rule 715827882
	$X_8(t)$	$X_6(t)$	$X_4(t)$	$X_2(t)$	$X_0(t)$	
0	0	0	0	0	0	0
1	0	0	0	0	1	1
2	0	0	0	1	0	0

28	1	1	1	0	0	0
29	1	1	1	0	1	1
30	1	1	1	1	0	0
31	1	1	1	1	1	0

To select a rule, it is important to place the value of the control cell in the code that is used. This local transition function must use an additional control bit, since its value at the next time step may depend on the state of the control cell (X_0) at the previous time. A similar situation arises when using the Moore neighborhood where the rules are represented by a large number of numbers (up to 15). It is better to describe such rules by a system of logical functions, as presented in [13].

$$b(t+1) = \begin{cases} b(t), \ if \ \ X_2(t) \wedge X_4(t) \wedge X_6(t) \wedge X_8(t) = 0 \\ 0, \ if \ \ X_2(t) \wedge X_4(t) \wedge X_6(t) \wedge X_8(t) = 1 \end{cases} \quad (1)$$

$$b(t+1) = \begin{cases} b(t), \ if \ \ X_1(t) \wedge X_2(t) \wedge \ldots \wedge X_8(t) = 0 \\ 0, \ if \ \ X_1(t) \wedge X_2(t) \wedge \ldots \wedge X_8(t) = 1 \end{cases} \quad (2)$$

where $X_1(t)$, $X_2(t)$, $X_3(t)$, …., $X_8(t)$ - signals at the outputs of neighboring cells in accordance with Figure 2 at time t; $b(t)$ - state of the cell at time t.

Both rules change the state of the cell only in one case, when all the cells of its neighborhood and the control cell itself belong to the image (corresponding to the logical "1"), and not to the background. An example of the application of such rules in Fig. 3 is shown.

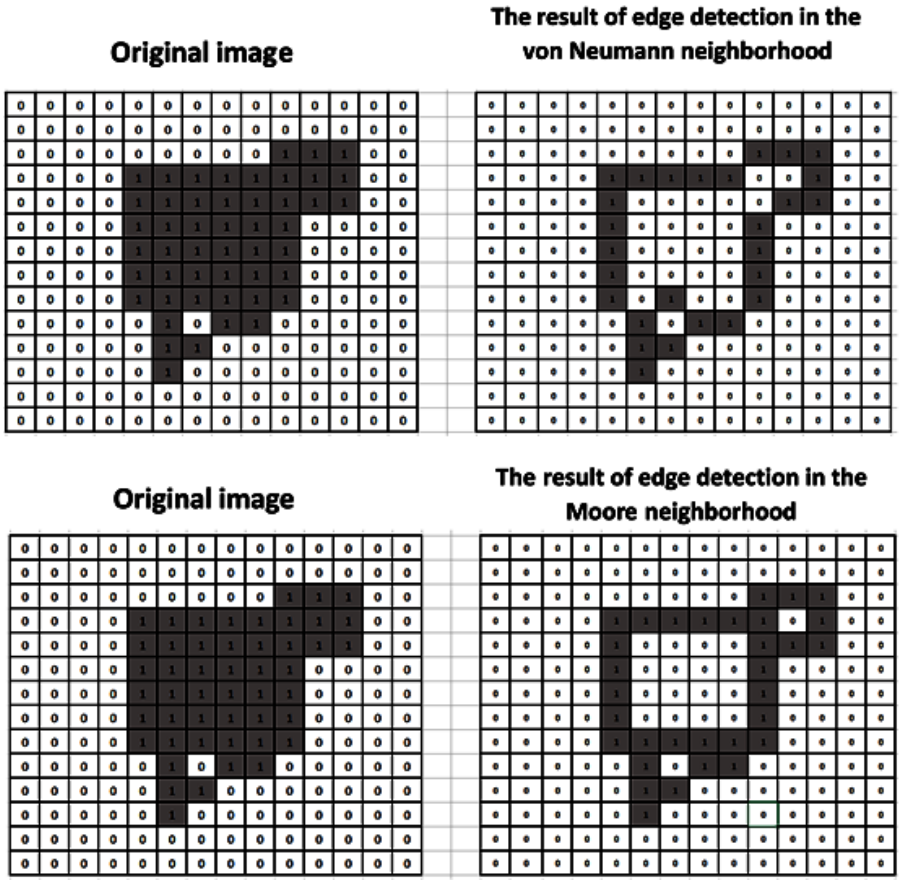

Figure 3: An example of applying the rules for selecting edge cells in two-dimensional CA

Figure 3 shows the differences in the results of applying different neighborhoods, which generally affects the further analysis of images. Applying the Moore neighborhood allows you to select more cells than after applying the von Neumann neighborhood. Highlighted edges in these cases may describe shapes that may differ in certain areas. The detected edges are thicker after applying the Moore neighborhood, and additional contours appear as not all neighboring (diagonal) cells have a logical "1" state. The more cells form the neighborhood, the more edge cells stand out in the binary image.

In this work, such rules are applied to all binary layers of a color image. In each binary layer, there are cells that already belong to the binary image (cell have the value of logical "1") and which change their state at the next time step of evolution (transition to the state of logical "1"). The cells that form the edges of the binary image in each binary layer remain in the logical "1" state. The binary image in each layer is not a binary version of the original image. The resulting binary layers at the next step of evolution form bits of codes for each pixel of a color image. In this case, the results of applying local transition functions are stored in the same layer and do not affect the next binary layer. In this case, the layer that performs the local transition function is transformed. Each binary layer is transformed according to its initial states. An example of applying the described rules for a color image in fig. 4 is shown. In this example (Fig. 4), the edge pixels are represented by a different color that distinguishes them from all other pixels in the image. If the edges in the original image are blurred, then the number of edge pixels stands out more. Moreover, the selected edges are thicker and can be represented by pixels of different colors. In fact, pixels of one color describe the edges of the image of a figure of one color that border the background pixels, and pixels of another color belong to the background pixels. In this case, you can reduce the thickness of the detected edges by selecting pixels of the same color

from all selected pixels, and reset the rest to zero. On Figure 4 also a fragment of the matrix that encodes the properties of the pixels is shows. On this matrix, pixel codes are presented in decimal notation. Binary encoding allows you to split the image into binary layers.

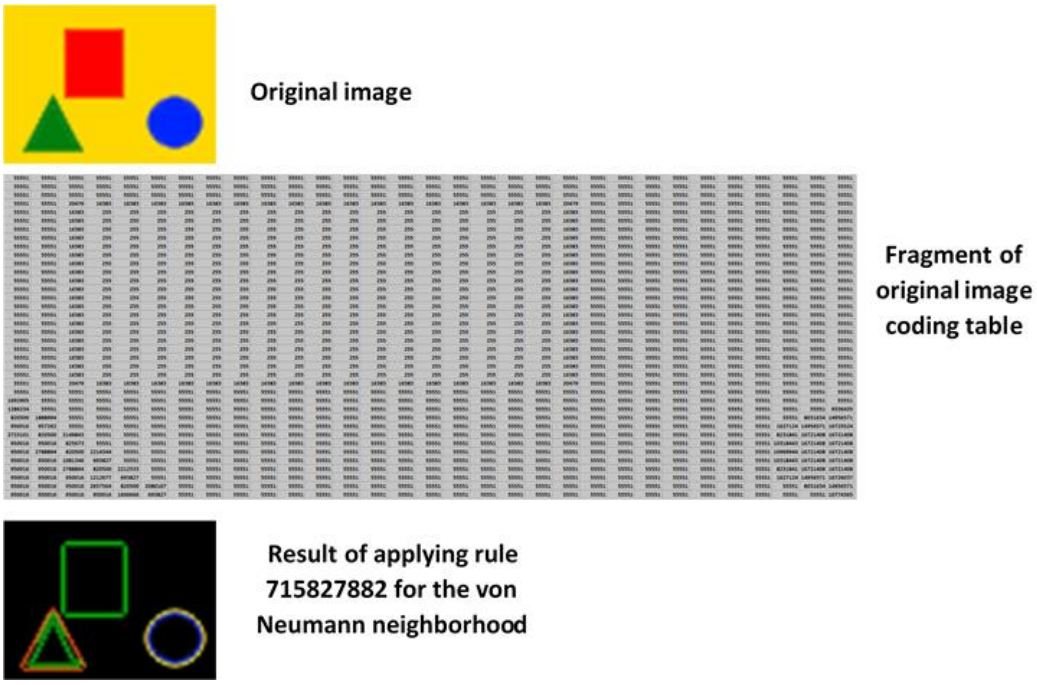

Figure 4: An example of applying the rule 715827882 for the von Neumann neighborhood, as well as an example of the encoding matrix for the original color image

Figure 5 shows the binary layers (5, 12, 21 and 24) after applying the local transition function. The fifth layer refers to the first byte, which encodes the red color and its shades, the twelfth layer refers to the second byte, which encodes the green color and its shades, and the 21st and 24th binary layers refer to the third byte, which encodes the blue color and its shades. Dark pixels encode the state of the logical "1". Different colors of selected pixels indicate the depth of color and brightness properties, as well as the predominance of bits and color bytes in the image. Binary slices (Figure 5) indicate the color byte that is involved in the formation of the local object. On Figure 4 the red rectangle is determined by the binary layers that form the byte of the red code of the pixel, since it is responsible for the formation of the red color. However, the first byte of the rectangle's pixels is merged with the bytes of the edge pixels, and so they go into a logical "0" state. The bits of the second byte of the boundary pixels remain in the single state, which form the green color. As a result, selected pixels display green color.

Layers also work on their own for images of objects of a different color and background color. All binary layers are arranged in parallel in a given order, and cells located at the same depth line form a binary code for each pixel. Accordingly, 24 binary layers are used, which form N×M 24 bit binary codes (where the dimensions of N×M correspond to the dimension of the image).

The priorities of using the edge detection method on the image depend on the task in the processing of color images. If one of the tasks is to reduce the time spent on the implementation of the method, as well as to simplify it, then the proposed operator is the most acceptable. At the same time, the information content of the method is not inferior to already existing methods. On Figure 6 shows the results of the selection of edge pixels for the proposed method, as well as for the Roberts, Sobel and Prewitt operators.On Figure 6 are shows the results of applying the edge detection operator based on CA technologies using the von Neumann and Moore neighborhoods. The visual difference of such application of different neighborhoods is obvious. The difference with other operators is also obvious. Different colors obtained from the application of the Roberts, Sobel and Prewitt operators are due to different coefficient matrices that implement the corresponding operators. At the same time, threshold processing was not applied after their application. Thresholding for the Roberts, Sobel, and Prewitt operators in detail in [5, 15, 18] is described. The quality of the obtained images for all operators is affected by the value of the specified threshold. Thresholding can also be applied to the considered CA-

based operators. The application of thresholding converts the resulting image to a binary form. However, it is possible to save the color for those pixels that exceed the threshold value. For this option, it is not necessary to convert the pixel color to only two types (white or black), but the initial state (color) should be preserved if the threshold value is exceeded.

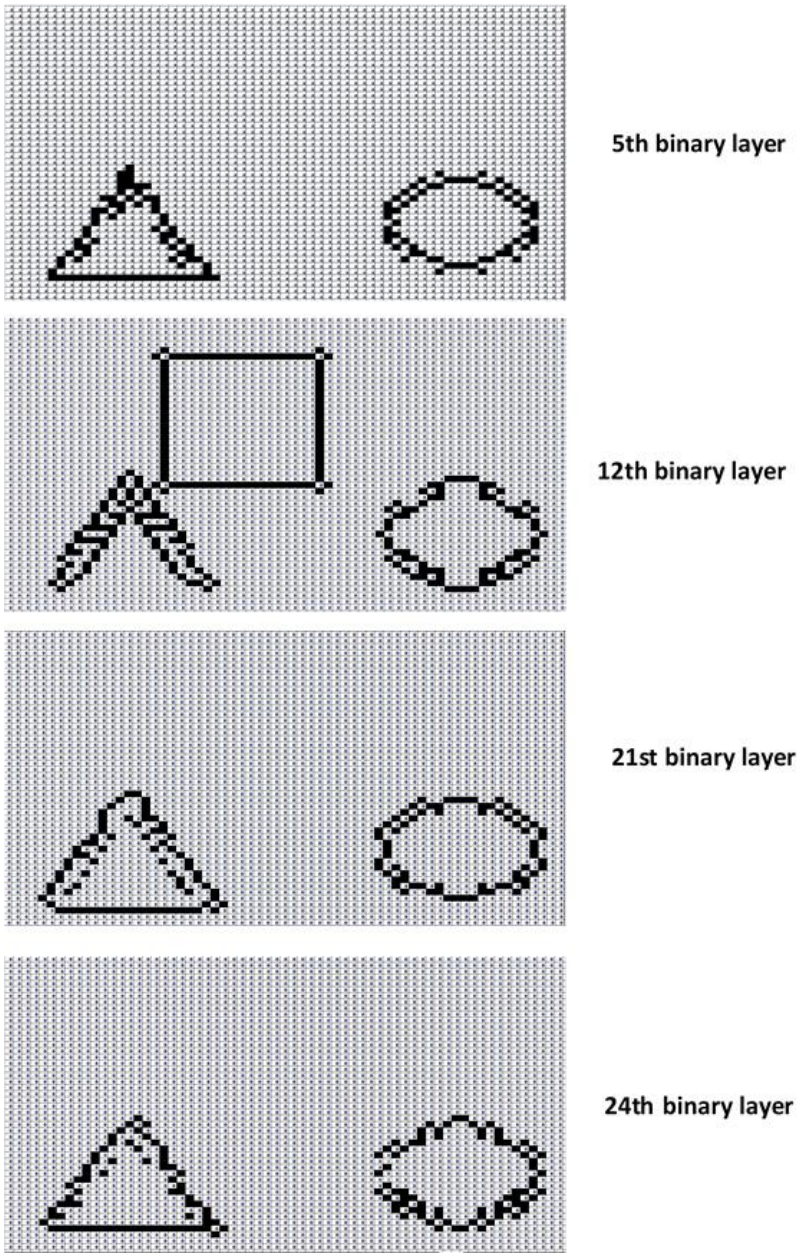

Figure 5: An example of splitting the original image into layers (layers 5, 12, 21, 24 are shown) and the results of applying the rule for the presented binary layers for the image shown in Figure 4

On Figure 7 presents the results of thresholding for the edge detection operator based on the CA with the Moore neighborhood. Figure 7 shows the thresholding results for the example edge detection operator shown in Figure 5 using the Moore neighborhood. Cells were detected, the numerical values of the codes of which were greater than the threshold value. Such cells took the value 16777215 (white). Analysis of the results of thresholding allows you to determine the required threshold value, which gives the optimal number of selected pixels and does not reduce information content. Visual analysis showed that the most acceptable threshold value is 500000. In the resulting image, the smallest number of white pixels belonging to the edges are detected. For this option, the edge pixels clearly define the original image. For different edge selection operators, different quantitative threshold values are used.

Such threshold values are determined by the formulas used and the matrices of the coefficients used. In this case, the most acceptable threshold values are determined experimentally.

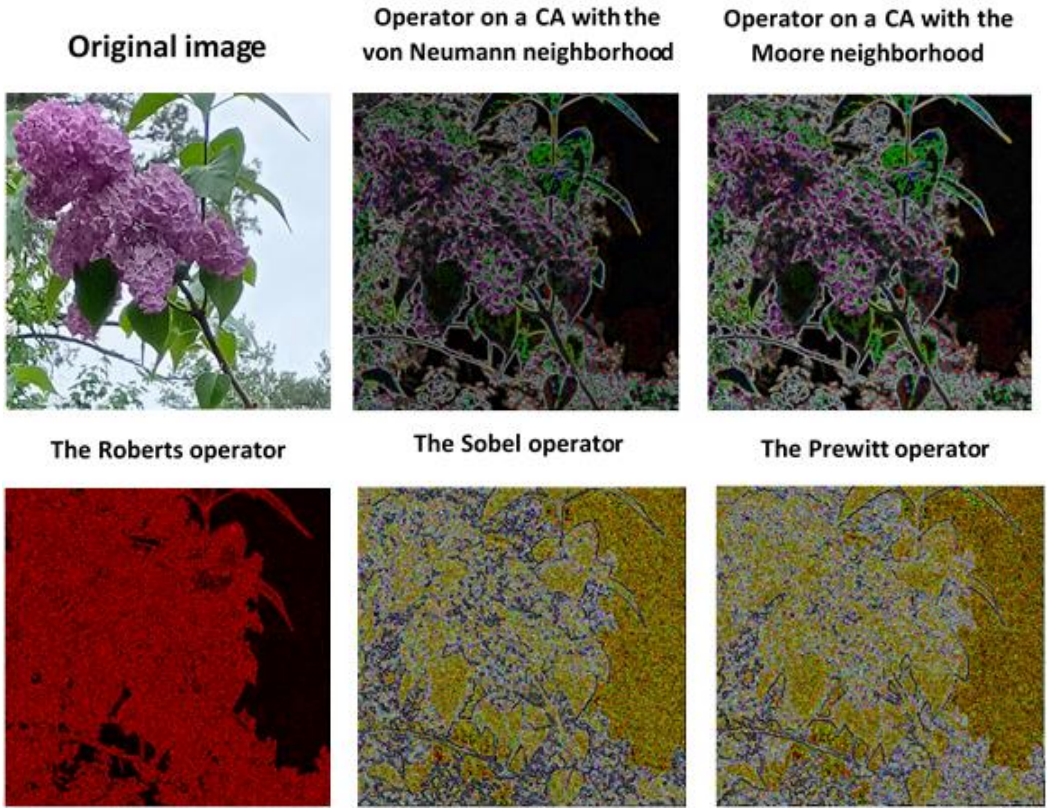

Figure 6: Results of applying the proposed edge detection operator and the Roberts, Sobel and Prewitt operators

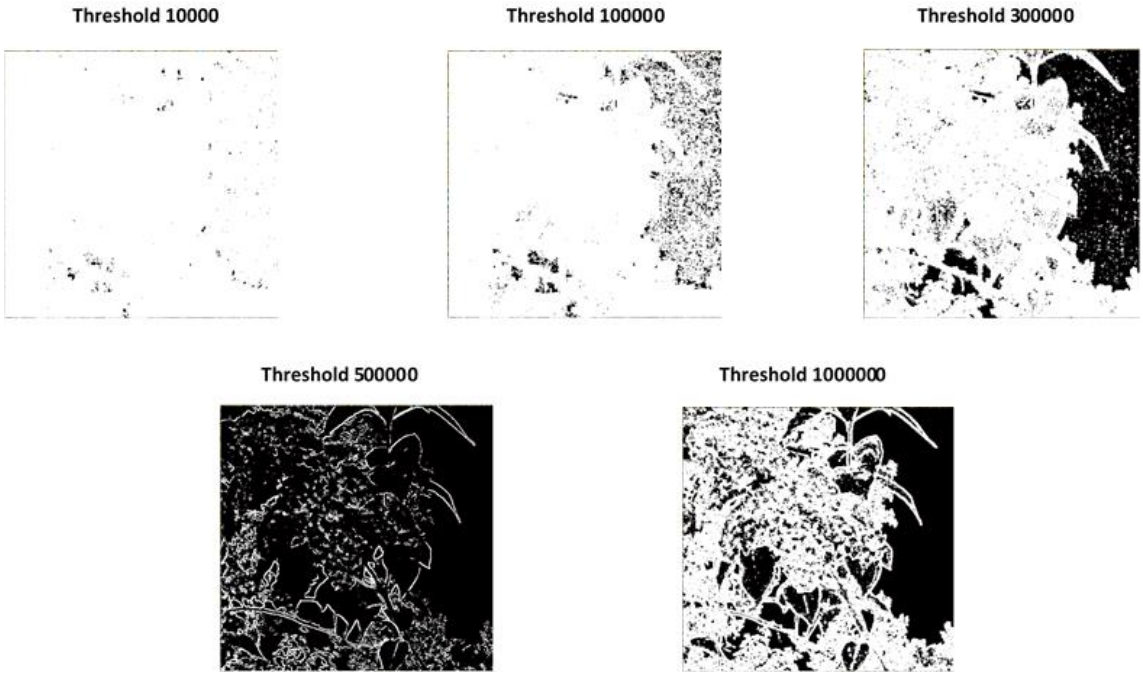

Figure 7: Thresholding results for the edge detection operator based on the CA with the Moore neighborhood

In hardware implementation, the execution time of the edge selection operation depends on the trigger switching time in one cell. Since the cell is implemented on a simple combinational circuit that implements formulas (1) and (2), all cells of binary layers perform such an operation in parallel.

Software implementation complicates the execution of the described statement. First of all, it depends on the characteristics of the computing system, and is also limited by the form of representation of images in a computer system. In this approach, the implementation time of the operator depends on the time of scanning the image and analyzing the codes of all pixels, which is determined by the execution time of the algorithm implemented programmatically in the chosen programming language.

6. Conclusion

The paper considers and investigates edge detection operators in an image based on cellular automata technologies. A method for implementing such operators that do not require complex calculations and do not use the implementation of mathematical models using masks of coefficients of various dimensions is described. This approach greatly simplifies the implementation of the edge selection operator and reduces the time to one time cycle. It is shown that for different shapes of neighborhoods, the results have differences in the thickness of the selected edge pixels. Using two-dimensional cellular automata technology preserves color. Although the colors are slightly changed in the converted image, the result allows you to determine the colors of the pixels in the original image. Studies have shown the possibility of applying thresholding, which allows you to save information while maintaining a smaller number of selected pixels compared to the initial application of the operator. In further research, the author plans to use various forms of neighborhoods, as well as other rules for two-dimensional cellular automata.

7. References

[1] Szeliski, R. Computer Vision: Algorithms and Applications, - Springer, (2011): 832.
[2] Rosin, P. L. Yu-Kun Lai, Ling Shao, Yonghuai Liu. RGB-D Image Analysis and Processing (Advances in Computer Vision and Pattern Recognition). – Springer, (2019): 953.
[3] Gonzalez R.C. Woods R.E. Digital Image Processing. 3rd ed. - Pearson, (2007): 976.
[4] Pratt W.K. 2016. Digital Images Processing. Third edition. Wiley, (2016): 738.
[5] Nixon, M. S. Aguado, A. S. Feature Extraction and Image Processing. – Newnes, (2002): 350.
[6] Parker J.R. Algorithms for Image Processing and Computer Vision. Second Edition. - Wiley Publishing, Inc., (2010): 504.
[7] Koschan, A. and Abidi, M. Detection and Classification of Edges in Color Images. IEEE Signal processing magazine, January (2005): 64-73.
[8] L. Roberts. Machine Perception of Three-Dimensional Solids, Optical and ElectroOptical Information Processing, MIT Press, (1965): 159-197.
[9] Sobel, I.E. Camera Models and Machine Perception, PhD Thesis, Stanford Univ, (1970): 60.
[10] Prewitt, J.M.S. "Object Enhancement and Extraction". Picture processing and Psychopictorics. Academic Press. (1970): 75-149.
[11] Prewitt, J. M. S. and Mendelsohn, M. L. The Analysis of Cell Images, - Ann. N.Y. Acad. Sci., 128, (1966): 1035–1053.
[12] Waheed, W. Deng, G. Liu, B. Discrete Laplacian operator and its applications in signal processin, IEEE Access, VOLUME 4, (2016): 1-17.
[13] Marr, D. Hildreth, E.. "Theory of Edge Detection". Proceedings of the Royal Society of London. Series B, Biological Sciences. 207 (1167), (1980): 187–217.
[14] Bilan, S. Models and hardware implementation of methods of Pre-processing Images based on the Cellular Automata, Advances in Image and Video Processing, Vol 2, No 5 (2014): 76-90
[15] Bilan, S. Riabtsev, V. Daniltso, A. Volume increasing of secret message in a fixed graphical stego container based on intelligent image analysis, - Information Technology and Security, Vol. 8, N2, (2020): 133-143.
[16] Bilan, S.M. Evolution of two-dimensional cellular automata. New forms of presentation, - Ukrainian Journal of Information Technologies, т. 3, №1, (2021): 85-90.
[17] Bilan, S.M. A Technique for Describing and Transforming Images Based on the Evolution of Cellular Automata, - Intelligent Solutions (Computational Intelligence & Decision Making Theory). Vol-3106, (2021): 106-115.
[18] Albdour, N. Zanoon, N. A Steganographic Method Based on Roberts Operator. Jordan Journal of Electrical Engineering, V. 6, N3, (2020): 265-273