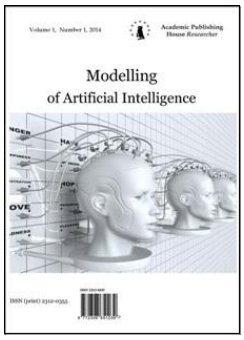
Copyright © 2014 by Academic Publishing House *Researcher*



Published in the Russian Federation
Modeling of Artificial Intelligence
Has been issued since 2014.
ISSN: 2312-0355
Vol. 3, No. 3, pp. 98-120, 2014

DOI: 10.13187/mai.2014.3.98
www.ejournal11.com

UDC 004;519;621

Visualization of Cellular Automata in Nanotechnology

[1] Gennady Ya. Krasnikov
[2] Igor V. Matyushkin
[3] Sergey V. Korobov

[1-3] JSC Mikron, Russian Federation
Moscow, 124460, 1st Zapadny pr., 12/1
[1] The Director General, academician of the RAS, professor
[2] Head of laboratory in Dep. of Persp. Technol. in Nanoelectronics, PhD (physico-mathematical sciences)
E-mail: imatyushkin@sitronics.com
[3] Ph.D., junior researcher

Abstract

We present classification methods of cellular automata (CA) visualization. A general CA visualization model has been formalized. Visualization goals are defined within the scope of CA simulators. We introduce a new class of CA with locking. Some visualization methods are illustrated on the base of a simple 3D CA imitation of this class.

Keywords: cellular automata; visualization; nanotechnology; modeling; multimedia.

Cellular automata theory associated with the names of Von Neumann and Conrad Zuse has great fundamental importance for the overall science and various application. Beginning from publications of T. Toffoli and N. Margolus in 1980ies [1] cellular automata (CA) have been used in physical-chemical processes simulation. By the middle nineties cellular-automata modeling has penetrated [2] into the humanities dealing with exploration of multiagent systems in urban planning (crowd, traffic jam). The recent ten-year period was marked by publication boom in different fields of science relating to CA models [3, 4]; at the same time the mathematical theory of cellular automata continues its development [5]. Provision of CA models with the special software, so called “machines of cellular automata” (MCA is not to be confused with CAM in terms of Toffoli; MCA could be considered as a particular case of Integrated Problem Solving Environments, i.e. PSEs), is inevitably related to some extent to visualization problems. The analysis of the existing MCA is given by L.Naumov in [6]; the review [7] is still actual too. However the authors of CA models, in spite of some spontaneous discoveries, did not approach the visualization problems as system problems. To fill in this gap is the aim of this paper continuing our previous work devoted to MCA SoftCAM development [8].

We pay special attention to 3D CA. First of all exactly these 3D CA models most adequately and naturally reproduce the gaining technology and properties of nanostructured materials (particularly, porous [9] and ceramic materials used in nano- and microsystems, and photonic crystals as well). Critical problems include material and energy transfer in VLSI interconnections,

intermediate low-K insulator (aerogel or xerogel SiO_2). Secondly, the problems of Data Mining (smart data analysis) are of great importance here. Those problems aimed to solve hydrodynamics issues have caused a relatively new (1987) discipline – "scientific visualization" – and are still actual in CA modeling [10]: «…One of the prime motivating factors for integrating graphics into scientific and technical fields is the evergrowing volume and complexity of information needing analysis. Computers are a prime cause of the information glut but also offers a solution through graphic imaging. When a researcher generates large quantities of data, they cannot afford to narrowly and deeply analyze every bit of it».

Besides, CA are often used directly for creation of art images (art visualization [11], algorithmic music [12]). It can be illustrated by Continuous-Valued Cellular Automata in the CAPOW program [13], modeling [14] the wave equation. In terms of possible applications in the mobile multimedia market which are popular among teenagers, the analysis of potential visualization strategies is of great interest too.

Visualization regarding MCA architecture

On one hand, visualization problems are solved by a separate subsystem within MCA, on the other hand, they are inseparably linked with the whole computational process (Figure 1). The interaction can be even unexpected: so, the parallel performance of CA model upon CUDA-technology [15,16] enables the same low-level hardware as the rendering. Low-level issues of visualization at the remote terminal with a cloud/network implementation of CA model also requires consideration. MCAs are developing along with the parallel computing technologies and usually, for example in MCA CAME&L [6] and CAMELot [17], provide their support in a standard distribution kit. Low-level visualization problems have rather a «back-end» nature than a scientific one, and depend on details of the computing system architecture and the concrete realization of data flows. However we shall concentrate upon much higher-level aspects of CA visualization having supposed a common situation in OpenGL or DirectX application.

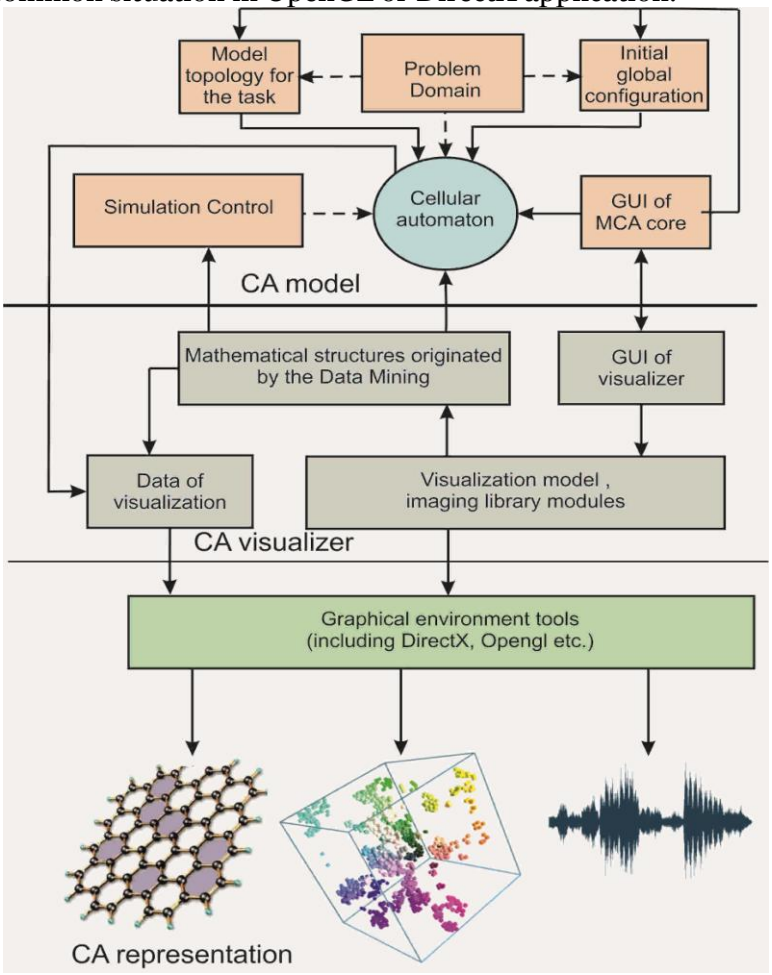


Figure 1. MCA architecture in terms of visualization.

Should we discriminate between the problems of visualization and the problems of editing of the CA field? In many programs like Golly these problems are solved for the users almost with the same interface tools. No doubt that a consistent MCA should provide the user with an ability, due to visualization results in real-time mode, to stop the computation at any moment, to modify the cell content (or the content of the cells in case there is need to download one of the necessary patterns from the library file), and to continue monitoring further computation. We clearly see that the visualization problems arise not only on the stage of representation of the output data, but already on the stage of preparation of the input data, even on the stage of initial global CA pattern which in terms of mathematics is an integral part of CA model. It expresses the active character of visualization [18]. However it would be more correct to discriminate between visualization of CA computation directly and visualization of CA model's initial data (even if these data are the previous results), the parameters of each of them may differ much at that. This statement is well illustrated by presence of rapidly increasing and unlimited configurations which is typical for many CA; to perceive the essence of the process it is necessary to enable the auto scaling. A large scale and a middle scale with detailed visual cell displaying are necessary for editing a CA field, and a middle scale and a small scale – for visualization of CA with an unlimited population when the screen pixel is equal to tens of cells.

Many CA simulators, first of all those ones which are created as java-applets and that is why are very much alike with entertaining games, make the MCA designers to pay much attention to the real time mode. In general often it is not considered that data generation process and data visualization may be separated in time. A kind of compromise might be considered as video compiled with computer screenshots captured during CA modeling. It seems to be more correct in terms of time sharing to generate shots of picture directly corresponding to the global CA configurations. Comparing with the MATLAB system graphics it would be more preferable to use the «imwrite» command than the «print/saveas» commands. In the above case for a 3D model it is possible to record simultaneously 4 video files (or even 1 video file with a doubled size of the shot) each showing the scene view with a fixed camera position (ignoring the real-time mode rendering).

The next question to the MCA designers is: to keep visualization parameters separately from the CA model or to include them its description? Such evident parameters for the Game of Life (GoL) are the color of the pixel or the square answering a live or dead cell. It is natural for the 3D CA to put conformity, but not identification between the cell and the voxel/metaball of the geometric model. Then instead of color identification of the cell state it would be interesting to use forms and sizes of the metaballs. Anyway, it is possible to treat the visualization parameters abstractly. Certainly, these parameters are linked to the visualization models and have the meaning within the frames of the latter ones. CA simulators offer default coloring scheme and provide with interface tools of colormap change (for example, a gradient filling option between two colors is offered in Golly). Even in advanced simulators the items of visualization and visualizer's interface are settled primitively. We consider the main point of the problem on base of the simulator Golly [19] and its embedded 2D CA "Generations", its specific pattern "Burst" has been given in the file burst.mcl (Mirek's Cellebration MCell format, widely popular simulator operating with modified RLE-format data):

01 #MCell 4.00
02 #GAME Generations
03 #RULE 0235678/3468/9
04 #BOARD 400x400
05 #SPEED 0
06 #WRAP 1
07 #CCOLORS 9
08#L 12.A$12.A$12.A$12.A$12.A$12.A$12.A$11.3A$11.3A$9.7A$9.7A$7.11A$25A$7.
09 #L 11A$9.7A$9.7A$11.3A$11.3A$12.A$12.A$12.A$12.A$12.A$12.A$12.A

(the lines are enumerated, the comment lines are deleted, the lines devoted to visualization are bold-typed)

The second line indicates the class of CA and simultaneously gives a prompt to Golly which colormap should be loaded (on default, gradient from yellow to red for live cells, i.e. from young to

old). The third line describes CA rules, in this case: life cycle of generation – 9 steps, and survival and birth rules as well. Sometimes in this line there is a text name of the transition function referring to the more detailed description. The fourth line specifies the initial size of the CA field and at the same time the size of the visualization field. The fifth and the sixth lines refer to the computation features. The seventh line states the number of the colors used - 9. In other variants of mcl-files it is just omitted for it results from the line 03 and the concept "each state has its own color". The lines 08 and 09 compactly specify the cross pattern according to the RLE formats («.» is a dead cell, A is a young cell, $ is the character of new matrix line, the figure before the dot or a letter is the number of repetitions).

While the lines 02 and 04 mainly refer to CA model and only indirectly to visualization, the line 07 only refers directly to visualization issues. This code does not indicate which colormap is to be used and encloses no references to the code indicating that. With advanced MCA the most promising way of data structuring for XML-code and addressing the DOM-standard promoted by the W3C consortium. But this recognition occurred only in the recent years [20,21]. In our SoftCAM MCA development we based on the following principles:

• The cells classes, the neighborhood template, the transition rules (this triad actually form the CA model), initial patterns and visualization parameters are distributed in the library files;

• The assembly is proceeded as a kind of a representing CA XML file, its structure fields are available for the user's editing (in particular, an user can insert his own C++ code into the SoftCAM editor);

• The visualization model and parameters designated in the XML file can be modified with the visualizer's interface tools.

The third issue is attributed to dimension. *It is important to underline that CA dimension and topology do not determine the visualization dimension and topology*. We identify the visualization dimension as dimension of array data elements forming the display image. For example, if the CA "Generations" cell is brought to correspondence with a certain height of the colored column proportional to the color (with the fixed colormap), the visualization model would not become tridimensional though it is clear that the 3D graphics has been used. If the voxels for different step numbers (applicate axis – time) are mapped in this 2D CA the visualization model would not become a 3D type. We are able to choose the image we need though a linear visualization is natural for 1D CA, and a chessboard kind – for 2D CA, and a cube or sphere kind – for 3D.

The fourth aspect concerns the need in generation of additional mathematical structures caused by necessity of CA "raw" data array analysis and, accordingly, those data which are subject to visualization first of all. It is the case when the CA is played in a cube having the edge of 256 cells in 256 steps, and the volume of "raw" data is 2^{32} byte = 4Gb. Generally, these structures are interactive, they accumulate the scientific meaning of CA computation because the CA simulation results must be treated semantically, and it could be realized with the above supporting structures. To fill them with number content it is convenient to foresee a call of a certain function after each global transition iteration of CA computation (that can be easily described in a certain partition of XML file SoftCAM). The necessity of carrying out such operations on the global CA configuration is caused by a complicated structure of CA cell assigned to simulate the physical process. It is obvious that complication of the cell state leads to increase the visualization variants. It is accepted in SoftCAM that the cell state is defined rather neither by a byte, nor by an integer byte, but by a union ("union/record/structure" in different programming languages) of any primitive objects.

Let us consider an instructive example produced by Denis Krylov in the early 90ies in DOS simulator of Conway's game. GoL itself is despondently bicolor and in order to paint its animation at each step the time arithmetic average of each cell state was memorized (or the total sum of bits having been in the cell), i.e. when $x_{ij}(t)$ - is the state of the cell at the t-th step, then the voxel color is determined as $y_{ij}(t) = \frac{1}{t}\sum_{k=1}^{t} x_{ij}(k)$. Color indexing was chosen as a real variable (see the Appendix) and in the place of the cell a colored voxel was mapped instead of a common "black and white" view. Gradient color transition allowed to get artistic images. Figure 2 given below shows an implementation of such a visualization method in MCA SoftCAM for a rather famous figure «*r*-

pentamino» that gives stochastic evolution at the first 1103 steps, then it becomes periodic with a two steps period. Such method allows to recognize the history of the previous steps of CA evolution. For instance, in the center of Figure 2(d) there is a green block which is absent in Figure 2(c); it means that this figure has existed for a long time but by the present step it has disappeared.

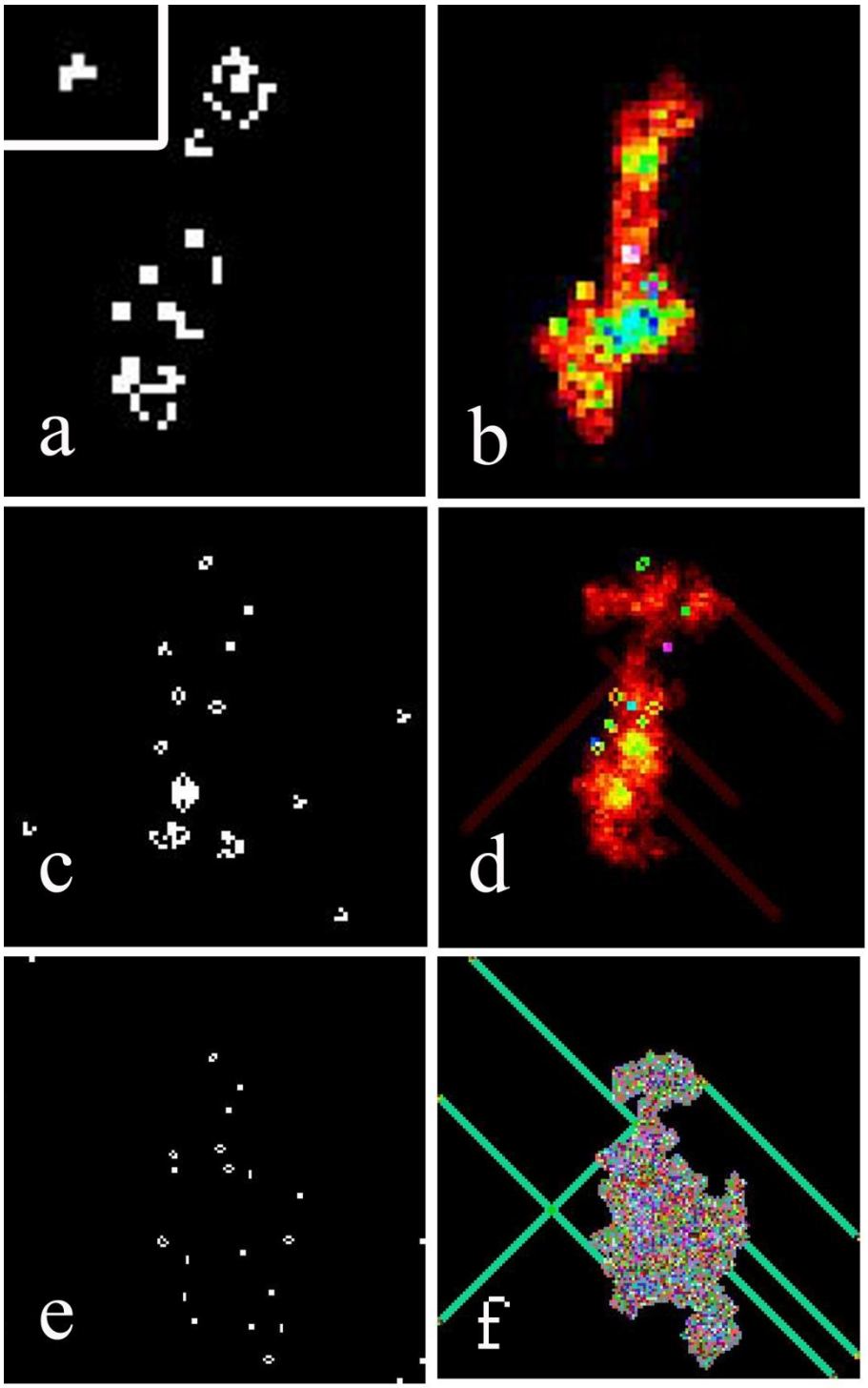


Figure 2. *r*-pentamino evolution in the GoL. The left column (a,c,e) presents classic visualization for the 25th step (a,b), for the 175th step (c,d) and for the 1103th step (e,f). The right column presents visualization with the arithmetic average, and (b,d) are given for the color map A, (f) – for the color map B (see Appendix). Fig. (a) shows frame-by-frame the *r*-pentamino having 5 squares. Diagonal row follows the "glider" flight path. The condition of "death line" on the edges of CA leads to the appearance of white spots caused by the flown away "gliders" on the edges of Fig. (e,f).

Hypothetically we can consider the CA model "Generations" concerning distribution of an epidemic (color indicates a stage of disease), having begun at the point center of the circle. Assume that the CA grid is square and we could evaluate the possibility of an onset of immunity with certain probability. An epidemiologist can be interested in dynamics of the number of those who have died and those a who have immunity depending on the distance to the nidus of infection. Then the visualization model would be 1D, and the "lost" of dimension is caused by integration along the concentric circles at each iteration. With the help of such a method we shall get an animated column diagram.

Generated on base of the "raw" CA data the structures could verify the CA model itself. With the cellular automata simulation of diffusion processes it is necessary that the local transition equations meet the substance conservation requirements. It is especially important because classic CAs are discrete. In block CAs of the Margolus automata type this requirement is met automatically but in other cases the problem is still urgent. As V. Vanag put it quite fairly [22], "There arises a question: How can we find then the local rules following which a set of separate elements of the system in total will reproduce the dynamics desired? This question still has no answer." That is why it is desirable to integrate along the whole field at each iteration of CA computing in order to identify an "unlawful" change of the number of atoms of any kind.

General principles of CA visualization

In the field of visualization research there is a "periodic table" specially developed [23]. However it is rather abstract and too oriented to presentation of data from humanities; CAs need their own special methods to be developed.

It is necessary *to treat the CA matrix* being saved in memory or in a hard disc *as an uncertain metaphysical entity permitting a variety of presentations*. This approach adopted from geometrical modeling is applicable not only for 3D but for any other CAs, and for both much simpler ones, and even much more complicated in topography (for example, for triangle and hexagonal grid). Experience in geometrical modeling can be used in solving the CA visualization problems, though partially, because: a) it has to do directly with three-dimensional solid body; b) CA model provides with comprehensive data on the object and the geometrical model tends to develop it from the compact mathematical formulas; c) as one solid body can be described with a number of presentations (models), the same is with one CA – it can be also described with a number of visualization models; d) methods of interaction with computer graphic environments are the same. It is useful to borrow the term "voxel" from computer graphics; here we define it as that that is on the computer display corresponding to the CA cell.

One of the principles of scientific visualization, especially important for 3D CA, is elimination of redundant data. Even in 70ies the forerunner of scientific visualization W. Bowman wrote [24]: "In the graphic figure natural features can be shown in terms of their physical view – or they can be formalized to underline the most essential details. The structure is understood as a physical constitution of the object which is not perceived in common circumstances. The structure stands exposed when in the graphic structure there appear those inner elements of the object which are usually hidden or those inner elements of the object which can be seen only having admitted that its outer elements are transparent".

Introduction of auxiliary mathematic structures in this case is one of the ways of Data Mining application to the cellular automation data. Data Mining methods for CA are still waiting their application but we can preliminarily outline the targets on the basis of only CA mathematic theory demands. For example, in case of massive configuration of the GoL (we assume, 300×300 cells) we are interested not in each cell behavior, but in the number of "gliders" born, of stable "hive" or "block" configurations, and of "space ships" and other popular patterns. With the visualization then we can draw any stylized sign or a metaphor instead of a flying "glider". As for the CA simulators of nanotechnologies, with describing the processes of substance crystallization or pore-formation it is desirable to implement smart functions of determination of the number of clusters in the CA system, of the moment (and conditions) of the percolation threshold, etc. At last the CA simulator has its research importance only in case when it is able to predict macroscopic parameters, and to

provide this it is necessary to aggregate microscopic parameters and those which are directly accessible for the CA data.

Let us give an abstract definition of visualization model. First of all we must identify the "raw" data *Y* of CA computation. We enter $B_1,B_2,\ldots,B_N$ – the elementary sets on the base of which the cell state *X* is built, the set *M* of CA-matrix indices built on the basic of time tuple *T* and a set of special indices M_s of dimension *d:*

$$X = B_1 \times B_2 \times \ldots \times B_N,\ M_s \subset \mathbb{Z}^d,\ \mathrm{T} = \langle 0,1,\ldots,T \rangle,\ M = M_s \times \mathrm{T}$$
$$f : M \to X \Leftrightarrow \quad Y \subset M \times X, \qquad (1)$$
$$(\forall m \in M)(\exists y \in Y : pr_1 y = m),(\forall y_1, y_2 \in Y)((pr_1 y_1 = pr_1 y_2) \Rightarrow (y_1 = y_2))$$

Here *pr* – is the projection derivation operation for the set or its element determined by Cartesian product (×). Notice that though the definition (1) looks like the CA definition, there is no concept of the transition function and the neighborhood template which essentially fill the parameter *d* with practical sense, i.e. "CA dimension". With visualization CA data dimension adopted from CA model has an insignificant value; in fact, coming from the data storage method in computer memory as a linear sequence of indicators of blocks of a fixed size, this dimension can be equal to 1.

Enter a set of indices *A*, belonging to *D*-dimension array and partitioning of the initial set *M* into several parts:

$$M = \bigcup_{\alpha \in A} M_\alpha, \quad A \subset \mathbb{Z}^D, \quad D = 1,2,3 \qquad (2)$$

If the array coincides with the initial CA model, then $D = d + 1$. This partitioning, from one hand, introduces an index visualization set and, accordingly, a dimension of the visualization model *D*, and from another hand, prepares the following data aggregation (if we do not carry out the aggregation we'll have $|A| = |M|$). The method of partitioning can be of any kind: either unconstrained consisting of, for example, synchronism fixation, i.e. $|A| = |M_s|$, or conditional depending on the cell states (then in a general case $|M_\alpha| \neq |M_\beta|$, $\alpha, \beta \in A$). For instance, when counting the number of nanocrystals in the CA crystallization model (or the number of pores in the pore-formation model) we can accept that the cells belong to one cluster if there exists 1 or 2 cells having the same component value as in one of component states (B_1) in the cell neighborhood. In this case the rules of neighborhood template definition *U* must be specified if this general case differs from that one used in CA computation:

$$(\forall \alpha \in A)(\forall m \in M_\alpha)(\exists m' \in M_\alpha)$$
$$((m' \in U(m)) \wedge (pr_1 f(m) = pr_1 f(m')) \wedge (pr_2 m = pr_2 m')) \qquad (3)$$

So, we can speak about the abstract partition function $g : 2^{(M \times X)} \to \{A\} \times \{M_\alpha\}_{\alpha \in A}$, i.e. each "raw" *Y* is responded by the index set *A* and the set partition family $\{M_\alpha\}$. To aggregate the data we'll introduce the rule representing the function (in terms of programming), permitting the argument list of variable length:

$$h : \{n \in \mathrm{N}, X^n\} \to Z \subset \mathbb{Z}^p, \quad 0 < p \le 5 - D \qquad (4)$$

In the example of Figure 2 *h* is performed by the rule of arithmetic average, *d*=2, *D*=3 (two-dimensional array and time) and *p*=2 (color and time). If we take into account a possibility of animation we'll be able to present (*p*+1)=3 of the parameter, notably: dynamics of color and height/size of 3D-voxel (with the assumption that color presents only one parameter). Therefore it could be highly possible to present 2 parameters, and taking into consideration the cognitive complicity of 3D on the 2D-screen we get only 1 presented parameter. In (4) we nominally wrote $\mathbb{Z}$, though there are possible cases of aggregation into integer variables (and with computer real type of data are still discrete) even in Boolean data.

We also give a generic color map for aggregated features that transform the state of a cell group *Z* into color magnitudes and voxel forms *Q* (herewith even common graphics like *y*(*x*) and

$z(x,y)$ can be treated as modifications of a voxel presentation, and the set Q' corresponds to the scene lighting schemes):

$$v: Z \to Q \times Q' \quad (5)$$

It is desirable but not obligatory to apply the requirement of one-to-oneness on the function v. *We call the basic visualization model the seven consisting of three sets and four functions:* $<X,M,A,f,g,h,v>$. Either aggregated features along the whole family of sets with index A or, on the contrary, a part of data within one set with the fixed index $\alpha \in A$ can be visualized. In the first case we shall amplify the basic visualization model with the set of V_1:

$$V_1 = \{\forall \alpha \in A \quad (\alpha, v(h(|M_\alpha|), f(M_\alpha)))\} \quad (6)$$

In the second case we shall amplify it with the set V_2:

$$V_2 = \{\forall m \in M_\alpha \quad (m, v(h(1, f(m))))\}_{\alpha \in A} \quad (7)$$

We do not consider the case when on the computer display there are several visualizations (for example, for the 3D automata – 3 cross sections and 3D animation). We should probably specify a mathematical structure inherited from basic model and the union of sets like V_1 и V_2. Similarly to that, it is possible to specify the animation scenario treated as a sequence of visualization models.

CA visualization methods (models) are classified in Figure 3. The first criterion is the visualization model dimension D, entered according to (2). To explain the case having some topology peculiarities we could imagine CA specified on the spherical surface where chemical reactions are proceeding, here we should take into account the effect of the molecule adsorption layer and the subsurface layer, we could assume it to be charged. Every layer is specified by the cell matrix having its own size. Such a model, from one hand, is no longer two-dimensional but not yet three-dimensional, from the other hand. We should mention coupled CA [25], where could also be some topology peculiarities. As far as we talk around the data indexation and there are four-dimension arrays known in programming, there could be 4D simulators and those of much more dimension; if we simulate with CA endofullerens it would be reasonable to use the presentation "cube-in-cube" in this case [26].

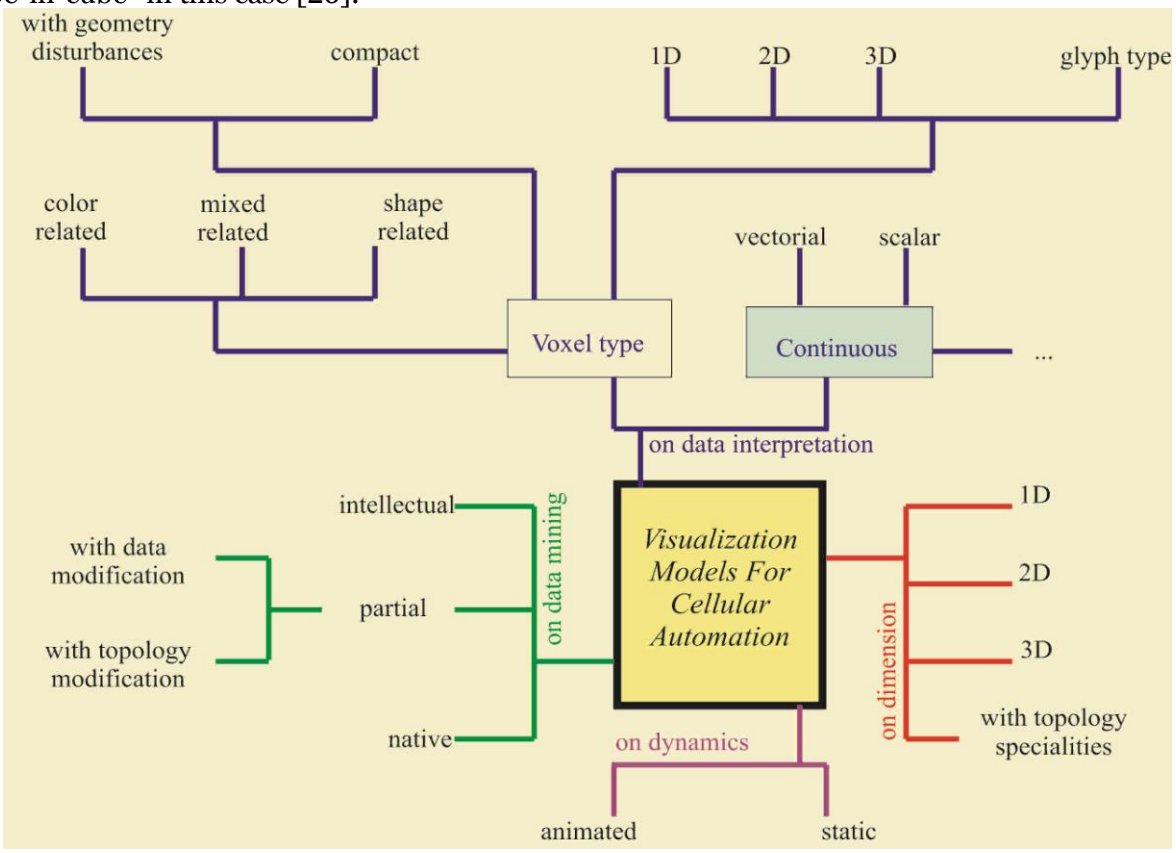


Figure 3. Classification of the methods of cellular automation data visualization according to four criteria

The second criterion is relatively simple – if we are interested in static or dynamic picture. We should only notice that animation can be in a real-time mode and a delayed (step-by-step) mode, it can also keep the data on the screen and remove the old data. As for the third criterion, by the model semantics, we distinguish native, partial and intelligent (smart) models. In the native model the analog approach WYSIWYG is implemented, i.e. the cell state is fully presented without any preprocessing ("what is in CA – that is what I see"). The native model could be called more exactly – complete model, and the word "natural" could be reserved for a narrower class of models: complete, animated in the real-time mode, voxel type, without violation of dimension and topology, i.e. when $D = d$. In the partial models either a certain cell state component (one of $B_1, \ldots, B_N$, and not X, see (1)) or a certain simple function of them $h(1,X)$ is presented (4). Another variant of partial models is based on the principle: a certain data slice in space is visualized that is formalized by partitioning (2) и (7). In smart models the Data Mining methods are supposed to be used with the inevitable aggregating (6), and Figure 2 (the right row) is the result of the smart visualization model.

According to the fourth criterion the models are divided into continuous and voxel. With creating an abstract visualization model (1-7) we implied first of all the voxel type meaning that some possible graphic dependencies can be interpreted through voxels. This proviso mainly deals with the visualization aspect; we used the condition that the part of the common plot $y=y(x)$, arranged between two vertical lines of the grid is considered as curved contour edge of trapezoid voxel with the number identified with the feature of abscissa axis scale. However, with a more thorough analysis this proviso only worsens the matter. The continuous models are doing well when the results of Data Mining having no CA specificity are mapped. These results may be probably processed with general tools and outside the MCA [14]. Following the conventional division for the scientific visualization we have assigned a scalar and a vector models, and the suspension points in Figure 3 mean a possibility of application of the famous, non-specific to the CA methods of classification. Besides, much depends on the initial problem definition: for example, if in each cell corresponding to rather a large area of the physical surface or space, the electric-field vector is specified by CA computation, the direct initial problem might consist in getting the potential continuous distribution. It is worth remembering about the space and time quanta used, and to proceed from the voxels indices mapping to meters and seconds, and we need an interpolation of CA data for that, and we have great interest towards the very type of the interpolator itself. To solve the last problems it is more reasonable to use such tools as the MATLAB software etc.; in this case the data reset into the text file in the process of the CA computation can be processed many times.

That is why we emphasized mainly the voxel simulations. The term "voxel" comes from computer graphics and, being the result of liaison of the English words "volume" and "pixel", and means the element of volumetric image which contains the item value of the object cell. With regard to CA we give a little different definition: voxel – is one or a group of adjacent pixels semantically linked with one CA cell or with one of the CA partition elements. Accordingly, its dimension, though we speak about it rather for convenience, is different: 1D – one pixel, a line segment, an arrow (to map the liquid flow rate, e.g., [27]) or an arc; 1,5D (text and graphics models, which is typical for old simulators [28] and that follows the technical concept of the character cell) – a character or a pictogram. We should not underrate the latter method that seems to be appropriate only for the linguistics CA-models (see [29] – the study of Perm scientists in dislocation generation kinetics. The most popular are 2D voxels (squares, circles, hexagons) and 3D-voxels (cubes, spheres), e.g., in the form of Lego blocks or dice [30]. CAs can be used even for morphing of 3D-objects [31].

As a rule, voxel's color and/or transparency visualizes the cell state (color dependent models) but it is possible to encode some additional data with a form or a size, and here indirectly text and graphics motivation arises. For example, Figure 2(f) allows 3D interpretation in case the voxel height is given proportionally to its color (precisely, to arithmetic mean value). By compact voxel models we mean a case of the screen space natural filling (linear, template, cubic) with voxels without missequencing and hole spaces. If the voxel is transparent, i.e. invisible, we still consider the model to be compact. However, the initial physical topology of the problem (e.g., nanopore material) or just the researcher's wish could demand another arrangement of voxels; imagine, for

example, it is necessary to compute mechanical stress in beams of the "Birds Nest" stadium in China or to study rats behavior in a labyrinth.

Methods of Visualization of one-dimensional CAs. Sounding generation.

Talking on one-dimensional CAs we cannot forget one of the CA theory developers Stephen Wolfram, the creator of Mathematica Software, a concept of experimental mathematics, and the author of the bestseller "A New Kind of Science" devoted to CA [32]. He introduced classification of 1D CA transition rules of the following type: a) the cell state – {0,1}; b) neighborhood template – the cell itself and one neighbor from the left and one neighbor from the right. Then each group of three will be corresponded by the future cell state 0 or 1. If we write out all combinations from (0,0,0) to (1,1,1), they will be corresponded by an eight digit coding the whole transition function. Using this method we get number 256 of the transition rules, some of which allow to get pretty configurations: rule 18 gives the "Sierpinski's carpet" fractal. 1D CA is easily visualized with a static 2D-model (Figure 4, and we intentionally used a 3D-voxel model) in the "length-time" axes.

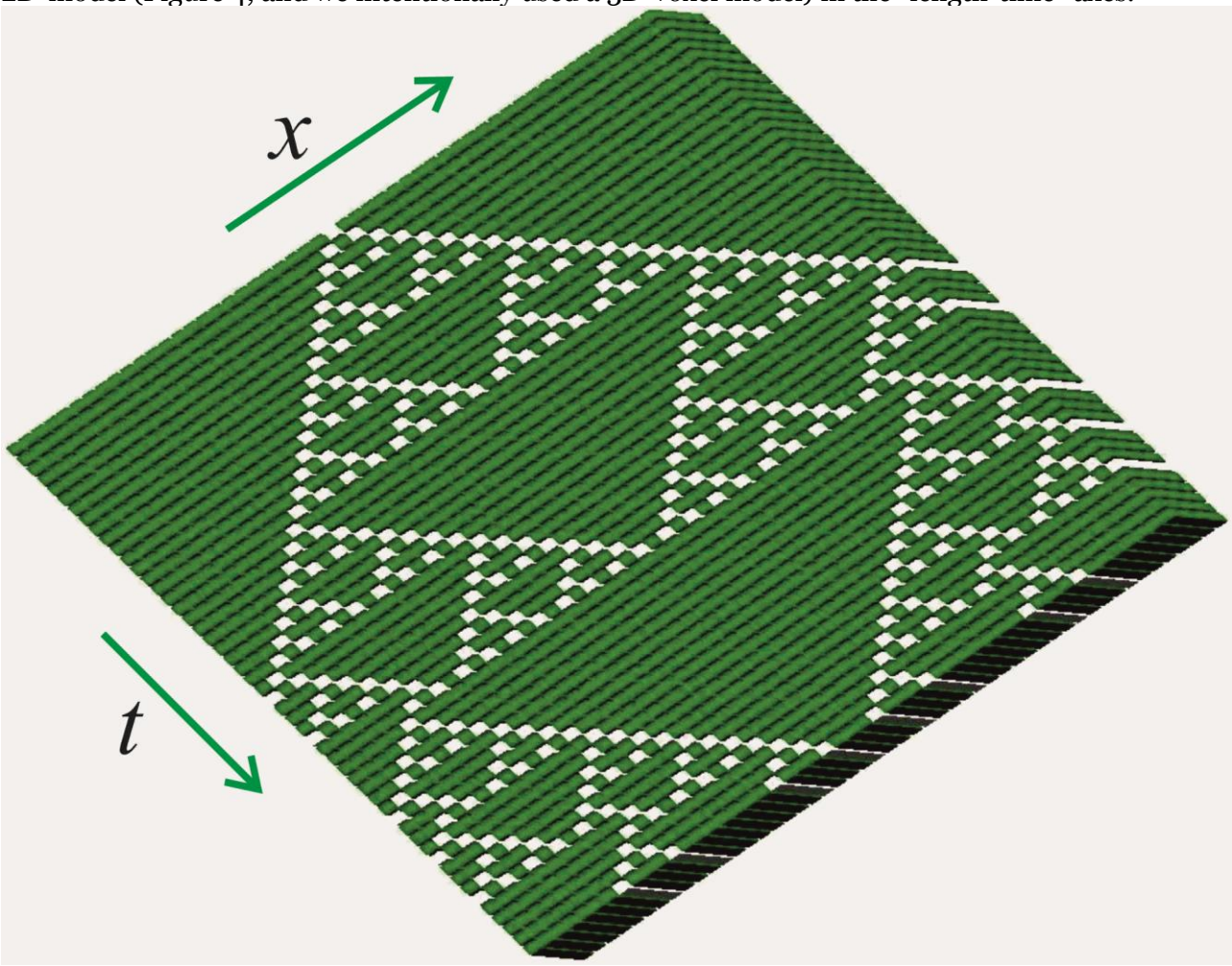


Figure 4. The "Sierpinski's carpet" fractal obtained according to the rule 18 in SoftCAM MCA. Dimension of the CA-model d=1, dimension of the visualization model D=1, dimension of the voxel – 3

A very long row of CA cells should be placed as a matrix, and the numbering can vary (Figure 5). This method application [32] to the natural number sequence with the means of the PYTHAGORAS system has lead to the results of the additive number theory; the idea is to write out 16 numbers in a line. The opposite is also correct: a small 2D CA in regard to the field size can be unrolled in line and mapped like 1D CA. However, in a compressed (rolled) format it is not advisable to disorder the initial neighborhood, as it is shown in Figure 5(c).

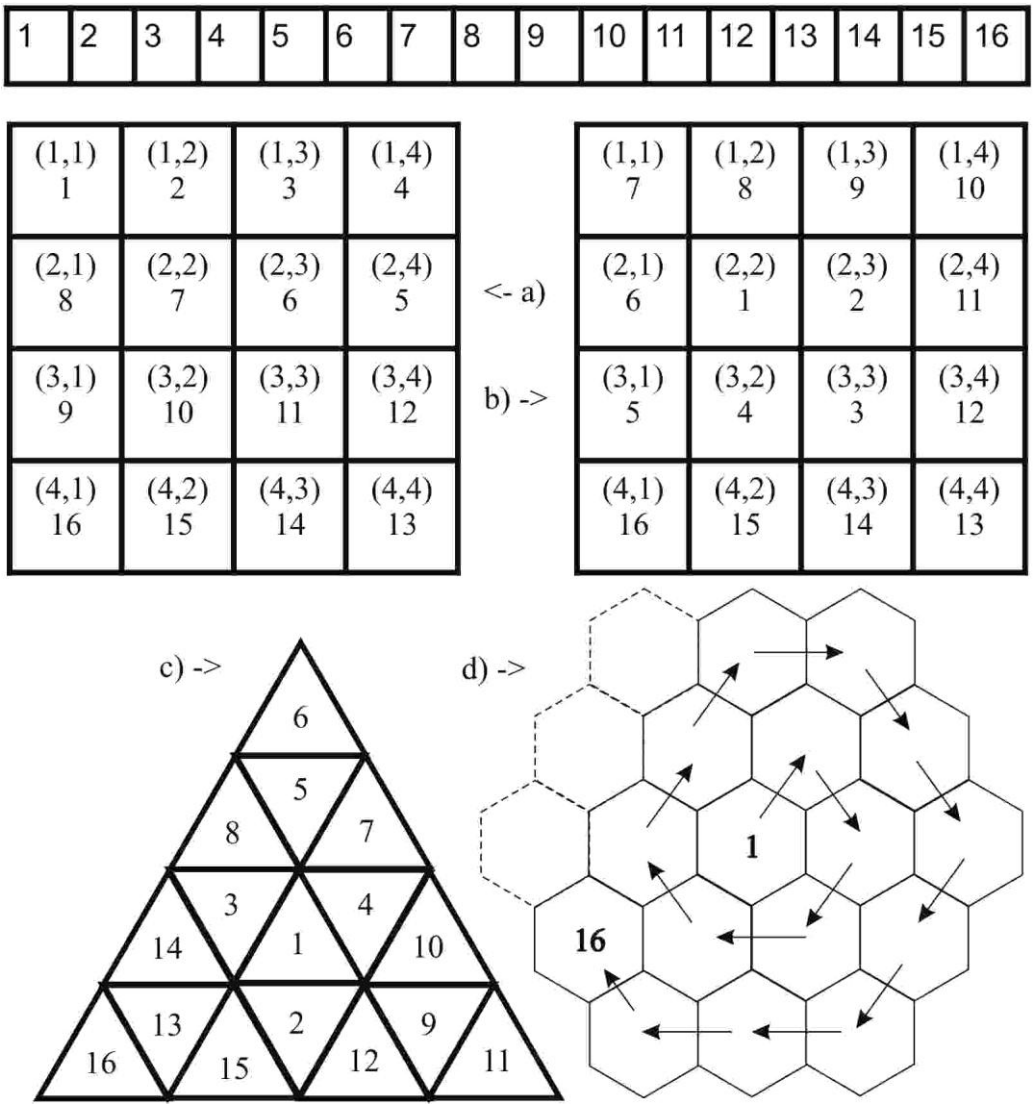


Figure 5. 16-cells 1D CA visualization with a two-dimensional model. a) a serial numbering of the 4×4 square; b) a spiral numbering; c) a triangle numbering "4 in 1"; d) a spiral numbering in hexahedral cells

Traditionally the 1D CAs are used for musical samples (http://tones.wolfram.com/) but of course, there are no limits in dimension of CAs; along with the visualization we can mention the audization of CA. Multimedia opportunities of CAs have been commercialized already. The market of such mini-games is rather large covering teenagers attracted not only with blinking lights but also with unique sounds, and beginners-composers who have no music education.

Recently the Novation company produced a USB-controller Launchpad, designed for live work with the Ableton Live program, having taken a step to the simple and visual control of music editor. The controller has 8x8 appointed keys and buttons for option selection providing the software control. In 2009 Jason Hotchicks programmed audization of the Conway's game for this application [34].

CA-audization presents one of the variants of algorithmic music since CA can generate periodical figures (and any music contains repetitive parts) and are able for chaotic behavior. In early 2000-ies Eduardo Mirando (of Sony Corp.) published a number of articles on this topic and created a CAMUS tool for CA audization [35-37]. There are some other works too [38, 39]. General principles of audization do not likely differ from principles of visualization though some specific features are not excluded. We briefly consider here the problem of CA data transfer into the midi-sequence. As for the mp3-format and more developed music forms including human voice imitation, we express our confidence that it is possible to generate such a kind of music with the

help of CA. If the CA cell state is more complex than of a bit one and is presented a byte (correspond to the notes from "do" to "ti") then it will be much easier to obtain midi-sequences.

In the simplest way, if we assume an L of instruments, then the tape of Boolean 1D CA is divided into $8L$ blocks; each byte codes (Figure 6): with two bits – volume (4 levels), the next group of three bits - duration (in regards to the cycle but not less than 1/8 of the cycle period), the last group of three – the note value (do-re-mi-…). As it is sometimes done [39], it is possible to limit the field with 8×8 cells, implying 8 orchestra instruments. And it is obvious that from the algorithmic point of view it does not matter whether 2D CA or 1D CA considered in the 2D model is audized.

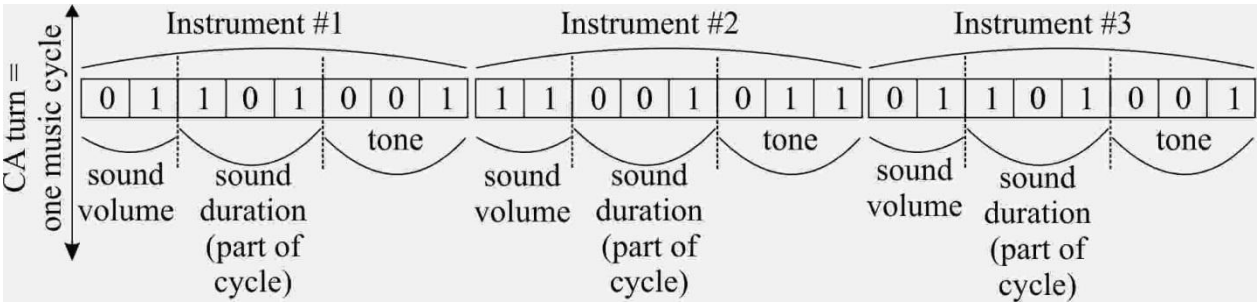


Figure 6. The simple scheme of generation of midi-sequence with the help of 1D CA. L=3

To generate a wav-file it is possible to use the following scheme: a) the total time of sounding is divided into equal small segments of duration T; b) on each k-th segment the wave is presented with pairs (amplitude, phase)= $(a_s, \varphi_s), 1 \le s \le S$, and the S number depends on depth of discrete Fourier transform; c) each pair is given by two adjacent cells of byte CA.

$$W(t_k \le t < t_k + T) = \sum_{s=1}^{S} a_s \cos\left(\frac{2\pi}{T} s + \varphi_s\right), \quad T \sim \frac{S}{440}\text{sec} \qquad (8)$$

Accordingly we divide the 1D CA tape (Figure 7) into multiple fragments, if we need to get an appropriate sounding duration, the length $2S$.

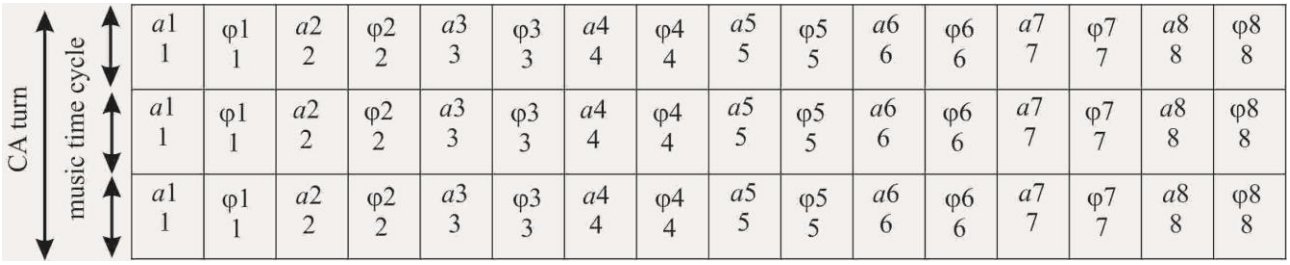


Figure 7. The simple scheme of generation of wav- sound with the help of 1D CA. Three time segments fit in one cycle (step) of CA. S=8, the cell state – 1 byte.

Visualization methods of two-dimensional CA. Isometric view.

Currently 2D CAs, including those used in nanoelectronics, present the majority. 2D CAs are visualized more naturally: the visualization 2D model of the 2D-voxel type with animation along CA steps. Of course, the visualization 3D model is also possible when the time coordinate is transferred into the third dimension. However, it is difficult to name the reason of the necessary reduction to 1D (Figure 5). There might be a difficulty triangularity or hexagonalarity of the grid, but it exists mainly within the CA-model frames, or to be more exact, within the neighborhood template, and slightly affects visualization. Then visualization models with topology modification become possible (Figure 3); Figure 8 shows some hidden problems associated with grid transitions.

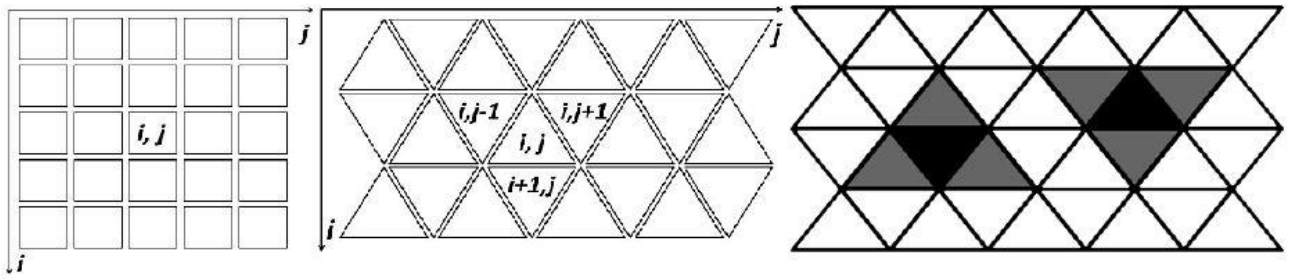


Figure 8. The rectangular grid and indices set in the triangular grid [38]. A 25 cells block is translated to the 24 triangles block. The Neumann neighborhood occupies 4 cells on the square grid and 3 cells – on the triangular grid. To fix the latter it is necessary to consider even and odd rows separately

As it was suggested in one paper, with weakness of graphic system the voxel of the triangular grid would rather have a rectangular form. Along with the description of crystallization on the triangular grid the authors [40] introduce the activity factor of CA on the global turn, i.e. the percentage of cells that have changed their state during a few last iteration. Data Mining processes by analogy, can compute for each of the cells on the basic of the last 10, for example, steps if the cell belongs to 2-, 3- or 4- periodical configuration, either invariably, or it varies chaotically. In this case with using a similar to Figure 2 visualization the smart model is produced.

Hexagonal grids should be used for the processes where isotropy is important, for example, in the problems of hydrodynamics and diffusion. O.Bandman and co-authors described [41] the analog of hydrodynamic FHP-models which imitate movement of tungsten comprising particles in the 2D-space densely filled with hexagons. In each cell there are several particles with their velocity vector directed along one of the 6 straight lines towards the centers of the neighbor cells. As the authors stated, “The simplest way of hexagonal grid mapping on the square one with the axis *j* extension is accepted here. It consists in the fact that one hexagon is set corresponded with a pair of pixels (*i*,*j*) and (*i*,*j*+1). These pairs in even and odd lines are shifted to each other. This method of image presentation makes the visualization process in cell array easier, since the back compression allows to put each cell in correspondence with one pixel on the monitor, its coloring power corresponds to the value under consideration”. And here we see again that the problems of the grids transition are solved with even and odd indices discrimination.

Hexagonal CAs can be used more efficiently in art images creation than ordinary CAs. The images in Figure 9 would be useful in the textile design in the ornament embroidery. We used similar totalistic 2D CAs to obtain a test video (via MCA SoftCAM tool) where the growth of snowflakes-like fractal structures (video1.avi) can be observed. Here again, as in the case of Figure 2, we resort to the trick of the arithmetic mean.

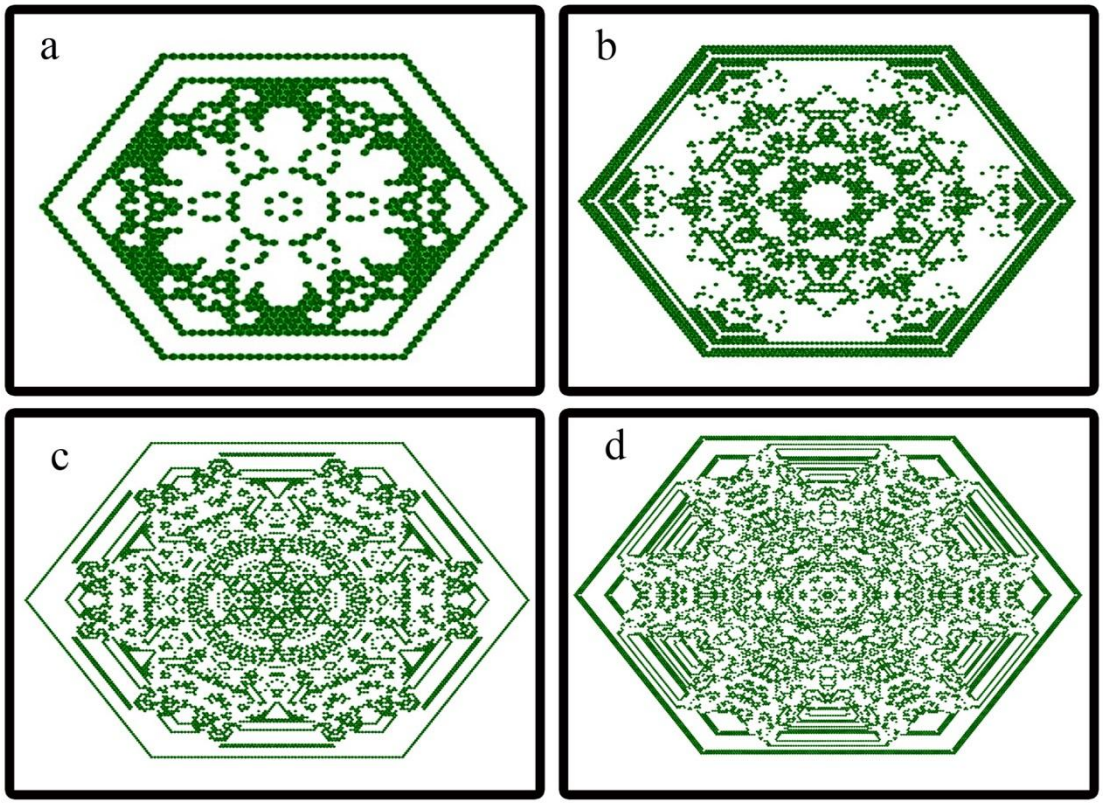


Figure 9. Hexagonal CA of totalistic type. The cell is not included in its neighborhood. If the number of live neighbors is 1 or 2, the cell goes into a "life" state, otherwise it dies. The initial state is a living cell in the center of the CA-field. a) - step 25; b) - step 50; c) - step 75; d) - step 100. The scales of drawings are different

The paper [27] shows, however, that a rectangular grid has not lost its importance in research. For example, the automata blockiness will be due to rectangularity of the mesh. The paper [42] considered block-synchronous and asynchronous CAs with neighborhood which can be detected with difficulty and may be dependent on the cell state; CAs simulate surface reaction of palladium oxidation with oxygen and carbon oxide. Difficulty of blocks specifying for the visualization is an interactive drawing of the neighborhood blocks/templates boundaries for the voxel chosen.

Further we shall describe the “isometric view” visualization method enriching our vision of 2D CA which is also applicable for 3D CA. Formally it applies to the partial one with 2D-voxel topology modification. If we consider Cartesian coordinate system O*xyz* in space, it divides the screen area into three parts: *xy*-slice with z=0, *xz*- slice with y=0, *yz*- slice with *x*=0. It is possible to project a section of data from space (*x*,*y*,*t*) on each area. For example, on one slice – current global configuration, on the other – last step configuration, and on the third slice – modulo 2 subtraction of two slices, i.e. for the slice cell 0 is set if the cell state has not changed, and 1 – if the cell state has changed (an analog to the activity ratio). Isometric view method can be applied for the case, when the cell state is complex, assume it is described with three components $\left(B_1, B_2, B_3\right)$, or if we deal with a partitioned CA – then each component is mapped on its slice.

We used just that very method to render 2D CA model of ion implantation on the hexagonal mesh. The length of every hexagon edge is 2.54nm (Figure 10), and the time quantum is 10^{-4}c. The case of moderate energy ion pulsed implantation (~ 50 keV) in silicon is simulated. The user simultaneously sees how many ions have already passed and their present distribution in the depth, and which ions continue to move, as well as the trajectories denoting the energy release in atomic collisions. Animation effects in the fields of 30×30 and 80×80 are provided respectively in the video2.avi and video3.avi files. Unlike the traditional methods of ion implantation modeling the usage of cellular automata allows to simulate the evolution of the film structure subjected to

bombardment as a whole, and to take into account the processes characterized with different time rates. In solving these problems the computer visualization plays an important role.

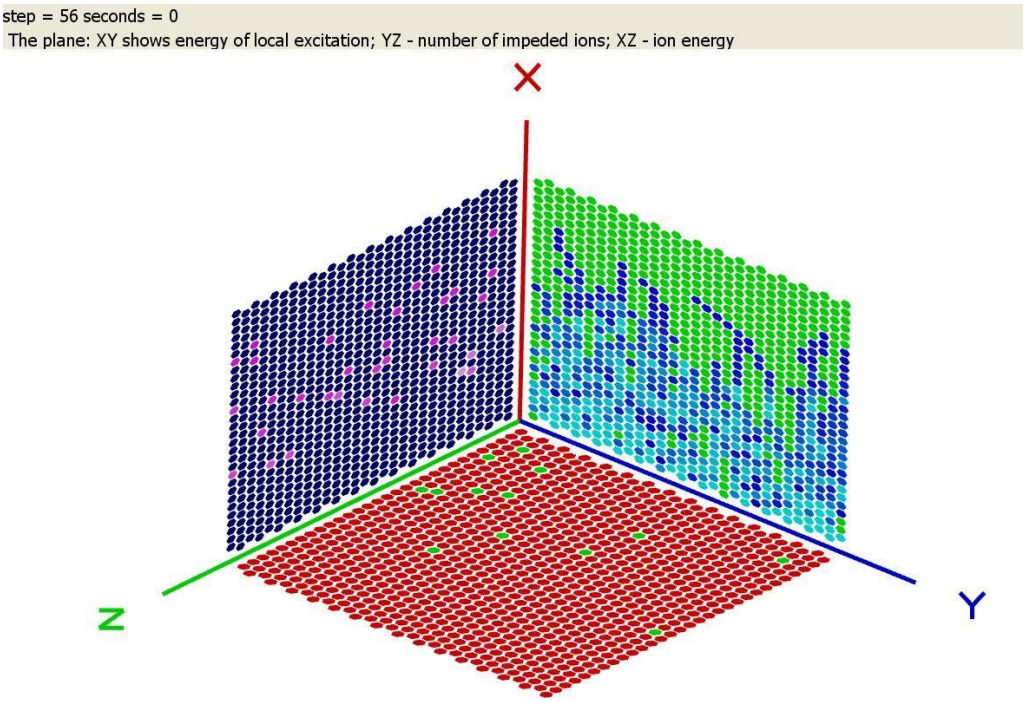


Figure 10. The fragment of ion implantation animation on the field of hexagons, 30 × 30 (step 56) large. In the XY plane lighter color corresponds to greater number of the excitation energy. In the first 10 steps a few particles are thrown at random on the CA-field

We should underline again that CA's two-dimensionality does not mean two-dimensionality of visualization. In 2007 a group of students of Warwick university produced a 3D simulator of 2D CA as a diploma work, and with a creative approach it is possible to get aesthetically valuable images [43]. On the website of the Mathematica project (S. Wolfram) there is a 3D–visualization of the GoL [44]: time is marked with the third dimension; smooth gradient of color reinforces transitions in time; voxel cell (and there is a mapping option of balls instead of cubes) is transparent if the cell is dead.

Methods of visualization of three-dimensional CA. 3D CA with locking

There are few CA-models, but it is them that are the most significant in physics and in nanoelectronics. Sometimes they come out as a result of generalization of 2D-models as, for example, if the case of Oono-Kohomoto's CA [4] for diffusion description [45] or the automata for simulation of the famous Belousov-Jabotinski reaction (The Project CA3D [46]). Nevertheless, 3D CA model of heart has been known in medicine for a long time already [47]. Assembly of a nanorobot consisting of 55 thousands 3D CA cells is presented in the paper [48]. Only after development of parallel computing 3D CA will be demanded in research, and now 3D CA is resource-intensive.

For nanotechnology a simulation of molecular assemblies both on solid surfaces and in the bulk material is of fundamental importance. CA serves as a natural language of description of such processes. Transition to 65 nm technology with less expenses, as well as development of nano- and microelectromechanical systems (N&MEMS) requires more attention to the processes of anisotropic etching (in the first case, the photoresist, and in the second – the sacrificial layers). 3D CA-models are known in these areas [49-51]. Much more works are devoted to the process of crystallization [52] and, generally, to the phase transitions; this generalization has lots of examples in nanotechnology, and pore formation presents one of them. Let us consider an old but excellent paper illustrating some specific issues that examines [53] melting tin cello, goblet of chocolate and lava from the unit point of view.

The movement of microvolumes was considered to be the result of force action of microgravity collecting, viscoelastic, and frictional forces. Three mechanisms of heating were considered: conduction, convection and radiation. The state of each of them was determined to be approximately 10^6 CA-cells with two Boolean and two real-type variables, the value of each of the latter lays in the range [0; 1] and they are responsible for the amount of liquid in a cell and the cell's potential to increase its volume. One of the real-type variables has linear relationship with the intensity of the color. The authors used the graphics accelerator VolumePro and additionally divided the voxel set into bricks/slice to speed up the animation (about 100 ms per move). The same authors studied visualization of chemical gardens, i.e. collection of plant-like structures that had been formed in aqueous solution of sodium silicate with soluble salts of metals [54]. It should be noted that modern DirectX/OpenGL support “volume texture” in the form of RGBA texture arrays as well as CLUT-tables (see Appendix) applied to the original scientific 3D data. This clears the way for MCA integration with high technologies of computer visualization systems.

A particular interest is a 3D surface CA, describing complex surfaces in space, for example, when modeling surface grains of polycrystalline materials or surface potentials of tertiary or quaternary protein structures. For simulation of copper corrosion S. Gobron used [55] the Voronoi diagram technique that sets division of the surface in the areas of active centers attraction. The originality of this work is that, on the basis of the Voronoi diagram a large polygonal mesh for rendering purposes was constructed, and then it was partitioned into small squares of the CA was made. The rendering function of the previous CA-computation was implemented this way. This again reminds us that CA-cells can be assigned not only to the space as in most cases (say, in models of lattice gas and hydrodynamics), but also to the substance. In the latter case the state variables can describe not only the three spatial coordinates, and the size/shape of the substance microvolume or a particle. This approach is evident in the concept of moving CAs proposed by S. G. Psakhie et al. The scientists simulated [56] the process of the fragile porous ceramics (ZrO_2) crack under mechanical stress. The presence of cracks in the sample means that two adjacent cells are not linked mechanically; non-standard rendering was that the cells themselves did not visualize, but it is gray short line at the junction of mechanically connected cells, and if the cells were separated, the color of this line was black.

There is a certain problem in 3D CA visualization: in a densely packed cube the outer layers hide the inner ones. We can headline four ways of solving this problem:

- Deleting (or transparence) of unnecessary voxels [57];
- Formation of sparseness, intervals between voxels [58];
- Use of slices (either as isometric view, or tied to the point [46]), Figure 11;
- Use of animated slices [59,60] and cuts (taken from the visualization methods in biology)

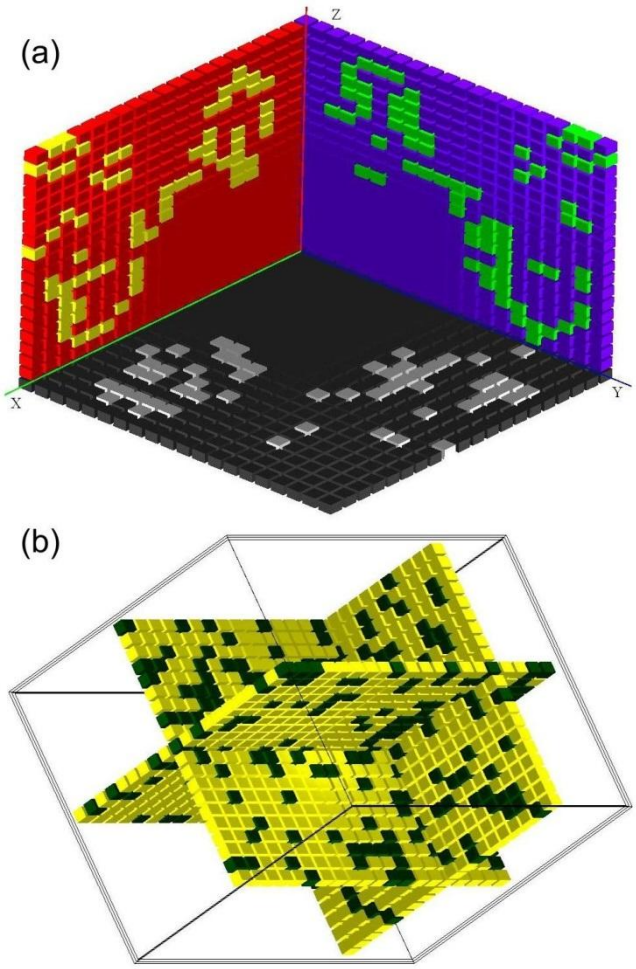


Figure 11. Isometric view method realized in MCA SoftCAM. (a): random configurations of GoL (on the wall *xz* – the state at the moment *t*–1, on the wall *zy* – the state at the *t*, on the floor *xy* – light voxels are the cells that changed the state on the current step). (b): three slices at a random point (*x*,*y*,*z*) of bit 3D CA

We consider below 3D CA, initially made for testing of MCA SoftCAM. This type of automata with rather simple transition rules has quite an interesting behavior resembling dynamics of crystallization or etching of substance. With more complicated rules it is possible to get more accurate and adequate, and physically richer in content, behavior of CA-model of this class. A similar idea, however, existing implicitly was presented by the authors [61] who considered crystallization of the material with tangled structure (i.e. in the sense of visualization with topology peculiarities). The simulation object was the hardening process of the two-component melt which is in a rectangular crucible (at the beginning the metal was liquid). During simulation of the crystallization process each cell was characterized with state (was free, growing, had hardened), had the scale and a certain feature of the figure that was grown (for example, the growing dendrite orientation or the label allowing to define associated areas). The cell that had just got the hardening state interacted with its neighborhood. When interacting with an active (that has just hardened) cell only not hardened cell were taken into account.

We shall consider our 3D CA in abstracto without giving it its physical meaning, and consider it as a representative of a new class of CA with locking. CA field is cubic, its size is 50×50×50. The cell state is given with a group of four $\langle \delta = \{0,1\}, r = \{0,\ldots,255\}, g = \{0,\ldots,255\}, b = \{0,\ldots,255\} \rangle$, spending 1 bit and 3 bytes. Neighborhood template is identified due to Moor's rule, i.e. 26 neighbors and the cell itself. As soon as $\delta = 1$, the cell is "frozen", and during the further computing does not change its state. Otherwise it has the following transition function (*i* runs all the neighbor cells):

$$\begin{cases} r(t+1) := [\max(r_i) + \max(g_i) - \max(b_i)] \\ g(t+1) := [g + (b - r)] \\ b(t+1) := [\min(b_i) + \min(g_i) - \min(r_i)] \end{cases},$$

$$\begin{cases} r(x,y,z)(t=0) = \left[c(1-2x)^2 \right] \\ g(x,y,z)(t=0) = \left[c(1-2y)^2 \right], \\ b(x,y,z)(t=0) = \left[c\sin(\pi z) \right] \end{cases} \quad \begin{array}{l} i \le I = 26, \\ c = 255 \\ 0 \le x, y, z \le 1 \end{array} \qquad (9)$$

$$\delta := 1 \Leftarrow \left\{ \exists i : \frac{r_i}{r} = \frac{g_i}{g} = \frac{b_i}{b} \right.$$

Values without indication of argument are taken at the moment *t*. The "square brackets" operation means rounding to the nearest integer; if the argument is more than 255 or less than 0, then it return 255 or 0, accordingly. The locking condition is maintained if in the neighborhood of cell there will be at least one cell with the same proportion of colors. The visualization model is 3D-voxel color-dependent and the colors are chosen according to the RGB-scheme (10):

$$R := r, \quad G := g, \quad B := b, \quad alpha = (1+\delta)/2 \quad (10)$$

Unlocked cells are rendered semi-transparently, and locked ones– fully (Figure 12).

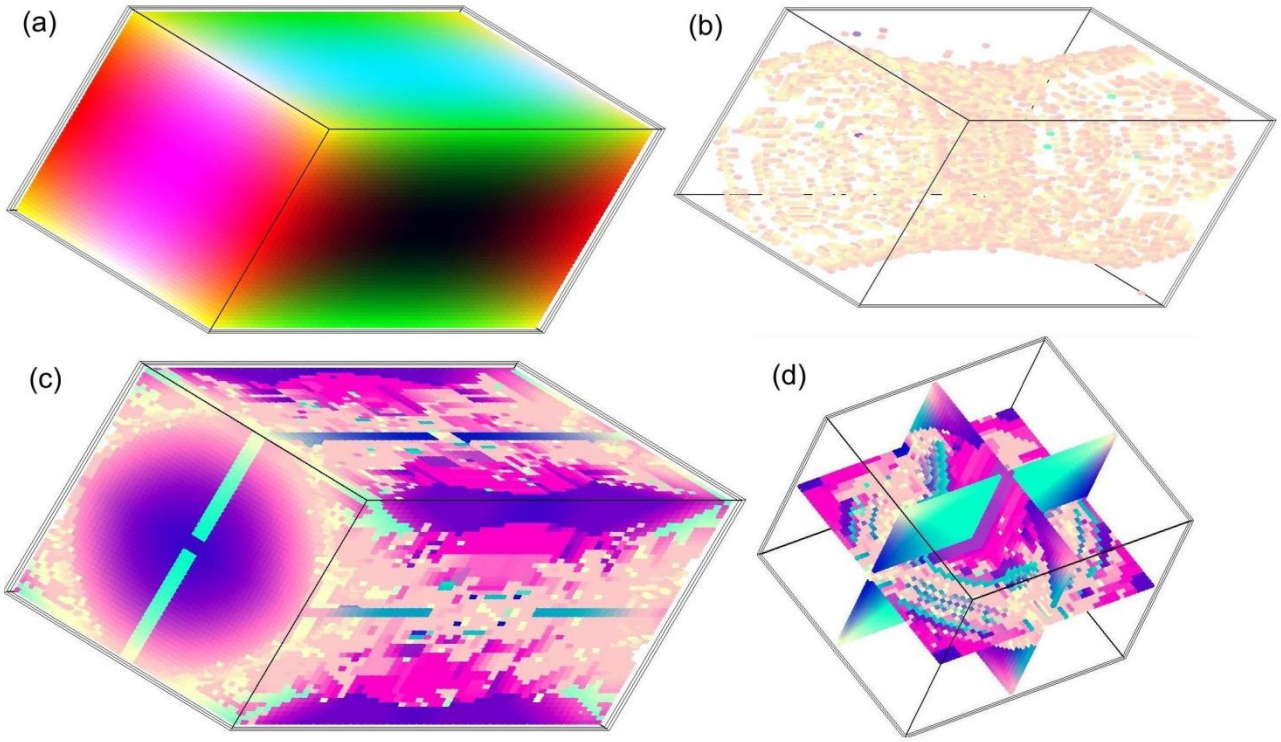


Figure 12. Evolution of 3D-CA with locking.(a) the initial state; (b) stationary state on the 25th step (locked cells are not shown); (c) stationary state on the 25th step; (d) slice view of the state on the 25th step

We shall speak about physics of cell state in terms of color, see (10). The initial state if it is chosen at random gives double-period oscillation in green after several global turns the average number of locked cells in asymptotic is ~ 83% (Figure 13). The CA initial state *t*=0 for each color is described by curves with one extremum; in (9) the values *x*,*y*,*z* are corresponded with cube edge indexation, and the constant *c* has technical value (see Appendix) of reduction *R*,*G*,*B* to the byte format. Approximately from the 16th iteration CA comes to two-period state with invariable number of locked cells and invariable, about 4 %, number of unlocked cells. This state, most

successfully mapped in Figure 12(b), has circular symmetry in regard to the axis O*x*, i.e. along the axis of change of red at the initial configuration, and the configuration in asymptotics looks like sand glass contour or slightly tightened tube, that can be interpreted as crystallization of substance, for example in spinning supersaturated solution or as a particular case of etching causing the pores ordered system.

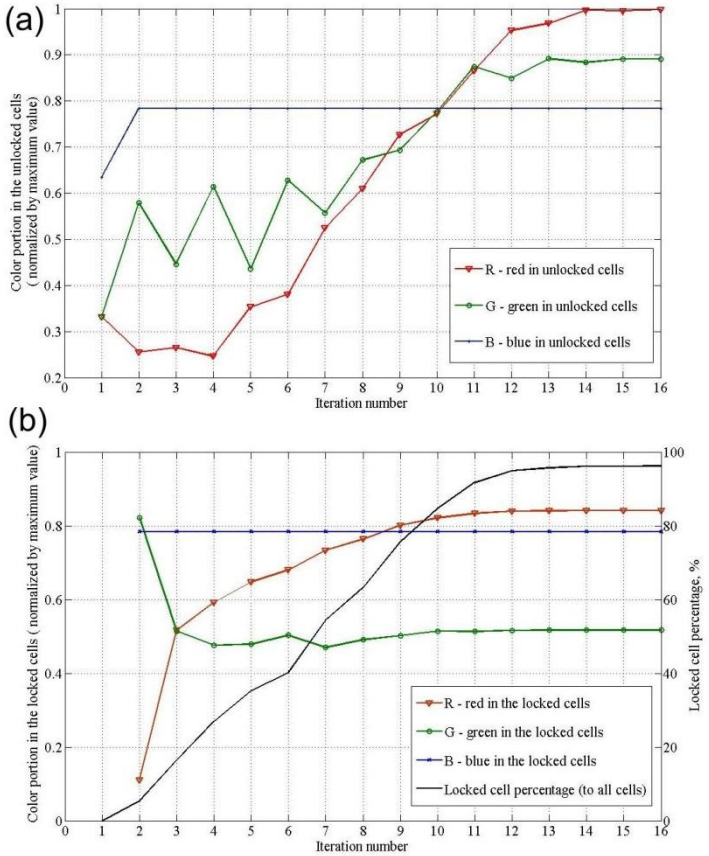


Figure 13. Example of intelligent consideration of data of 3D CA with locking

What is the idea of CA with locking? Adding the Boolean variable to the cell state of an ordinary CA, we can use it as a transition rule selector, and having specified by turn the transition rule for a Boolean variable, we can get directly in the process of CA-computing a variety of topologies that has, in particular, an application value for 3D-morphing. Difference in Boolean variable modifies the actual cell neighborhood template blocking the influence of the part of cells built into the initially specified template, common for CA cells.

The example of 3D CA with locking shows peculiarity of CA in nanoelectronics. The initial automata state contributes more (meaning that if we compare the influence of transition rules and initial configuration on CA evolution, subjectively the second factor can be evaluated as 50 %, while for ordinary "games" CAs the first factor dominates).

Conclusion

We made attempts and were the first to structure methods of cellular automata visualization. We considered the problems of visualization from the point of view of MCA, gave the general description of the CA visualization model, proposed the simplest methods of CA audization, focused on application of isometric view for 2D CA visualization, classified methods of solving the problems of 3D CA visualization.

General classification (Figure 3) of visualization remains valid for the CA-model in nanotechnology, some particular cases may need details. Specific problems of CA visualization in nanoelectronics are:

- CA cell state assignment complexity;
- Often, CA neighborhood template specifying complexity;

- Lack of fast-growing configurations beyond the boundaries of the finite field;
- Necessity of intelligent post- and real-time processing of scientific data (moreover CAs themselves can serve as one of the Data Mining methods [62]);
- Great importance of the initial configuring (even as compared to transition rules construction).

The paper does not concern the problems of low-level interaction of the visualizer, and relevance of some other CA visualization methods as regards to the certain research goals. We also didn't pay attention to the quantum cellular automata (QCA or QDCA) and questions of visualization, such as spintronics. Creative approach to visualization models in the long run will make modeling more efficient, and we hope this article will be useful for that.

Appendix. Colormap specification.

CA can generate mathematical structures that contain real numbers. Very often this real number X needs to have corresponding a triplet of real numbers (R,G,B), each within 0 to 1, giving the color decomposition in "red-green-blue". This correspondence is called a colormap. Sometimes there is an interim step, and X is normalized with its maximum value getting the indexed color value C in the range [0;1] in the format of 4-byte number with a floating point. In its turn, the RGB-color is coded with three bytes, i.e. the parameter, let's assume R, varies discretely within limits from 0 to 255.

It is rather difficult to fit an analytical function $f : C \to R \times G \times B$, that provides the desired properties. In practice to specify a colormap so called lookup tables are used. As for the desired properties our choice is as follow: a) the minimum value of X corresponds black and the maximum – white, i.e. $f(0) = (0,0,0)$, $f(1) = (1,1,1)$; b) pure rainbow colors, i.e. combinations like (0,0,1), (1,0,1), must be present in the table in natural order; c) the table should be one-to-one. These requirements simplify understanding of the visualized data, for example, meaning that blue is more than red. These requirements are roughly met by the transition sequence in RGB 000→100→110→010→011→001→101→111 along the variable C in the direction from 0 to 1. This sequence which we call a colormap A (see the Figure 14), may serve as an example of Gray's code modulo 3.

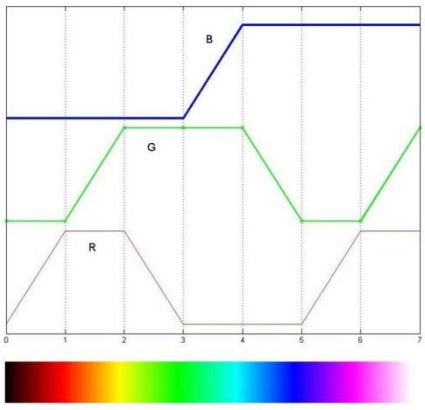


Figure 14. Colormap A: The scheme of RGB-channels changing in indexed color and rendering of the last

Another colormap B is based on simplicity of computing: the first byte of mantissa C gives R, the second – G, the third – B, and the fourth byte can be responsible for voxel transparence (alpha-channel). However, among the desired properties the second is broken because there is no natural color order.

References:

1. Toffoli Tommaso. *Cellular Automata Mechanics*. The Univ. of Michigan: Report. 208, Comp. Comm. Sci. Dept., 1977.

2. Pinto NN, Antunes AP. Cellular automata and urban studies: a literature survey. *Architecture, City and Environment* 2007; 1: 368-399. Link

3. Ilachinski A. *Cellular Automata: A Discrete Universe*. Singapure: World Scientific Publishing Co. Pte., 2002, p. 728.

4. Adamatzky A. Game of Life Cellular Automata. London: Springer, 2010, p.579.

5. Aladjev VZ. Classical Cellular Automata. Homogeneous Structures. Palo Alto: Fultus Corporation, 2010, p.478.

6. Talia D, Naumov L. Parallel Cellular Programming for Emergent Computation. In: Peter MA (eds) *Simulating Complex Systems by Cellular Automata*. Springer, 2010, pp. 357-384.

7. Hiebeler D. A brief review of CA packages. *Physica D* 1990: 45, 463-476.

8. Matyushkin IV, Khamukhin AV. UML usage in the design of cellular automata machines. *Izvestiya VUZov. Elektronika*, 2010: 86(6), 39-48. (in Russian) Link

9. Bandman OL. Using cellular automata for porous media simulation. *J. of Supercomputing*, 2011: 57(2), 121-131.

10. Wiebe EN. Scientific Visualization: A New Course Concept for Engineering Graphics. *Engineering Design Graphics Journal*, 1992: 56(1), 39-44. Link

11. Hopkins D. Fun with Cellular Automata, http://www.art.net/~hopkins/ Don/art/ cell.html (2012, accessed 2 april 2012)

12. Project «Automatous-monk», http://automatous-monk.com/ (2012, accessed 2 april 2012)

13. Project «CAPOW». http://www.cs.sjsu.edu/faculty/rucker/capow/examples.html (2012, accessed 2 april 2012)

14. Rucker R. Continuous-Valued Cellular Automata in Two Dimensions. In: Griffeath D (eds) *New Constructions in Cellular Automata*». Oxford University Press, 2003, p.342.

15. Kalgin K. Implementation of algorithms with a fine-grained parallelism on GPUs. *Numerical Analysis and Applications*, 2010: 4, 46-55. Link.

16. Kim E, Shen T, Huang X. A Parallel Cellular Automata with Label Priors for Interactive Brain Tumor Segmentation. In: *The 23rd IEEE International Symposium on Computer-Based Medical Systems* (ed S Antani), Perth, Australia, 12-15 October 2010, pp. 232-237. Link

17. Folino G, Mendicino G, Senatore A, et al. A model based on cellular automata for the parallel simulation of 3D unsaturated flow. *Parallel Computing* 2006: 32, 357-376. Link

18. Galaktionov V. Computer Graphics Techniques in CAD Applications. In: *12th Central European Seminar on Computer Graphics* (CESCG), Budmerice Castle, Slovakia, 24-26 April 2008, pp. 24-26.

19. Simulator Golly Homepage. - http://golly.sourceforge.net/

20. Veenhuis C, Köppen M. Document Oriented Modelling of Cellular Automata. In: *2nd International Conference on Hybrid Intelligent Systems* (HIS02), Santiago, Chile, 1 – 4 December 2002. Link.

21. Weimar JR. Translations of Cellular Automata for Efficient Simulation. *Complex Systems*, 2003: 14, 175–199. Link.

22. Vanag VK. Study of spatially extended dynamical systems using probabilistic cellular automata. *Physics-Uspekhi*, 1999: 42, 413–434. (in Russian) Link.

23. A periodic table of visualization methods. http://www.visual-literacy.org (2012 , accessed 2 april 2012) Link

24. Bowman WJ. *Graphic Communication*, New York: Wiley & Sons Inc, 1968, p. 222.

25. Kaurov V. *Wolfram Demonstrations Project: Coupled Cellular Automata: Symbiotic Patterns and Synchronization*. http://demonstrations.wolfram.com (2012, accessed 2 april 2012) Link

26. Séquin CH, Hamlin JF. The Regular 4-Dimensional 57-Cell. In: *The 34th International Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH), San Diego, California, USA, 5-9 August 2007, Link.

27. Medvedev Yu. The wall cells in the cellular automaton fluid flow simulation. *Bulletin of the Novosibirsk Computing Center. Series: Computer Science*, 2003: 19, 51-59. (in Russian)

28. Rucker R, Walker J. CelLab: User Guide: Rug and ASCII. http://www.fourmilab.ch/ (2012, accessed 2 april 2012) Link

29. Zoubko I, Keller I, Trusov P. Nonlocal model of dislocation pattern formation in crystal in terms of cellular automata. *Phys. Mesomech*, 1999: 2, 17-25. (in Russian)

30. Site of Stewart Dickson on portal http://emsh.calarts.edu/. (2012, accessed 2 april 2012) Link

31. Sudhanshu KS, Chandrashekhar K. Cellular Automata for 3D Morphing of Volume Data. In: *The 13th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, Plzen, Czech Republic, 31 January – 4 February, 2005, p. 195-202.

32. Wolfram Stephen. *A New Kind of Science*. Wolfram Media, 2002, p. 1197.

33. Zenkin AA. Waring's problem from the standpoint of the cognitive interactive computer graphics. *Mathematical and Computer Modelling*, 1990: 13, 9-25.

34. Cellular automata music on Novation Launchpad (movie). http://video.ariom.ru/v/rWq4AppMm8A.html (2012, accessed 2 april 2012)

35. Miranda ER. *Evolving Cellular Automata Music: From Sound Synthesis to Composition*. In: European Conference on Artificial Life ECAL, Prague, Czech Republic, September 2001. Link

36. Burraston D, Edmonds E, Livingstone D, et al. *Cellular Automata in MIDI based Computer Music*. In: International Computer Music Conference, Miami, USA, 2004, Link

37. CAMUS: Cellular Automata music generator. http://tamw.atari-users.net/camus.htm (2012, accessed 2 april 2012)

38. Otomata: Cellular Automata Music Generator. http://www.earslap.com/projectslab /otomata (2012, accessed 2 april 2012)

39. Audization CA Brian's brain. http://jmge.net/brain.htm (2012, accessed 2 april 2012)

40. Afanasyev IV. Research of evolution of cellular automata modeling «phase separation» process on triangular mesh. *Prikl. Diskr. Mat*, 2010: 4, 79–90. (in Russian) Link.

41. Bandman OL, Kinelovsky SA. Cumulative synthesis: a cellular-automata model of physical-chemical processes on the stage of powder revetment collapsing. *Prikl. Diskr. Mat.*, 2011: 2, 113–124. (in Russian) Link. See also: Bandman OL. A cellular automata convection-diffusion model of flows through porous media. *Optoelectronics, Instrumentation and Data Processing* 2007: 43(6), 524-529.

42. Markova VP, Sharifulina AE. Parallel implementation of asynchronous cellular automata for modeling CO oxidation over palladium surface. *Prikl. Diskr. Mat.*, 2011: 1, 116–126 (in Russian). Link.

43. Project "Cellulate". http://cellulate.sourceforge.net/ (2012, accessed 2 april 2012)

44. Demonstration of CA. http://demonstrations.wolfram.com/GameOfLifeIn3DLayers/; http://webmath.exponenta.ru/ad/aj/dem/020401.html (2012, accessed 2 april 2012)

45. Weimar JR. Three-dimensional Cellular Automata for Reaction-Diffusion Systems. *Fundamenta Informaticae*, 2002: 52, 277-284. Link.

46. Project CA3D. http://www.ctechnet.com/ca3d/Gallary.html#6 (2012, accessed 2 april 2012)

47. Siregar P, Sinteff JP, Julen N, Le Beux P. An Interactive 3D Anisotropic Cellular Automata Model of the Heart. *Computers and Biomedical Research*, 1998: 31, 323-347.

48. Huw D, McWilliam R, Purvis A. Design of Self-Assembling, Self-Repairing 3D Irregular Cellular Automata. In: Salcido A (ed) *Cellular Automata - Innovative Modelling for Science and Engineering*, InTech, 2011, p.436. Link.

49. Zai-Fa Zhou, Qing-An Huang, Wei-Huan Li, et al. 3D photoresist etching simulation using cellular automata. In: Conference of the International society for optics and photonics (SPIE), Florida, USA, 2006, 6034, 60340Q.

50. Zai-fa Zhou, Qing-an Huang, Wei-hua Li, et al. A cellular automaton-based simulator for silicon anisotropic etching processes considering high index planes, *J. Micromech. Microeng.* 2007: 17, S38.

51. Gosálvez MA, Xing Y, Sato K, Nieminen RM. Discrete and continuous cellular automata for the simulation of propagating surfaces. *Sensors and Actuators A*, 2009: 155, 98-112.

52. Svyetlichnyy DS, Matachowski JL. Three-Dimensional Cellular Automata For Simulation Of Microstructure Evolution During Recrystallization. In: Cesar De Sa (ed) *The 9th International Conference on Numerical Methods in Industrial Forming Processes* (NUMIFORM), Porto, 17-20 June 2007 (American Institute of Physics, Conf. Proc., vol. 908, pp. 1357-1362).

53. Xiaoming Wei, Wei Li and Kaufman A. Melting and Flowing of Viscous Volumes. In: *the 16th International Conference on Computer Animation and Social Agents* (CASA'2003) New Jersey, USA, 7-9 May 2003, pp. 54-59. Link

54. Xiaoming Wei, Feng Qiu, Wei Li, et al. Visual simulation of chemical gardens. In: *Conference of Computer Graphics International* (CGI), New York, USA, 22-24 June 2005, pp. 74-81. Link

55. Gobron S. and Chiba N. 3D Surface Cellular Automata and Their Applications. *J. of Visualization and Computer Animation*, 1999: 10, 143–158.

56. Smolin AY, Roman NV, Loginova DS, et al. Influence of Porosity Percolation on Mechanical Properties of Ceramic Materials. 3D Simulation using Movable Cellular Automata. In: DRJ Owen (ed) *2th Int. Conf. on Particle-Based Methods* (Particles'2011), Berlin, Germany, 9-12 July 2011, pp. 249-255, Link

57. Movie. http://www.youtube.com/watch?v=CYgbnI_R1E0 (2012, accessed 2 april 2012)

58. Graw F, Regoes RR. Investigating CTL Mediated Killing with a 3D Cellular Automaton. *PLoS Comput. Biol.* 2009: 5(8), e000466. Link

59. Project "Cellulate". http://cellulate.sourceforge.net/ (2012, accessed 2 april 2012)

60. Matyushkin IV, Korobov SV. Cellular automata data visualization for the issue of nc-Si growth within the SiO_x matrix. *Izvestiya VUZov. Elektronika* 2011: 92, 39-48. (in Russian)

61. Belankov AB, Stolbov VYu. Application of cellular automata to modeling the microstructure of a material under crystallization. *Sib. Zh. Ind. Mat.* 2005: 8, p. 12–19. (in Russian)

62. Fawcett T. Data mining with cellular automata. *SIGKDD Explorations* 2008: 10, 32-39. Link

УДК 004;519;621

Особенности визуализации клеточных автоматов в области наноэлектроники

[1] Геннадий Яковлевич Красников
[2] Игорь Валерьевич Матюшкин
[3] Сергей Владимирович Коробов

ОАО «НИИ молекулярной электроники», Российская Федерация
124460, г. Москва, Зеленоград, 1-й Западный проезд, д. 12/1
[1] Генеральный директор ОАО «НИИМЭ и Микрон», академик РАН
[2] Начальник лаборатории отдела функциональной электроники, к.ф.-м.н.
E-mail: imatyushkin@sitronics.com
[3] Младший научный сотрудник, аспирант

Аннотация. Представлена формализация моделей визуализации клеточных автоматов (КА), рассмотрена их классификация. Также описаны возможные подходы к генерации звукорядов. Приведены частные случаи вариантов визуализации для КА различной размерности. На примере простого 3D КА указаны особенности визуализации наноразмерных систем.

Ключевые слова: клеточные автоматы; визуализация; нанотехнология; моделирование; мультимедиа.