From pseudo-random numbers to stochastic growth models and texture images

Leonid P. Yaroslavsky

From pseudo-random numbers to stochastic growth models and texture images

Leonid P. Yaroslavsky
Centre Emil Borel – UMS 839 (CNRS/UPMC)-Paris
and
Department of Interdisciplinary Studies,
Faculty of Engineering, University of Tel Aviv,
Tel Aviv 69978, Israel

Pseudo-random number generators, stochastic evolutionary growth models and texture images are treated in a unified way in terms of dynamic systems with nonlinear feedback and their evolutionary behavior.

1 Introduction

The problem of describing and modeling formation of stochastic patterns is addressed. Two kinds of patterns are dealt with in the paper: patterns that simulate processes of growth in biology and similar processes in material sciences, and images that are studied in image processing, computer vision and computer graphics and are conventionally called textures. Traditionally, two completely different methodological approaches are applied to treat these two kinds of pattern formation problems. While simulating growth processes is naturally based on phenomenological modeling and imitating biological and similar evolution processes involved, in texture synthesis and analysis attempts prevail to customize techniques of the theory of random processes that originate from mathematical models in statistical physics and communication theory.

Both approaches suffer from a certain “one-sidedness”. Phenomenological growth models frequently ignore stochastic nature of growth processes and growing formations. Random process theory based texture image models and texture analysis methods that have been a subject of intensive research efforts for many years ignore physical meaning of imaging and images, the fact that texture images as any other images are reflections of objects of the real physical word and therefore carry a dint of real processes that generate and govern those objects.

The paper is intended to show that, in fact, there is no any substantial difference between textures and growing formations and that they can be treated in a unified way. Basic points of this unified approach are:

- Both texture images and growing patterns can be modeled in the same way in terms of dynamic systems with nonlinear feedback and their evolutionary behavior.

---

1 On leave from Institute of Information Transmission Problems, Russian Academy of Sciences, 101447, B. Karetny, 19, Moscow, Russia
Both texture images and growing patterns are stochastic. This means that they are controlled by a set of parameters that define only an ensemble of the patterns of the same kind rather than each individual one. Each individual pattern is obtained as a result of a “random” choice from the ensemble.

The process of the “random” choice can itself be regarded as a zero-dimensional growth (evolution of a single variable) and modeled by generating pseudo-random numbers conventionally used in Monte-Carlo computer simulations.

The exposition is arranged in the following way. In Section 2 two modifications of generators of pseudo-random numbers are represented in terms of dynamic systems with nonlinear feedback and described by schematic diagrams. In Section 3 it is shown that rather simple modifications of these schematic diagrams give rise to a family of stochastic growth models that are illustrated by Eden’s type models and by several modifications of evolutionary models that originate from Conway’s “Game of Life”. Section 4 is devoted to texture image modeling. It demonstrates how texture images can be generated and treated as a product of evolution of dynamic systems and illustrates several concrete examples of such systems of different complexity. In conclusion, a short glossary is provided of terms used in the paper.

2 Pseudo-random number generators

First generators of pseudo-random numbers were suggested by Von Neuman at the very beginning of the computer era. Since then many attempts have been undertaken to improve “randomness” of the generated numbers including even attempts to introduce hardware random number generators that exploit “random” nature phenomena such as radioactivity or brownian motion. Finally, software generators have become commonly accepted for generating numbers that seem “random” in particular applications. Since these numbers are generated in a complete deterministic way they are called “pseudo-random” which reflects a faith that true random numbers do exist. Usually, these generators produce pseudo-random numbers recursively from one initial number by using relatively simple computational rules. For instance, Knuth recommends an algorithm that can be described by the following recursive relationship

$$\xi(t) = (C_1 \xi(t-1) + C_2) \mod C_3$$

(1)
Fig. 1 Schematic diagram of a pseudo-random number generator. The graph shows transfer function of the point-wise nonlinearity unit.

Fig. 2 A modification of the pseudo-random number generator with a linear filter in the feedback (a) and examples of an initial image (b, left) and generated images after one (b, center) and 10 (b, right) iterations.
Here $\xi^{(t)}$ is a pseudo-random number generated at $t$-th iteration, $C_1$, $C_2$, and $C_3$ are certain constants, $\left(\right)\text{mod }C_3$ is an operation of finding residual of division of the input value by $C_3$.

The algorithm generates, one by one, pseudo-random numbers with uniform distribution density in the range [0,1]. The very first number is called “seed” and has to be set in advance. A schematic diagram of the algorithm is shown in Fig. 1. Represented in this way, the algorithm assumes using the following processing units: a multiplication unit, a summation unit, a point-wise nonlinearity that implements operation $\left(\right)\text{mod }C$ (its transfer function is shown in Fig. 1), and a one sample delay unit. The latter is a very important component of the scheme that makes it iterative, or, in another word, evolutionary.

This scheme represents an example of a very simple nonlinear dynamic system. It is well known now that such systems potentially may exhibit cycles and fixed points in the process of iterations (system evolution). A common requirement to the pseudo-random number generators is that they should avoid cycles and fixed points and provide numbers with nearly uniform distribution and without noticeable correlations. In practice it is achieved by a careful selection of the model parameters $C_1$, $C_2$, and $C_3$.

The above scheme is obviously a simplified version of a more general one presented in Fig. 2. Multiplication and summation units in the scheme of Fig. 1 are replaced here by a linear filter, a device that computes output signal by weighted summation of certain number of input samples, the weights being defined by the filter impulse response (point spread function), and one-sample delay unit is replaced by a one-frame delay unit where frame is a certain group of samples. If the signal samples in this scheme are arranged in a form of a rectangular array they can be displayed as an image. Fig. 2b illustrates an example of evolution of an image taken as a “seed”. The linear filter in this example is a simple two-dimensional “box” filter with a uniform 3x3 samples impulse response. Such a filter computes, for each image sample (pixel), image local mean value over the window of 3x3 pixels centered at this sample. A constant $C_3$ in the point-wise nonlinearity was set equal to the half of the image maximal gray level. One can see on this image how the nonlinearity and feedback destroy in only a few iterations all pixel correlations that existed in the initial image.

3. Growth models

Turing (1952) has demonstrated by mathematical methods that certain types of dynamic system which are initially homogeneous undergo a progressive change which leads to the appearance of spatial heterogeneity.

J.W.S. Pringle, F.R.S.³

3.1 Eden’s type growth models

Stochastic growth models aimed at simulating biological grows have been studied since very first years when digital computers became available³⁴. One of the first models was suggested by M. Eden⁵. In Eden’s model, growth was simulated as a
sequence of random “births” taking place on a rectangular lattice (raster) with the probability proportional to the number of already “live” cells in the nearest spatial 3x3 vicinity of the given cell (left and right neighbors at the same row, 3 and 3 neighbors on the rows from above and from bottom). Eden’s model can be mathematically represented as an iterative equation:

$$output(k,l)^{(t)} = \text{randb}(S_{k,l}^{(t-1)}/8) \oplus output(k,l)^{(t-1)}$$

(2)

where \((k,l)\) are pixel coordinates on the lattice, \(S_{k,l}^{(t-1)}\) is the sum of pixel values in 8 neighbor points in the 3x3 neighborhood of the given pixel, \(t\) is iteration index, \(\text{randb}(P)\) is a binary random variable that takes value of one with probability \(P\) and \(\oplus\) denotes modulo 2 addition of binary numbers. Fig. 3a shows how this model can be implemented in a system that is just a slightly modified and extended version of the system of Fig. 2. This system contains, as an individual unit, a pseudo-random number generator of Fig. 1 which is now included in the same loop of a linear filter, point-wise nonlinearity and a delay unit. Impulse response of the linear filter and transfer function of the point-wise nonlinearity are shown in the corresponding boxes in Fig. 3. The combination of the pseudo-random number generator and the point-wise nonlinearity with a threshold transfer function forms a unit that implements an operation \(\text{randb}(P)\) of generating binary numbers 0-s and 1-s with a given probability of 1-s. The model assumes that each cell (pixel) has one of two possible states, zero and one. On binary images, the linear filter with impulse response as shown in Fig. 3a computes the number of 1-s in the 3x3 neighborhood (8-neighbor sum \(S_{k,l}^{(t)}\)) of each pixel thus defining the threshold level of the point-wise nonlinearity.

Clearly, this simple model describes unlimited growth. One can, however, easily modify this model to simulate drain of “sources of food” by measuring the size of the growing formation and introducing a corresponding saturation to the probability of “birth” as it is shown in schematic diagram of Fig. 3b. In this scheme, the \(\text{randb}(P)\) unit of Fig. 3a is preceded by a \((x^{-y})\) - point-wise nonlinearity that implements the saturation. This modified model can be described by equation

$$output(k,l)' = \text{randb}\left(\left(S_{k,l}^{(t-1)}\right)^{(y)} / 8\right) + output(k,l)^{(t-1)}$$

(3)

where \(S_{k,l}^{(t-1)}\) is a “global” sum over the entire field that defines the size of the formation on \((t-1)\)-th iteration (evolution) step.

If the saturation is introduced to all probabilities but to the probability of “birth” from only one neighboring live cell one arrives at a modification of the model which begins growing dendrites after (statistically) the cell reaches a certain size. Figs. 3c) and d) illustrate the work of these models. Images are displayed here in color that
Fig. 3 a,b  Schematic diagram of two modifications of Eden’s model. The tables in boxes “Linear filter” present the linear filter impulse response. The graph in the box “Point-wise nonlinearity” in a) shows the nonlinearity transfer function.
Fig. 3 c, d. Examples images generated by the Eden’s model with saturation (c) and its modification that evolves into growing dendrites (c).
corresponds to the “age” of each pixel (number of evolution steps from its birth). Color bars show “age”-to-color coding scale. Other modifications of the model aimed, for instance, at imitating dependence of growth from “age” of cells are also more or less straightforward.

3.2 Conway’s “Game of Life” and its modifications

A mathematical model known as Conway’s “Game of Life” represents yet another type of growth models where cells on a rectangle lattice (raster) can give a “births” or “die out” depending on the number of “alive” and empty cells in their nearest spatial neighborhood. The rules of the original “Game of life” are very simple: if an empty cell has exactly 3 “alive” neighbor cells in its 3x3 neighborhood, birth takes place on the next step of the evolution; if an “alive” cell has less than 2 and more than 3 “alive” cells in the neighborhood it will die on the next step; otherwise nothing happens. These rules can be formally described by the equation:

$$\delta(t_{i+1}) = \left[ \delta(t_i) \right] \left[ \delta \left( S_{i+1} - 2 \right) + \delta \left( S_{i+1} - 3 \right) \right]$$

where \( \delta(x) \) is Kronecker delta \( \delta(0) = 1; \delta(x \neq 0) = 0 \), \( S_{i+1} \) is the sum of the values in 8-neighborhood of \( (k,l) \)-th pixel ("cell" in the growth model terminology), \( t \) is iteration number and “alive” and “empty” cells are represented by “ones” and “zeros”, respectively. Original model assumed a deterministic initial distribution of zeros and ones in the field. By introducing “random” initial distribution of “alive” and empty cells the model can be made stochastic. The corresponding schematic diagram of this model is shown in Fig. 4 a. One can see that this diagram contains essentially the same units as Eden’s model but in 2 parallel branches (one for “births” and one for “deaths”), and the \( \text{randb}(P) \) generator of the Eden’s model is placed at the input of the model and is used for generating “random” initial distribution of 1’s and 0’s for “alive” and empty cells.

Evolutionary behavior of the model is very interesting. The model generates different types of formations: stable formations that once appeared keep staying unchanged unless they are destroyed by other formations; growing crystal-like formations that grow until their fragments form stable formations or die out; cyclic formation that repeat themselves with a certain period; moving formations also featuring iteration-wise cycles (“gliders”). Boundary conditions of the model are important for its evolution. Under pseudo-random boundary conditions, when pseudo-random binary numbers are permanently generated at the borders of the field the model generates patterns that do not converge to a fixed (stable) ones though always contain certain number of formations that “live” during considerably large number of iterations (evolution steps). The pattern evolution is illustrated in Fig. 4 b-g. One can see on these images randomly placed stable formations such as 2x2 pixel square blocks and hexagonal formations called beehives, formations that grow like crystals, and “gliders”
that move across the lattice with a period of 4 evolution steps (at the right upper corner of south-west quarter of images), etc.

An important component of the model is the direction of the spatial interaction. It is defined by the linear filter impulse response. In the original Conway’s model, the spatial interaction is isotropic: all cell’s 8 neighbors play the same role in the defining next state of the cell on each iteration step. In the model of Fig. 4 a), this is reflected in the linear filter isotropic impulse response equal to 1 for all 8 neighbor pixels. In general, the filter impulse response may be anisotropic. In particular, it may define only one-dimensional interaction (only left and right neighbors of each cell affect its next state) thus producing one-dimensional models. An interesting special case of such a 1-D model is one described by the equation:

Fig. 4 a) - Schematic diagram of Conway’s model. Tables and graphs in boxes show impulse response and transfer functions of the corresponding units.
Fig. 4 b-g. Evolution of pattern b) in the Game of Life model. On images c-f one can see how different formations evolve. In particular, in right upper corner of the left bottom quarter of each image two “gliders” are seen that move with a period of 4 iterations.
Fig. 5 Evolution (downward in vertical direction) of a one-dimensional (in horizontal direction) modification (Eq. 5) of the Game of Life (initial rate of white points in the first row was 0.3)

\[
\text{output}(k,l) = \text{output}(k,l^{-1}) \delta(S_2) \oplus \delta(S_2 - 1),
\]

(5)

where \( S_2 \) is the sum over 2 neighbor cells of the \( k \)-th cell (from the left and from the right). Fig. 5 shows row by row an example of the evolutionary behavior of such a one-dimensional model. It is interesting to observe that patterns that appear in the process of the evolution remind the so-called Sierpinski Gasket and those patterns that some see-shells develop in their life.

One can further modify this model by introducing random “death” and “birth” events:

\[
\text{output}(k,l) = \text{randb}(P_d) \cdot \text{output}(k,l^{-1}) \delta(S_2^{-1} - 2) + \text{randb}(P_b) \delta(S_2^{-1} - 3),
\]

(6)

where \( \text{randb}(P_d) \) and \( \text{randb}(P_b) \) are the same binary pseudo-random number generators as in Eden’s model (Eq. 2) that produce “ones” with probabilities \( P_d \) (probability of “death”) and \( P_b \) (probability of “birth”), respectively. In the initial Conway’s model \( P_d = P_b = 1 \). If \( P_d < 1 \), the evolutionary behavior of this model radically changes. It begins to produce labyrinth-alike formations with irregular dislocation whose positions depend on the realization of the initial pattern. While the “body” of the patterns stabilizes after a few iterations their periphery continue growing independently until the pattern fills the entire lattice. Depending of the probability of
Fig. 6 Examples of the modified Conway’s model evolution with $P_d = 0.25$ and $P_s = 1$ (a) and e) - initial binary patterns; b), c), d, and f – corresponding evolution results).
“1s” in the initial pattern it may happen that several such formations arise and start growing until they merge into one larger labyrinth alike formation. An example of such an evolution is shown in Fig. 6. One can see that most cells in the initial pattern (Fig. 6, a) “die out” after a few iterations. There are however some “lucky” combinations of cells that survive and give rise to growing formations (Fig. 6, b) that grow until they merge (Fig. 6c) and finally fill the field (Fig. 6d). The same is observed if the initial binary pattern is highly correlated as it is shown in Fig. 6e, f.

Far more rich evolutionary behavior can be observed with a modification of the Conway’s model in which a Kronecker delta-function $\delta(l)$ in Eq. 4 is replaced by a “fuzzy delta” 11-13:

$$output^{(4)}(k, l) = \left[ output^{(k-1)}(k, l) \right] \mathcal{A} \left( L_1^{(k-1)} - C_1 \right) + \mathcal{A} \left( L_2^{(k-1)} - C_2 \right)$$

(7)

where $\mathcal{A}(l)$ is a nonmonotonic unimodal function (a “fuzzy delta”) that replaces delta function in the model of Eq. 4., $L_1$ and $L_2$ are outputs of linear filters that replace summations over 8 neighbors in the model of Eq. 4 and $C_1$ and $C_2$ are constants that replace thresholds 2 and 3 in the model of Eq. 4. In this modification, states of cells are not binary and are modeled by numbers that can have any value in the range $[0,1]$.

Experiments show that, with this model, the following three major types of the evolutionary behavior can be observed depending on the spread of the “fuzzy delta” and constants $C_1$ and $C_2$: “stable chaos”, “ordering of chaos” and “reemerging of chaos”. In the “stable chaos” mode initial chaotic pattern produced by the primary 2-D random number generator gradually evolves into visually correlated patterns that then remain to look (visually) similarly though pixel values keep changing with iterations. In the “ordering of chaos” mode, the initial chaotic pattern degenerates in the course of iterations into spatial constellation-alike or labyrinth-alike patterns that remain stable spatial-wise but may exhibit “temporal” (iteration-wise) cycles. Obviously these are the model fixed points. The most complex and varying is the behavior of the “reemerging of chaos” type. Its basic feature is rapid degeneration of the initial pseudo-random pattern into a uniform field (a trivial fixed point of the model) or into “constellations”. After that a new chaotic pattern emerges through growing crystal-alike formations from the constellations left from the initial pattern, through spatial waves from the borders when they kept to be random, or through the appearance of different types of “gliders” that move across and collide producing clouds of new “particles”. These emerging formations gradually fill in the field with visually correlated patterns similarly looking to those characteristic for the “stable chaos” mode. Examples of the evolutionary behavior of such a model are shown in Fig. 7 a - d. It is interesting to note that being initially non-quantized, cell values of the patterns tend, in the process of evolution, to stabilize at a certain set of quantized levels. This evidences in favor of a hypothesis that the quantization phenomena are characteristic not only for spatial behavior of the evolving patterns (spatial quasi-periodicity of labyrinth-alike formations), or for their temporal behavior (temporal cycles of constellations), or for their spatial-temporal behavior (gliders as spatial-temporal cycles) but also for cell value levels.
Fig. 7 Examples of evolutionary behavior of the modified Conway’s model of Eq. 7: a, b - stable constellations that, once appeared, stay forever (model fixed points), c - “dying clouds”, d - labyrinth-alike pattern. Cell value levels in the images are varying here from 0 to 255 and are coded in color as it is represented by color bars.
4. Algorithmic models of texture images

“Texture is an elusive notion which mathematicians and scientists tend to avoid because they cannot grasp it. Engineers and artists cannot avoid it, but mostly fail to handle it to their satisfaction”

B. Mandelbrot

Texture images represent an important class of images very frequently used, analyzed and generated in science, art and technology. Usually they are described by examples. There exist, for this purpose, quite a number of special albums of texture images for educational and applied purposes such as, for instance, Brodatz’s album for artists and designers. However, there is no commonly accepted formal definition of what is texture image and synthesis and analysis of textures remains to be one of the most controversial issues in image processing. This section outlines, in the same framework of dynamic systems, an algorithmic approach as a possible constructive solution of the texture image formalization, synthesis and analysis problem. It demonstrates also the relation of the problem to problems of generating pseudo-random numbers and growth modeling.

4.1 The algorithmic approach to synthesis and analysis of texture images

Within the algorithmic approach, texture images are regarded as a result of certain transformations of an initial standard pseudo-random pattern (primary texture) in a transformation system composed of a set of certain standard (elementary) units (Fig. 8). Parameters of these units and the transformation system structure form the set of parameters that define texture of a certain class. Thus, with the algorithmic approach, texture image synthesis is reduced to the synthesis of the transformation system, and texture image analysis is reduced to the problem of parameter identification of the transformation system.

Fig. 8 Algorithmic model of texture images

The specific selection of the set of structural units of texture algorithmic models is governed by considerations of convenience of their parameterization and their computational complexity. It is only natural to use, for instance, units that form basic instrumentation tool of digital image processing such as

10,11
Above mentioned point-wise nonlinearity (PWN) that transforms signal samples according to the relationship:

\[ output(k,l) = F(input(k,l)) \]

where \( F(.) \) is a, generally, nonlinear function that defines the transformation (unit transfer function) and \((k,l)\) are sample indexes.

- Above mentioned linear filters (LF). Linear filters are defined by the equation of weighted summation:

\[ output(k,l) = \sum_{m,n} h(m,n;k,l) input(k,l) \]

where \( h(m,n;k,l) \) is the filter impulse response.

- Rank filters (RF)\(^{11,16,17}\). Rank filters operate with image local histograms computed over certain neighborhood of each pixel and are defined by the equation:

\[ output(k,l) = F_{nh}(input(k,l)) \]

where \( F_{nh} \) is a function defined by the local histogram of the image over a certain neighborhood \((nhh)\) of the pixel \((k,l)\).

- Logical filters. Logical filters assume working with binary images and are defined by a certain Boolean function of input pixels. For binary images, logical filters can implement both linear and rank filters.

- Image geometrical transformation (zoom, rotation, etc.) unit.

One is free to further extend or to modify this list to include other units proved to be useful components of image processing and computer graphics tool.

For the connection of units into a system, the following types of interconnections can be assumed:

- serial connection;
- parallel connection;
- feedback.

As a primary 2-D pseudo-random number generator one can assume using the generator described in Sect. 2.

Some particular illustrative algorithmic models of textures are presented in Figs. 9-16. Fig. 9 represents the simplest PWN-model that contains only a point-wise nonlinear transformation (PWN) unit. Note that a particular version of this model with a threshold nonlinearity was used in the growth models of Fig. 3 to implement operation \(\text{randb}(P)\).
In the PWN-model, one can easily control distribution density of the gray levels of generated patterns by an appropriate selection of the nonlinearity.

The next step in the hierarchy of texture algorithmic models are LF-models that consist of a linear filter as the transformation unit (Fig. 10).

With LF-models, one can generate pseudo-random patterns with Gaussian distribution of pixel gray levels. Selection of the linear filter frequency response (Fourier Transform of its impulse response) controls Fourier power spectrum (spectral density) of the pattern and, therefore, LF-models can be used for generating correlated textures with a given correlation function (power spectrum). Fig. 11 represents two examples of such textures obtained from initial pattern of uniformly distributed non correlated pseudo-random numbers. The top one was generated with the linear filter frequency response in a form of a ring in frequency domain, the bottom one with the isotropic filter frequency response which is inversely proportional to absolute value of spatial frequency. This latter texture image illustrates what is known as $(1/f)$-fractals.

Even such a simple model as LF-one allows to imitate quite a number of natural texture images as it is illustrated in Fig. 12. Left column of the image shows natural textures from Brodatz’s album (pieces of textile, mohair and wood). Right column represents corresponding synthetic texture images generated from pseudo-random numbers in LF-model.
Fig. 11 Examples of texture images generated by LF-models and the corresponding filter frequency responses. Images in the right column are obtained from uniformly distributed independent random numbers at the output of the primary 2-D pseudo-random number generator by multiplication of their Fourier spectra by masks shown in the left column (ring shown as an image and monotonically decaying function of spatial frequencies shown as a plot).
Fig. 12 Natural texture images from Brodatz’ album\textsuperscript{15} (left column) and their synthetic copies (right column) generated from uniformly distributed pseudo-random numbers.
Combination of the threshold type point-wise nonlinearity and a linear filter in PWN-LF models (Fig. 13, a) allows to generate patterns of randomly distributed filter impulse responses as those shown in Fig. 13, b).

Fig. 13 PWN-LF-model of texture images (a) and 4 examples (b) of generated textures (shown in pseudo-colors)
Reversion of the order of point-wise nonlinearity and linear filter in PWN-LF models results in LF-PWN-models (Fig. 14a). LF-PWN-models allow to generate textures with a given correlation function (controlled by the linear filter impulse response) and a given distribution density (controlled by the nonlinear unit). Four examples of such texture images are shown in Fig. 14b.
All above described models contain only one branch (several units in cascade). The texture model can have several branches whose outputs can be combined in different ways. For instance, outputs of branches can be multiplied, or output of one branch can be used to switch between outputs of other branches, or output of one branch can control parameters of the transformation units in another branch, etc. Above described growth models of Fig. 3b and 4a exemplify such multiple branch models. Fig. 15 shows four examples of texture images generated by models with more sophisticated multiple branches.

Fig. 15 Texture images (represented in pseudo-colors) generated by models with multiple branches

Up to now no feedback connection was assumed in the texture models except the one in primary pseudo-random number generator. Clearly, feedback gives to the models evolutionary features. However, introducing feedback loop to simple PWN- or LF- models does not provide to them anything nontrivial since the result of their evolution can be obtained in appropriately modified models without feedback. Systems
with nontrivial evolutionary behavior should contain both linear filter (or any other filter with spatial interaction) and nonmonotonic nonlinearity in the loop. The above random number generator of Fig. 2 and growth models exemplify such systems. Inserting into the loop rank filters\textsuperscript{11,16,17} that combine spatial interaction and substantial nonlinearity in a more sophisticated way then just by cascading linear filters and point-wise nonlinearity gives rise to a new family of evolutionary models.

An example of such an evolutionary model with a rank filter is illustrated in Fig. 16. In this model, a particular rank filter is used that replaces, in each iteration, gray level of every pixel by the value that corresponds to the maximum of the image local histogram computed over certain spatial neighborhood of the pixel (this operation is called $\text{MODE}(\text{nbh})$). As one can see from Fig. 16, b), evolution of the texture in this model results in piece-wise constant images with “randomly” arranged patches whose size depends (statistically) on the size of spatial neighborhood used in the rank filter. These images are fixed points of this model. It is remarkable to see how well they resemble some natural crystal textures (Fig 16, c).

Fig. 16 Evolutionary texture model (a) with a rank filter and examples of primary pseudo-random pattern ((b),left) and texture images generated by the model ((b) ,center and right) and natural texture image of crystals (c).
5. Glossary

This section provides verbal explanation of some signal processing terminology used in the paper.

**Dynamic system** is a system in which output signal is fed back to its input (system with a feedback loop). An important component of the feedback loop is **delay unit** which delays signal by certain amount of time before it is put at the system input. Because of the feedback dynamic systems exhibit evolutionary behavior in time. Signals that do not change when they pass through a dynamic system are called system **fixed points**. There might be system output signals that repeat themselves periodically. This is the case of system’s **cycles**.

**Image histogram** shows how frequently particular gray levels occur in the image. If the histogram is computed not for entire image but for certain neighborhood of a pixel it is called **local histogram**.

**Linear filter** is a signal processing device that computes output signal samples by weighted summation of certain number of input signal samples. The set of the summation weight coefficients is called **impulse response**, or **point spread function** of the linear filter. Fourier Transform of the filter impulse response is called its **frequency response**

**Pixel** is picture element (sample). Spatial arrangement of image samples (pixels) is called **raster**.

**Point-wise nonlinearity** is a signal processing device whose output, for each input signal sample, is a certain, generally, nonlinear function of only this sample. This function is called **transfer function** of the device.

**Pseudo-random numbers** are computer generated numbers that appear, in a particular application, random, although are generated by means of a fixed algorithm and are fully reproducible.

**Rank filters** operate with pixel values in more sophisticated way than linear filters do. They measure, for each input sample, certain parameters of image local histograms and then use them to produce the output.

**Stochastic patterns** are patterns that exhibit (visually) elements of randomness, irregularity, chaoticity.

6. Acknowledgement

The work was completed during the author’s stay at the Center Emil Borel, Institute Henri Poincare (Unité mixte de Paris VI-UPMC/CNRS) in the frame of the trimester “Mathematical questions in signal and image processing” and of Arc en Ciel/Keshet program.
7. References