# SYNTHESIS OF IMAGES BASED ON CELLULAR AUTOMATA

*Belarusian State University of Informatics and Radioelectronics*

*Losyukov L.N, Zhao Qihui*

*Supervisor – Prof Kanapleka V.K.*

A cellular automaton (pl. cellular automata, abbrev. CA) is a discrete model studied in computability theory, mathematics, physics, complexity science, theoretical biology and microstructure modeling. Cellular automata are also called cellular spaces, tessellation automata, homogeneous structures, cellular structures, tessellation structures, and iterative arrays.

A cellular automaton consists of a regular grid of cells, each in one of a finite number of states, such as on and off (in contrast to a coupled map lattice). The grid can be in any finite number of dimensions. For each cell, a set of cells called its neighborhood is defined relative to the specified cell. An initial state (time t = 0) is selected by assigning a state for each cell. A new generation is created (advancing t by 1), according to some fixed rule (generally, a mathematical function) that determines the new state of each cell in terms of the current state of the cell and the states of the cells in its neighborhood. Typically, the rule for updating the state of cells is the same for each cell and does not change over time, and is applied to the whole grid simultaneously, though exceptions are known, such as the stochastic cellular automaton and asynchronous cellular automaton.

**Applications**

1. Computer processors. Cellular automaton processors are physical implementations of CA concepts, which can process information computationally. Processing elements are arranged in a regular grid of identical cells. The grid is usually a square tiling, or tessellation, of two or three dimensions; other tilings are possible, but not yet used. Cell states are determined only by interactions with adjacent neighbor cells. No means exists to communicate directly with cells farther away. One such cellular automaton processor array configuration is the systolic array. Cell interaction can be via electric charge, magnetism, vibration (phonons at quantum scales), or any other physically useful means. This can be done in several ways so no wires are needed between any elements. This is very unlike processors used in most computers today, von Neumann designs, which are divided into sections with elements that can communicate with distant elements over wires.

2. Cryptography. Rule 30 was originally suggested as a possible block cipher for use in cryptography. Two dimensional cellular automata are used for random number generation. Cellular automata have been proposed for public key cryptography. The one-way function is the evolution of a finite CA whose inverse is believed to be hard to find. Given the rule, anyone can easily calculate future states, but it appears to be very difficult to calculate previous states.

3. Error correction coding. CA have been applied to design error correction codes in a paper by D. Roy Chowdhury, S. Basu, I. Sen Gupta, and P. Pal Chaudhuri. The paper defines a new scheme of building single bit error correction and double bit error detection (SEC-DED) codes using CA, and also reports a fast hardware decoder for the code.

**Description of the mathematical model**

The cellular automata is a mathematical model of time - space discrete and state - discrete. The general cellular automaton is a quaternion A=(D, S, N, F). D represents the dimension of A, S is a finite state set. N is a primitive vector of n different elements of $z^d$, $N=(x_1, x_2, \ldots, x_n)$. F is the local programming of the state function of the cellular automata $CA_n$. The cells are arranged in a finite (infinite) d-dimensional array, and the position is indexed by $z^d$.

D is the dimension of the cellular automaton. Cellular automata have one-dimensional, two-dimensional, three-dimensional and multidimensional models. One-dimensional cellular automata model is to divide the lines into equal parts, each of which represents a cell; The two-dimensional cellular automata model divides the plane into many square, triangular or hexagonal meshes, which represent the corresponding cells; The three-dimensional cellular automata model divides the space into many three-dimensional grids. With the increase in the number of bits, the phenomenon of cellular automata will be more complicated.

S is a finite state set, The cellular automata characterizes the main features of the simulated system with a finite number of states. When using cell automata to simulate any system, the main features of the real system, the law is extracted, the time is divided into a series of discrete moments, the space according to different dimensions to choose different rules of the grid or grid, In the case of a simple system, the cell automaton can take 0 or 1 for each grid's state value. That is, only a finite state set, S=｛0,1｝, The complex state of the state can take the state set of multiple values k, S=｛0,1…,k｝. In the generation of the cell automata pattern, the pattern main feature is a color, and the number of colors of the generated pattern is represented by the number of states.

N is a primitive vector of n different elements of $Z^d$. The updating of a cell state value of a cellular automaton is determined by itself and its surrounding n cell states, called n-neighborhood cellular automata. In general, the position $C_{ij}$ of the neighborhood of the two - dimensional cellular automata is satisfied: ｛$C_{ij}$:|u-i|＜＝r;|1-j|＜＝r｝. R is the neighborhood radius. I, j for the cell line, column position, U, l is the position of the neighborhood up and down.

F gives a local rule for a cell to derive a new state based on the state of its nearest neighbor. The update rules depend on the qualitative understanding of the system's macro-processes and real physical mechanisms, linear rules, and non-linear rules.

Assuming that at time t, the configuration of one-dimensional cellular automata is $a^{(t)}$, then at time t+1, the configuration of cellular automata is $a^{(t+1)}$, then

$$A_i^{(t+1)}=f(a_{i-r}^{(t)}, \dots , a_{i-1}^{(t)}, a_i^{(t)}, a_{i+1}^{(t)}, \dots , a_{i+r}^{(t)})$$

among them, the mapping f is a local mapping or local rule of the previously mentioned cellular automata. Independent of i and t. R is the neighborhood radius or spatial scale of the one-dimensional cellular automata. So it can be expressed as a general mathematical form:

$$a_i^{(t+i)} = \int [ \sum_{j=-r}^{j=r} \mathrm{aj} \quad \mathrm{ai} + \mathrm{j}^{(t)}$$

One-dimensional cellular automata can be determined only in the simplest case by the cell state on the left and right adjacent lattice points. Often assumed to be consistent with Boolean dynamics, f is determined by the modulo-2 addition value of the two cell states of $a_{i-1}$ and $a_{i+1}$, at this time, weight $\alpha_j$=1. So at the time t+1, The state of cell $a_i$ is

$$A_i^{(t+)} = f(a_{i-1}^{(t)} , a_{i+1}^{(t)}) = a_{i+1}^{(t)} ð a_{i+1}^{(t)}$$

In contrast to one-dimensional cellular automata, the configuration of two-dimensional cellular automata is more complex.

Suppose that at time t, the configuration of the two-dimensional cellular automata is $a^{(t)}$, the number of states is k, and the neighborhood radius is r. Then the configuration of the cell automaton at time t + 1 is $a^{(t+1)}$, where the cell state value at position $a_{i,j}^{(t+1)}$ is:

$$A_{i,j}^{(t+1)} = f(a_{i-r,j}^{(t)} , \dots , a_{i-1,j}^{(t)} , a_{i+1,j}^{(t)} , \dots ,a_{i+t,j}^{(t)} , a_{i,j}^{(t)} , a_{i,j-r}^{(t)}, \dots , a_{i,j-1}^{(t)} , a_{i,j+1}^{(t)} , \dots , a_{i,j+r}^{(t)}) \bmod k$$

List of sources used:

1. S. Amoroso; Y.N. Patt. Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures. Journal of Computer and System Sciences. October 1972, 6 (5): 448–464.

2. Joel L. Schiff. Cellular Automata: A Discrete View of the World. Wiley & Sons, Inc. – 2011.