ECONTECHMOD. AN INTERNATIONAL QUARTERLY JOURNAL – 2019. Vol. 8. No. 2. 39–44

Improving Image Sharpness by Surface Recognition

D. Zerbino

Automated Control System
Lviv Polytechnic National University
28a, S. Bandery Str., Lviv, 79008, Ukraine;
Lviv, Ukraine
zerbino@polynet.lviv.ua

Received May 01, 2019: accepted June 22, 2019

Abstract. The article proposes a rule for improving image sharpness and analyzes its implementation by means of the cellular automata formalism and neural networks. It has been proved, that the previously known contrasting algorithm, which uses a template and 3x3 pixels, can be improved considerably by repeatedly applying the iterative process over templates 2x2 with the rule "anti – blur" (C 11 = C 11 x F - (C 12 + C 21 + +C 22) x S) and gradient color correction at each step after the "anti – blur". Colors of images in the template are presented as real numbers (R, G, B). To correct the gradient (C11 < C12, C11 < C21, C11 <C 22, C 12 < C 22, C 21 < C 22) it is necessary to choose a number Cij, that requires minimal tightening in the direction of the neighbor's color. Number of necessary iterations of the rules application depends on the image.

Key words: cellular automata, image contrasting, sharpness, correct gradient, logical correction of colors, neocognitron

INTRODUCTION

All natural phenomena can be interpreted using logic. Logic exists both at the Macro-level and at the color level in the image that we see. It forms an internal trust assessment of each image fragment and adjusts the depth of the logical analysis of each detail in the image. The task of raising the contrast of the image is not new. In [1] (p. 387) the rule using the template 3 x 3 pixels is reviewed. It increases the brightness of the central pixel fivefold and subtracts from it the brightness of neighboring pixels on the left, right, up and down. This rule usually works very fast and gives a good result of increasing the contrast, but it's good to apply when there is almost no noise in the image.

USAGE OF LOGIC IN IMAGE ANALYSIS

In the presence of noise in the image, the classic method of raising contrast also enhances the noise that spoils the image. The phenomenon of perception of a person's image is that a person screens the image through a "logical filter". In [2, 7, 8] A model of the neural network (neocognitron [12, 13]) is proposed which allows to make similar logic filters based on comparison of different fragments of the image. We propose the general definition of the logical filter: "A person sees what is logically expected to be seen, and the elements that it does not see are complemented by his/her logic with a contrast level that is sufficient for this logic".

One thing is to deal with real data in the form of incomplete, fuzzy, or distorted image, and the very second thing is to understand and have information about every detail in the image. Information is a clarification of data [3], and logic helps to clarify the data at the lowest level. It is obvious that without given explanation the data set worth nothing, their interpretation and interconnections should be given, their usage should be explained. Logic allows to imagine the process that created the data, and also helps to understand why other data could not appear as a result of this process.

Logic can be used not only for analysis of images, but also for their synthesis. To improve the contrast, realism and trust in the image it is necessary to distinguish noises, and replace them by more plausible pixels.

The purpose of this work is to apply logic to the analysis of the correctness of the smallest elements of the image (pixels), to find logically sound ways to improve the resolution of images. We will try to improve the image by analyzing its logical content. To do this, we will make a logical assumption on what is shown on the image that has to be improved.

Let us assume that the image depicts real object which consists of surfaces. To improve this image, it is necessary to understand how these surfaces are located and lit. Sometimes it is easier to synthesize the object model than to restore it distorted image items.

Image sharpness enhancement is also needed to build 3D models of an object with 2D images. It is important for a variety of medical devices, as well as for architectural projects in robotics and contemporary art using 3D models. Approaches to these tasks are based on the laws of reflection of light from surfaces, as well as on the basis of the laws of focusing lenses and stereoscopy [4].

For the solution of image processing tasks, artificial neural networks are traditionally used [2, 9, 10, 11]. But the entire concept of the neural network is aimed only at recognizing certain elements of the image. In addition, any neural network needs minimal learning. In the proposed approach it is necessary not only to analyze, but also to optimally correct the image. Therefore, for the solution of this problem, the formalism of cellular or distributed cellular automata is more suitable.

Distributed computing technology for this trend is very important, because in order to improve the image in real time, a large number of operations of adding and multiplying for each pixel is required. The cellular automata algorithm can be implemented in the form of specialized programmable logic matrix (FPGA) with deep parallelism, and it is resistant to local defects in electronic equipment [5, 12–14].

APPLICATION OF THE PRINCIPLE OF CELLULAR AUTOMATA

The cellular automaton is data processing models based on replacing some pre-defined local patterns with others, and possibly do this in an arbitrary order [5]. This at first glance, is a simple model allows organize a complicate information processing [6]. Amazingly, this model is also the basis of all known chemical and biological processes. It makes no mistakes only due to the precise logical interpretation of each rule that is presented as a template, as well as a clear logical interpretation of the data for which this rule is applied or prohibited.

Each cell of a cellular automaton has its own coordinates. If the cell fails, another cell will be programmed to replace it, which will respond to the same coordinates. The task of each cell is to convey its status information to adjacent cells that are part of the rule template, which simultaneously replaces all the states of the template cells with other values. The adjacent cell should not be physically close, but should have programmed coordinates, when accessed to it, it is activated - sends or receives data from adjacent cells.

Electronic equipment in cells can be virtual. For example, if a cell uses multiplication device, this device can be alternately communicated to the cells that need it, and in the memory of the cell only the result of the operation is stored.

In classical computers, the active element is the processor, and the memory consists of passive cells. In the model of cellular automata, on the contrary – the cell is seeking a free arithmetic device for the implementation of its rule and receives data from neighboring cells.

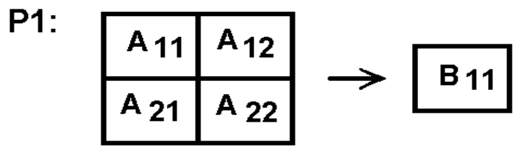
The concept of cellular automata is just a model of computations based on elementary rules. Cellular automata therefore may have virtual cells and virtual arithmetic devices. You can communicate with adjacent cells by one of free data buses that can be on each step in a cellular automata device. If each cell need to have an expensive device, for example, such as a device for multiplying, then such devices can alternately connect to the cells that they need. In the cells to which they are connected, only the multiplication result will remain. That is, cellular automata with virtual cells and devices are possible. There are models of cellular automata based on the cyclic shift of data of the automaton space of cells on a certain cyclic trajectory, which contains a rule pattern (or trajectories – depending on the programmed rules). This allows for cheaper equipment, since it does not require a large number of data buses.

Each design of a hardware or software cellular automata is unique and depends on the tasks assigned to it. However, regardless of technical implementations, the theoretical foundations of cellular automata are based on the concept of replacing data over spatial patterns that intersect. Often the rules are very simple, understandable and universal.

Applying the rule at certain coordinates of the automaton data space means that you choose another valid state of the automaton that brings it closer to the purpose in accordance with this rule. An automatable space will be considered an entire image. We will consider one pixel as an automaton cell, which includes the vector of real and integers. Real numbers mean the components of the color: “Red”, “Green”, “Blue”, and integer values mean pixel`s status, logical markers or counters, for example: “circle center”, “reference point”, “border”, “distance to border”,... The integer values in the automaton cell can also be event counters that occur in adjacent cells (pixels) or pixels distant from it, the distance of which depends on the number of iterations of the rule that calculates events at a distance.

IMAGE CONTRAST RULES

The rules of the automaton for processing images can be formulated on templates of 4 cells (2x2). There are a lot of rules can be processed on automaton space simultaneously (in parallel mode). Consider the rules of the cellular automaton to increase the contrast of the image.



$$B_{11} = (A_{11} + A_{12} + A_{21} + A_{22}) \times F2$$

Fig. 1. Rule **P**1 of the cellular automaton for color averaging.

Figure 1 shows the rule **P**1 for finding the average value of the color of 4 neighboring pixels (***F***2 <0.25). The question is: how finding a medium color can affect the contrast of the image? In [6], formulas have been found to increase the contrast, which are based on the stretching of the color interval from the minimum to the maximum pixel value on the selected part of the image.

However, in our opinion, such transformations are not natural and do not preserve the realistic contrast of the image. In order to maintain the realism and increase the contrast, it is necessary to invert the process of uniform image blurring. A uniform image blurring can be done by iterative replacing the colors in each of the 4 neighboring pixels by their average color (as shown in Fig. 1). After several iterations, the image will be realistically blurry.

To invert the blur process, it is necessary to replace it by the opposite process. I particular, each of the 4 pixels will be iterative subtracted from the average value of the color **B**11, as shown in Fig. 2:

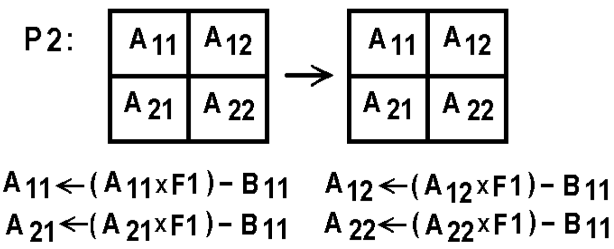


Fig. 2. The rule **P**2 of the cellular automaton to increase the contrast of the image.

In order not to reduce the color of **A**ij to zero or to a negative number, before subtraction it is necessary to multiply the brightness of each pixel **A**ij by some coefficient ***F***1 > 1.0 and ***F***2<0.25 (in Fig. 1). We used ***F***1=1.027 and **F**2=0.006 for best results. However, the formulas used to improve contrast can also produce non-natural colors on random occasions. The formula for color correction is discussed in the next section.

SEARCH FOR REFLECTION SURFACE

One of many brain phenomena is that we logically, paying attention to the smallest details, process the image from the point of view of its content, lighting, location in the space and hints of certain ideas or hypotheses. After formulating these ideas (using the means of its internal logic), the brain begins to seek confirmation of its ideas and hypotheses in the image. After finding the elements of the image that confirm the ideas and hypotheses put forward, the brain comes to understanding of the three-dimensional scene, which looks like a description of what we see, in the form of internal proofs proved in this way. It is likely that there is a reverse process, when by the assertion, the brain forms an image.

The question arises: can one pixel of an image give information that would trigger a hypothesis about an object that is being analyzed? For a positive answer to this question, it is necessary to assume that the data from each pixel should be combined with the data from adjacent pixels, and the data from the combined fragments should be combined with other data of the combined fragments, and so on. Thus, at each step of combining local areas information from their parts should be linked together. And this unity will give a local hypotheses about the causes of differences in the colors of the four selected pixels in the image. A hypothesis, expressed in the form of a rule, has to define whether to adjust the colors of individual fragments to increase the realism of the image. Based on this hypothesis, the brain also concludes about the composition of the material from which the object is made and the angle to the surface of the reflection of light. Whatever material the surface of the object is made of, the brightness of any four pixels for each component of color should satisfy the system of inequalities presented in Fig. 3. This system of conditions is invariant with respect to the rotation of the matrix 2×2 by 90 degrees; thus, in reality, the conditions are 4 times greater than that shown in Fig. 3.

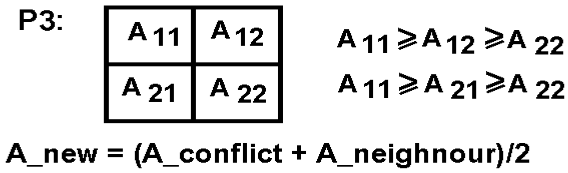


Fig. 3. The conditions that must be fulfilled for each color component

Fig. 4. Example of color corection after 25 iterations

Thus, after each step of the contrast correction (Fig. 1, Fig. 2) for each component of color (RGB), it is necessary to check the correctness of the colors of the neighboring pixels under the conditions presented in Fig.3 If at least one condition is not fulfilled, the correction algorithm has to find among 4 pixels a pair of pixels with minimal differences in which the value of the contradictory color component could be rewritten in the adjacent pixel and thus fulfill all the conditions (Fig. 3). If such a pair of pixels is found then the conflict pixel value has to be corrected: A_new = (A_conflict + + A_neighnour)/2. This is done to maximize the colors of the original image.

After cyclic applying the rules {**P**1, **P**2, **P**3} to the real image (Fig. 4) we get a contrasting and improved image:

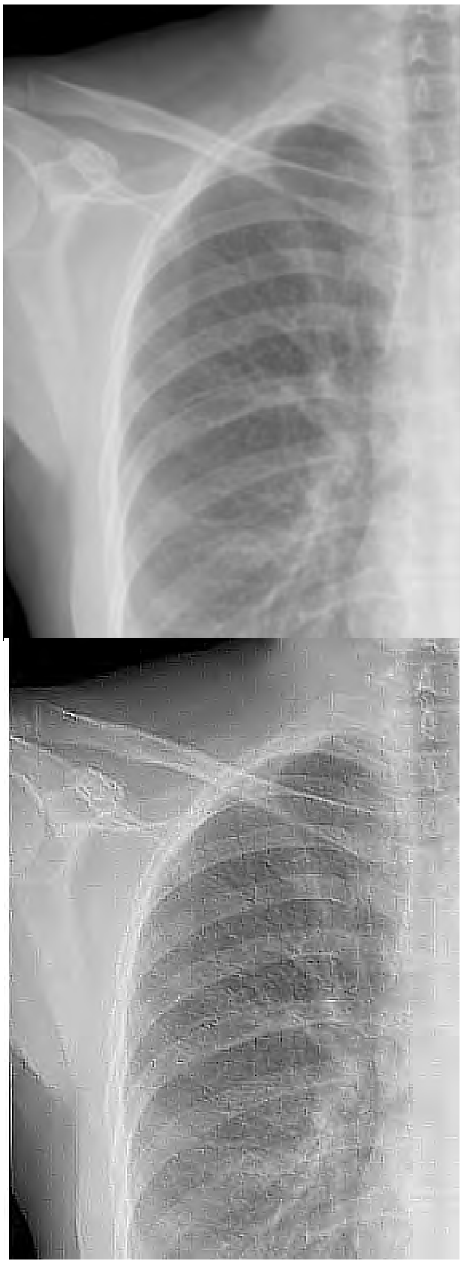


Fig. 5. An example of contrasting an image after 30 iterations

F1 and ***F***2 can adjust the brightness of the image. For the example given, ***F***1 = 1.027, and ***F***2 = 0.006 with 25 cycles of application {**P**1, **P**2, **P**3} to each pixel of the image. If the image does not require lighting (Fig. 5), and requires the allocation of small dark details, then the ***F***2 ratio has to be increased to 0.0065. It is a very sensitive regulator to highlight dark signs in the image. The coefficient ***F***1 is associated with the number of iterations, so it is better not to change it. If it is enlarged, the image will start to light very quickly, and this will result in an imbalance between the number of iterations and ***F***2, which will distort the precision of the contrast. In order to highlight the smallest signs without distortion in this example, it was necessary to raise the number of iterations to 30:

DETERMINE THE DEPTH OF THE PIXEL

In some applications it is necessary to build a 3D scene in 2D image. When constructing a 3D scene that appears in a picture as a flat image, it would be good to know the depth of each pixel, that is, the distance from the pixel to the observer. To understand the problem, let's try, for example, to recognize the depth of each pixel on the surface of the object, which has a uniform chaotic texture (Fig. 6). The first step for this is to maximize the contrast of the image so that the texture has the slightest visible signs. The second step is to identify the following logical features on the object's texture, which would indicate which part of the object is closer to the observer, and which is located further.

Fig. 6. Contrast to determine the statistical characteristics of each pixel environment

If the texture has certain informative features, the statistical frequency of which can be estimated, then these features at different distances will vary in frequency. This means that the cellular algorithm has to take into account not only the indications themselves but also the distances between them. For example, if the texture is green and red, then you need to count the lengths of the green sequences, the lengths of the red sequences, as well as the distance between the red and green sequences, and the average thickness of the borders between any spots.

It can be said that in areas of the images where the indicated distances and sizes are smaller (the statistical frequency of the indications is greater), that part of the object is further away from the observer, and where the distances are enlarged (the statistical frequency of the same indications is less) – there they are closer to the observer. Thus, the specified statistical characteristics can determine the depth of the pixel in the image. Fig. 7 shows the directions of movement of logic markers that will collect the statistics of surrounding pixels:

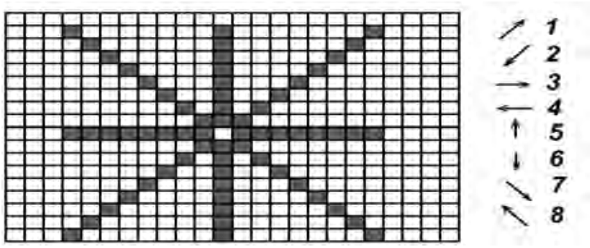


Fig. 7. Areas of the collection of statistical data

You can implement statistical calculations using the following statements. For any logical indication in the image there should be a rule for setting the logical marker in a given cell. Installing the marker at the same time sets the check box for pixel counting with this feature. Resetting a logical indication leads to the preservation of a character count in the current pixel. As a result, each cell of the automaton will contain the vector of indications of the given pixel. By this vector it will be possible to identify the pixel in different images, and according to the value of the meters to calculate the depth of the pixel.

So logical markers initiate processes for counting logical indications, and the counting flags move with each step to other cells until they encounter another indication. Let's consider in more detail, as with each step of the application the rule set flag will be distributed to adjacent cells.

In Fig. 8 it is considered, as fragments on 4 pixels intersect each other. The block of pixels 2x2 of section "**A**" will be considered as one pixel of section "**B**" and vice versa. When averaging 4 pixel colors by the rule **P**1 (Fig. 1), the pixel of the section "**B**" will have an average color of 4 pixels in the "**A**" section. When implementing rule **P**2 (Fig. 2), each pixel of section "**A**" is corrected by data from section "**B**". The principle of processing a cellular automaton remains the same: the information from 4 pixels of one section is transmitted in one pixel of the second section, and in the next step information from 4 pixels of the second section is again transferred to the first section, and so on. When reusing the rules, a logical indication or an indication counter or a distance counter to the indication "splits" immediately into all sides of the adjacent cells, increasing the information about the "neighbors" in each passage pixel.

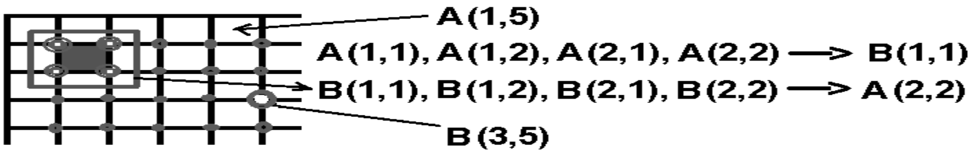


Fig. 8. Interaction between Sections **A** and **B** of a cellular automaton

CONCLUSION

Formalism of cellular automata has been proved to be very convenient and easy for the cyclic implementation of the proposed rules "anti-blur" and "correct gradient". The correctness of these rules has been virtually confirmed. An attempt has been made to use cellular automata formalism for the formulation of more complex algorithms, namely, to determine the depth of each pixel, and to identify identical points on the stereo image. The introduction of special rules, which include a minimal real number, gives new ideas for developing the theory and implementation of cellular automata, possibly in the direction of hybrid automata.

Comparing neural networks with cellular automata, it should be pointed out that none of models can replace another. Models based on neural networks are intended for the classification and recognition of images, and models of cellular automata are designed for parallel processing of information. In our example, processing refers to the replacement of a fuzzy image by contrasted one by means of proposed rules.

However, implementation of the proposed method and obtaining good results requires a large number of operations of addition and multiplication in the format of real numbers. In real time, for modern video systems, it can be applied only under the condition of its hardware implementation in the form of a powerful multi-threaded specialized processor.

REFERENCES

1. Deep Learning Tutorial. The National Conference on Communications, 2017, http://www.ee.iitm.ac.in/ncc2017/tutorials/NCC_DL_Tutorial.pdf
2. **Kunihiko Fukushima. 1980**. Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. Biol. Cybernetics 36, 193 202: 193–202.

3. https://en.wikipedia.org/wiki/Data.
4. https://web.itu.edu.tr/hulyayalcin/Signal_Processing_Books/2010_Szeliski_ComputerVision.pdf
5. Zerbino D., Farid T. 1999. Realization of Complex Arithmetic on Cellular Automata”. Parallel Computing Technologies, Lecture Notes in Computer Science, Springier, 1999, Vol. 1662: 479–480.
6. **Zerbino D.D., Tereshkiv S.M. 2017**. The Connection Between Prime Numbers”, In: Computer Science and Information Techologies CSIT 2017, 05–08 September, 2017, Lviv, Ukraine: 438–441.
7. **Brytik V., Grebinnik O., Kobziev V. 2016.** RESEARCH THE POSSIBILITIES OF DIFFERENT FILTERS AND THEIR APPLICATION TO IMAGE RECOGNITION PROBLEMS. Econtechmod. Vol. 5, No 4: 21–27.
8. **Veres O., Kis Ya., Kugivchak V., Rishniak I. 2018.** Development of a Reverse-search System of Similar or Identical Images. Econtechmod. Vol. 7, No 2: 23–30.
9. **Shumeiko A., Smorodskyi V.. 2017.** Discrete trigonometric transform and its usage in digital image processing. Econtechmod. Vol. 6, No 4: 21–26.
10. **Rybchak Z., Basystiuk O. 2017.** Analysis of computer vision and image analysis technics. Econtechmod. Vol. 6, no 2: 79–84.
11. **Sajad Sabzi, Yousef Abbaspour-Gilandeh, Ginés García-Mateos, 2018.** A new approach for visual identification of orange varieties using neural networks and metaheuristic algorithms. In: Information Processing in Agriculture, Vol. 5, Issue 1: 162–172.
12. **Lessmann, M. 2015.** Learning of invariant object recognition in hierarchical neural networks using temporal continuity. In: ELCVIA Electronic Letters On Computer Vision And Image Analysis, 14(3), 16–18.
13. **Romanuke, V. V. 2016.** Optimal Pixel-to-Shift Standard Deviation Ratio for Training 2-Layer Perceptron on Shifted 60 × 80 Images with Pixel Distortion in Classifying Shifting-Distorted Objects. Applied Computer Systems, 19(1), 61–70.
14. **Ryan N. Rakvic, Hau Ngo, Randy P. Broussard, Robert W. Ives. 2010.** Comparing an FPGA to a Cell for an Image Processing Application. EURASIP Journal on Advances in Signal Processing.