Fractal Objects in Computer Graphics

COSTIN-ANTON BOIANGIU, ADRIAN GABRIEL MOROSAN, MARIAN STAN
Computer Science Department
"Politehnica" University of Bucharest
Splaiul Independentei 313, Sector 6, Bucharest, 060042
ROMANIA
costin.boiangiu@cs.pub.ro, adrian.morosan@cti.pub.ro, marian.stan@cti.pub.ro

Abstract: This paper presents methods that can be used in generating an entire planet from mathematical objects, possibly starting from a small random seed. The planet will be generated only from specified mathematical objects, fractals, procedural models and constructive solid geometry. This planet will include elements such as vegetation, mountains, water, waves, rocky and sandy soil and clouds.

Key-Words: fractal geometry, fractal generation procedures, water generation, fractal vegetation, random fractal terrain, fractal-generating software, computer graphics.

1 Introduction

1.1 History

Unconventional mathematician Benoit Mandelbrot created the term fractal from the Latin word "fractus", which means irregular or fragmented, in 1975. These irregular and fragmented shapes are all around us. Fractals are a visual expression of a repeating pattern or formula that starts simple and becomes progressively more complex. [1]

One of the first applications of fractals emerged long before the term had been created. Lewis Fry Richardson was an early twentieth century English mathematician who studied the length of the coast of England. He reasoned that the length of the coastline depends on the length of the measuring instrument. Measuring with a measuring instrument gives a length, but considering the more irregular coastline by measuring with a smaller apparatus gives a longer length. [2]

If there is a philosophical conclusion to reach, there is an infinite coastline containing a finite space. The same paradox was presented by Helge von Koch's snowflake. [3]

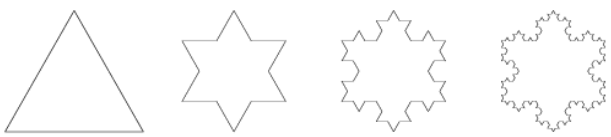

Fig. 1 Helge von Koch's snowflake (2008) [3]

This fractal involves taking a triangle and transforming each central third of each segment in a triangular bump, in a way that makes the fractal symmetric.

Each bump is certainly longer than the initial segment, but still contains finite space inside. What is strange is that the perimeter moves towards infinity rather than to converge to a special number.

Mandelbrot saw this and used this example to explore the concept of fractal dimension and to prove that the measuring of the coast is an exercise of approximation. [3]

1.2 Characteristics

The characteristics of fractals were for the first time indicated by Alain Boutot: "It has a fine structure with details on all scales of observation; It is too irregular to be described in the language of Euclidean geometry, both locally and globally; It is a self-similar structure: is the analogon of the whole; It has fractal dimension higher than the topological dimension".

1.2.1 Self-similarity

Property of self-similarity means that the parts are similar to the whole, with variations. Fig.2 contains, on the left side, a zoomed-out image of Sierpinski Gasket, the poster child of fractals, and on the right side, there is a zoomed-in image, revealing the scaling and self-similarity the characterize fractals.[17]

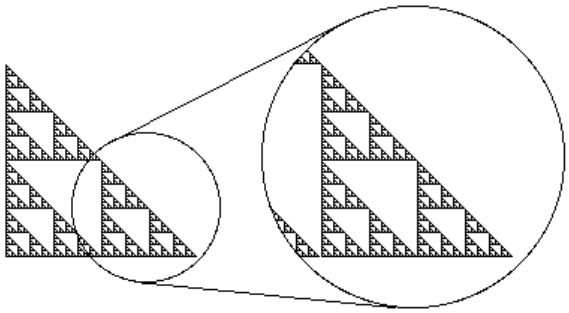

Fig. 2 Sierpinski Gasket [16]

1.2.2 Scaling

Because of self-similarity, features at one spatial resolution are related at other spatial resolutions. The smaller features are smaller copies of the larger features. The length at finer resolution will be longer because these finer features are included. How the measured properties depend on the resolution used to make the measurement is called the scaling relationship. [17]

1.2.3 Bounded infinity

Bounded infinity means that one can trace infinite length within a finite boundary, as demonstrated in the Koch snowflake's infinite line length circumscribing a finite area.

In other words, the fractal object can be represented by using a recursive function:

$$x, f(x), f(f(x)),... \qquad (1)$$

The recursive process ensures that connections will be made, but we see only parts of the whole. [17]

1.2.3 Fractal dimension

In Euclidian geometry, one dimension is represented by a line. In two dimensions, it is Cartesian place and in higher dimensions, it is a coordinate space with three or more integer number coordinates. Mandelbrot said that the fractals have the property of fractional dimensions. For a self-similar fractal, the dimension (Hausdorff dimension, named after mathematician bearing the same name) is defined as:

$$s^d = c \quad (2)$$

, where d is the Hausdorff dimension, c the amount of new copies you get after one iteration and s is the scaling factor. Using the logarithmic laws, the formula is: [18]

$$d = \frac{\log c}{\log s} \qquad (3)$$

1.3 The elements of a fractal

A vector-base fractal is composed of two parts: the initiator and the generator. In Fig.3, are represented the generator and initiator for Koch Snowflake. It starts with an equilateral triangle as the initiator and a line that is divided into three equal segments as the generator.

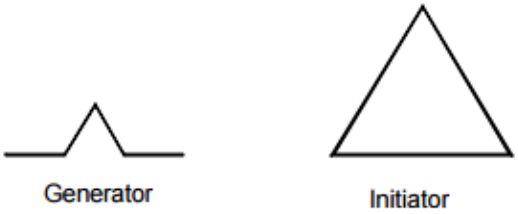

Fig. 3 Generator and Initiator by M.M. de Ruiter [19]

The first iteration is realized by replacing every line of the initiator with the full generator. A snowflake can be approximated by iterating this operation again and again (Fig.4), replacing every line of the new initiator with the full generator. To generate a real Koch Snowflake is necessary an infinite process, but in practice, the process ends after a finite number of iterations. [19]

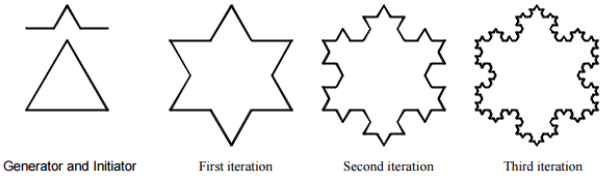

Fig. 4 The Koch Snowflake – Iterations by M.M. de Ruiter [19]

1.4 The generation of fractal objects

For generating fractal objects are several techniques, all these techniques are using feedback processes (Fig.5) in which the output of one iteration is used as input for the next one. [20]

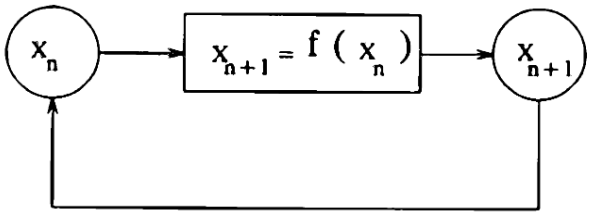

Fig. 5 Diagram of a feedback process by N. Mukaia [20]

1.4.1 Generation of fractals using formal languages

These languages, which are used for biomorphological descriptions, are known as L-systems (Lindenmayer systems). Filamentous organisms can be formalized and described by combining L-systems with branching patterns. Firstly, L-systems generate strings in a feedback loop (Fig.5). Then, in order to translate the strings into morphological description, additional drawing rules are necessary. This translation is necessary because the strings do not contain geometric information. [20]

Formal languages are the most suitable for generating plant-like objects.

1.4.2 Generation of fractal objects using Function Systems

A second approach for generating fractal objects is a method based on Iterated Function Systems (IFS). This method is used for a large class of fractal objects and it may also be applied for modelling natural objects. The IFS consists of a d-dimensional space set of mapping into itself (4) and a set of corresponding probabilities (5). [20]

$$M = \{M_1, M_2, M_3, ..., M_n\} \qquad (4)$$

$$P = \{P_1, P_2, P_3, ..., P_n\}, \text{ in which: } \sum_{i=1}^{n} P_i = 1 \qquad (5)$$

1.4.3 Geometric construction of fractals

Many fractal objects may be generated with geometric constructions. The class of fractals created using this method is known as linear fractals class. A part of linear fractals can be represented by the initiator (initial polygon), generator and the production rule. In addition to the production rules geometric information is also necessary, which is stored in data structures.

The main advantage of this class of fractals is that geometric production rules can be designed interactively and in a, relatively, easy way. [20]

1.4.4 Non-linear complex mappings generation

The most important part of the fractals is represented by fractals in which relation between input and output (Fig.5) is non-linear, using complex variables and parameters. Two examples of fractals included in this class are: Mandelbrot and Julia sets. [20]

$$z_{n+1} = F(z_n) \qquad (6)$$

$$F = f + ig \qquad (7)$$

From (6) and (7), using decomposition the complex mapping into real and imaginary part:

$$\begin{cases} x_{n+1} = f(x_n, y_n) \\ y_{n+1} = g(x_n, y_n) \end{cases} \qquad (8)$$

For each point in the complex field it can be determined how many iterations are needed until a point escapes to other points of attraction or towards infinity. For both cases are formulated stopping criteria and for each mapping, it is possible to count how many iterations are necessary until one or more of these criteria are reached. The number iterations is used as an argument for color function. [20]

2 Algorithms

In this paper will be presented ways for building objects like water waves, lands with mountains, clouds, vegetation, soil with sand or rocks and land textures, generated mathematically. [4]

2.1 Generation of water

To generate water waves there are several algorithms. The main types of waves used are sine waves, Gerstner waves and FFT waves. There are also algorithms for generating turbulent waves, algorithms for generating waves for shallow water and algorithms for generating movement of incompressible viscous fluids using the Navier Stokes equations.

Everything can be calculated mathematically, including water color and its reflection and refraction. There are also algorithms for generating caustic and foam, water drops and bubbles, using systems of particles. [4]

ISBN: 978-1-61804-313-9

2.2.1 Sine Waves

Sine waves are calculated using sums of sines, which are continuous functions that describes the height and orientation of the water surface at all points of the plane.

The status of each wave according to the horizontal position (x, y) and time t is defined as:

$$W_i(x, y, t) = A_i \cdot \sin(D_i \cdot (x, y) \cdot w_i + t \cdot \varphi_i) \qquad (9)$$

Where:

A_i is the amplitude,

D_i is the direction, the horizontal vector that is perpendicular to the wave front which moves along the top of the wave,

w_i is the frequency,

φ_i is the phase constant and

$\varphi = S \cdot \omega$, where S is the speed, also known as the distance the wave travels at the top of a frame, and $\omega = 2\pi / L$, where L is the length of the wave, that is the distance between the peaks of the waves.

For all (x, y) in the horizontal plane 2D, the 3D position of the sine waves surface is:

$$P(x, y, t) = (x, y, H(x, y, t)) \qquad (10)$$

where $H(x, y, t)$ is the total area, and is defined as: $H(x, y, t) = \sum W_i(x, y, t)$. [4]

2.1.2 Gerstner waves

Unlike sine waves, which have a rounded appearance and are suitable for calmer waters, Gerstner waves can control steepness, thus sharp waves can be generated, making them more suitable for rough waters.

For all (x, y)in the horizontal plane 2D, the 3D position of the Gerstner surface waves is:

$$P(x, y, t) = \left(x + \frac{\partial(H'(x, y, t))}{\partial x}, y + \frac{\partial(H'(x, y, t))}{\partial y}, H(x, y, t) \right) \qquad (11)$$

where t is the time,

$\frac{\partial(H'(x, y, t))}{\partial x}$ is the partial derivative of $H'(x, y, t)$ in x direction and is defined as:

$$\frac{\partial(H'(x, y, t))}{\partial x} = \sum (Q_i A_i \times D_i.x \times \cos(w_i D_i \cdot (x, y) + \varphi_i t))$$

,

$\frac{\partial(H'(x, y, t))}{\partial y}$ is the partial derivative of $H'(x, y, t)$ in y direction and is defined as:

$$\frac{\partial(H'(x, y, t))}{\partial y} = \sum (Q_i A_i \times D_i.y \times \cos(w_i D_i \cdot (x, y) + \varphi_i t))$$

,

$H'(x, y, t) = H(x, y, t) \cdot Q_i / w_i$,

and $H(x, y, t)$H (x, y, t), W_i, D_i, A_i and φ_i are the same as in the case of sine waves.

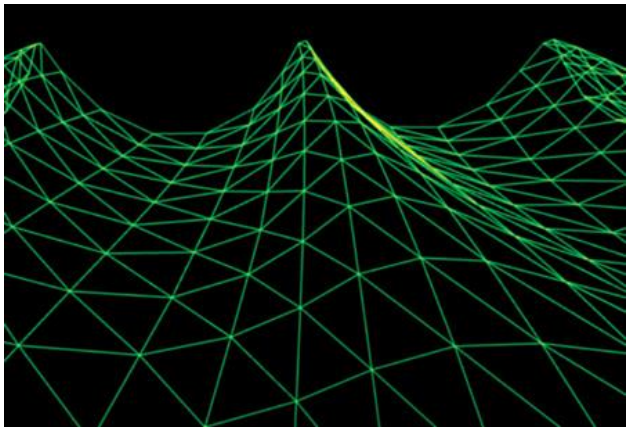

Fig. 6 Gerstner Waves (2007) [4]

In addition to the sine waves, Gerstner waves control the inclination of the wave through the Q_i parameter. If its value is zero the obtained Gerstner waves are similar to sine waves.

Unlike sine waves, the Gerstner waves and points move sideways and even if the overall wave height is the same, Gerstner waves are more realistic because sharp waves can be obtained. [4]

2.1.3 FFT waves

FFT waves are not based on any physical model, but they are based on statistical models on actual observations of oceans and seas. These waves have been used commercially several times, especially for movie animation seas and oceans.

In statistical models and crafts, wave height is a random variable, being a function of horizontal position and time. In order to obtain FFT waves the wave height field is split into a set of sine waves with different amplitudes and phases, after which inverse FFT transform is used to rapidly assess the amount obtained. [4]

ISBN: 978-1-61804-313-9

2.1.4 Agitated waves

The FFT algorithm for generating waves produces waves that have rounded edges, suggesting calmer weather. To get restless waves, with sharper peaks and flatted bases, instead of directly modify the height field, positions of the points need to be moved horizontally.

FFT waves are an alternative for sine waves, while the restless waves are an alternative Gerstner waves, and although the algorithms to generate the waves are different, the results are similar. [4]

2.1.5 Waves for shallow waters

For shallow water, waves are generated with the Saint Venant equations that are partial hyperbolic differential equations which describe the flow of a fluid underneath a surface of pressure. [4]

2.1.6 Fluid Movement

Incompressible viscous fluid motion is described entirely by the Navier Stokes equations. In these equations there are three types of forces that act: body forces, pressure and viscous forces.

Body forces are the forces acting on the entire surface of the water. It is generally assumed that these forces are formed only from gravity. Pressure forces act to the inside of the fluid and to the surface normal, and the forces due to friction of the water are the viscous forces and they act in all directions over the entire surface of the water. [5]

2.1.7 Water Color

The color of water is given by the reflection and refraction general, but the water may also have a color which depends on the direction of the incident light beam, the direction of viewing and the properties of the water. [5]

2.1.8 Reflection and refraction of water

Most visual effects of the water are caused by the reflection and refraction. This occurs when a ray strikes the surface of the water, and part of it is reflected back into the atmosphere and the other part is refracted in the volume of water.

One of the most important visual aspects of realistic rendering of water through the Fresnel equation which defines a factor of combination between reflection and refraction. [5]

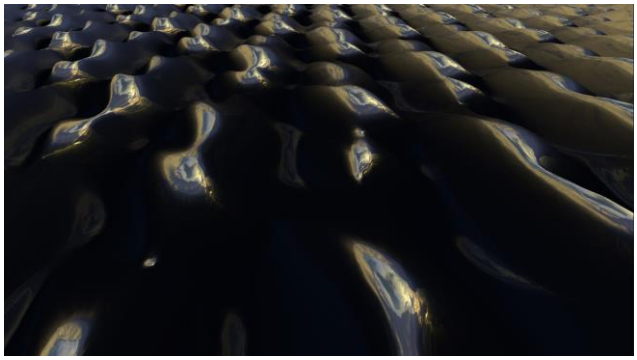

Fig. 7 Ocean with Gerstner waves, reflection and refraction by Jerry Tessendorf (2005) [5]

Fig.7 presents Gerstner wave generated water, and water color formed by reflection and refraction.

2.1.9 Caustics

Caustics result from reflected or refracted light rays on a curved surface and therefore they focus only in certain areas of the receiving surface. They can be generated “fractally” or mathematically. [6]

2.1.10 Foam, drops and water bubbles

When the water surface is very agitated or when it encounters obstacles it generates foam, water drops and bubbles due to breaking waves. This can be achieved by means of a system of particles that will be based on Newtonian dynamics. [6]

2.1.11 Generation of rivers

The classical method for generating rivers is a part of height-map generation algorithm. The algorithm generates a terrain model around a precomputed set of ridge lines and rivers network. First, it is used a rapid method that generates the ridges and rivers network. Then, an extension of the basic midpoint displacement method is applied for generating fractal terrain model around a pre-filled ridges and rivers network. [15]

ISBN: 978-1-61804-313-9

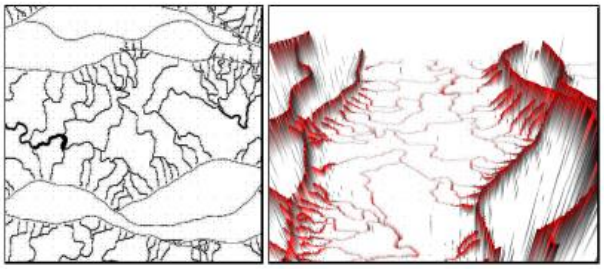

Fig. 8 The Midpoint Displacement's Inverse process by F. Belhadj (2005) [15]

In Fig.8, the left image shows: in black color the computed elevations and in the white color the elevations that still have to be computed. The right image, in black/gray color is represented the tridimensional view of D.E.M, in red the elevations computed with the M.D.I. process and in white the elevations that still have to be computed. [15]

The second approach is a post-processing step on the existing height-map. For the first one, a generated river network forms a basis from which a height-map is inferred. For the last one, a height-map is analyzed to find the potential stream routes from mountains into valleys. [12]

2.1.12 Generation of oceans and lakes

Procedural water bodies, such as oceans and lakes and theirs connections, stream networks, deltas and waterfalls are similarly generated. Oceans are commonly generated setting a fixed water level and for lakes, the classical method is a flooding algorithm from points of low elevation. [12]

2.2 Generation of land

For the generation of fields with mountains there are multiple algorithms. [7]

3.2.1 Transcendental land

Transcendental lands are lands mathematically calculated using sine and cosine functions. Transcendental lands have a rounded appearance and relate to sine waves resembling water. [7]

3.2.2 Fractal land

Fractal terrains are more realistic than the transcendental terrains, due to the sharp mountain peaks. These fields can be generated using several algorithms, among which the most important are the median shift algorithm, Diamond-Square algorithm and FFT. [7]

2.2.3 Median shift algorithm

Median shift algorithm provides realistic results. This algorithm assumes that the start is a line between two points, then the median of that line is moved with a random value in the vertical direction, then the medians of the two new segments are moved with random values in the vertical direction and then the previous step is repeated until the desired level of detail is reached. [8]

2.2.4 Diamond-Square algorithm

The Diamond-Square algorithm generates more realistic terrain than the previous algorithm, because the latter one leaves square objects in the field. The Diamond-Square algorithm mitigates this by alternating calculated values of the middle points of squares and diamonds.

This algorithm assumes the start by assigning a random heights of the four corners of the grid, and then averaging four corners, plus a random disturbance and assigns this value to the middle point of the square formed by four points. Then it takes each diamond achieved, calculate the average of the four corners of each diamond, plus a random disturbance and assigns the middle point of the diamond. The previous two steps are then repeated until reaching the desired level of detail. [8]

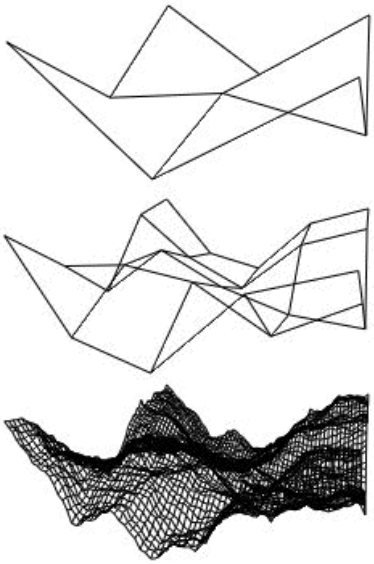

Fig. 9 Fractal terrain generation using Diamond-Square algorithm by David P. Feldman (2012) [8]

Fig.9 shows the generation of a fractal terrain using Diamond-Square algorithm, from the initial stages of low level detail in the final stages with high level of detail. [8]

ISBN: 978-1-61804-313-9

2.2.5 FFT algorithm

Unlike the movement of the median algorithm and Diamond-Square algorithm which have linear running time, the FFT algorithm's runtime is logarithmic.

Another difference of the FFT algorithm is that the terrain has a more rounded texture and has no raised or pointed peaks. [9]

At distance either less frequencies can be considered, or the terrain simplified for speed purposes. [11]

2.2.6 Erosion fractal

The height-maps can be transformed using simulations of physical phenomena, such as erosion. In order to diminish sharp changes in elevation, it is used thermal erosion, by iteratively distributing material from higher to lower points, until the maximum angle of stability for a material is reached.

Another type of erosion is caused by rainfall or fluvial erosion. This type can be simulated using, cellular automaton (model of a system of "cell" objects), where dissolved material that flows out to other cells and the amount of water are calculate based on the local slope of the elevation profile. [12]

2.2.6 Digital elevation models (DEM) method

Besides other methods, this setup allows the user to interactively edit the height-map. A user draws a 2D map of polygonal regions, each of which is marked to have a certain elevation profile. The straight boundaries of the regions are perturbed and rasterized in a grid. Then, for each region, DEM data is selected to match the requested elevation profile. The selection of data is realized using genetic algorithms.

The main advantages of this method are: realism, extensibility and ease of use (intuitive control, with low input requirements). Although, the generated transitions at the boundaries between regions are still rather abrupt. [13]

2.1 Cloud Generation

To generate clouds, noise functions are used in general. The noise that offers the most realistic results is Perlin noise. Meteorological phenomena, such as lightning, can be mathematically generated or through fractals. Rain and snow, can be generated using a particle system. [4]

2.3.1 Noise functions

The role of noise functions is to provide a pseudo-random signal, efficiently implemented and repeatable over a three-dimensional space. In general, the sound functions receive an integer as a seed and they return a pseudo random number based on the received parameter. [4]

2.3.2 Perlin Noise

To create a Perlin noise function, a noise and an interpolation function are required. The noise function must be, in general, a random number generator that will return the same value if two numbers with the same value are sent as input and different values if two different values are sent as a parameter.

A standard interpolation function receives three parameters, two of which represent the values between the return value that must be interpolated and the third parameter based on which returned value is interpolated. [10]

2.4 Generation of vegetation

Vegetation can also be generated with fractals or mathematically. Procedural vegetation is a classic research topic in the field of procedural modelling and includes both procedures for generating 3D plant models and trees and specific methods for placement on a given surface. [4]

2.4.1 Generating plants

There are many plants that can be generated by fractals or mathematically, including ferns, grass, flowers, fruits, vegetables or leaves. Plants can be generated using L-systems or using iterated function systems. [4]

Branching patterns may be defined as L-systems by using K denoted by <G, W, P>, in which G is a set of symbols, W is the starting string and P is the production rule. Monopodial branching, in which the growth of the main axis continues throughout the plant's life, may be represented by the following L-system [20]:

$$KM = \langle G, WM, PM \rangle \quad (12)$$

$$G = \{0,1,[,]\} \quad (13)$$

$$WM = 0 \quad (14)$$

$$PM = \{0 \to 1[0]0,1 \to 1,[\to[,]\to]\} \quad (15)$$

<Iteration>	<Generated String>
1:	0
2:	1[0]0
3:	1[1[0]0]1[0]0
4:	1[1[1[0]1]1[0]0]1[1[0]0]1[0]0

The drawing rule showed in Fig.10 is one possible way to visualize the generated string.

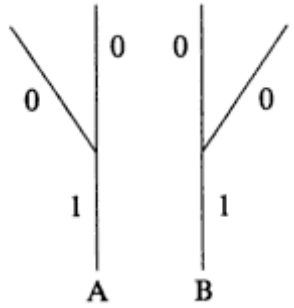

Fig. 10 Drawing rule for the string 1[0]0, shape A and B by N. Mukaia [20]

An alternative system to procedurally model plants is by placing plant components in a graph. Connected components can be structured in sub-graphs. The system traverses this graph, generating and placing instances of the components in an intermediate graph that is used for geometry generation. A set of components is connected by the user to describe the structure of the plant. The algorithms are controlled by graphical user interface on the basis of spline functions. [14]

2.4.2 Generation of trees

Trees can be generated using L-systems or iterated function systems. L-system is a classical and often used example of a rewriting system. Although the L-systems are used for rewriting strings of text, the resulting set of symbols can be interpreted in 2D and 3D. One of the best known such tree is the tree of Pythagoras, which is built recursively. [4]

2.5 Soil generation

The most important elements of this category that can be generated are rocks and sand. Sand dunes using noise functions can also be generated. [4]

2.6 Urban environments

The common approach for procedurally generating cities is to start from a dense road network and identify the polygonal regions enclosed by streets. Building lots are the result of subdivision of these regions. Then, for populating these lots, the lot shape is used directly as the footprint of a building (Fig.11). Another method is to fit the building footprint on the lot. By simply extruding the footprint to a random height, it can be generated a city of skyscrapers and office buildings. These approaches are used to create a macro environment, for more complex details, are necessary several rule-based methods. [12]

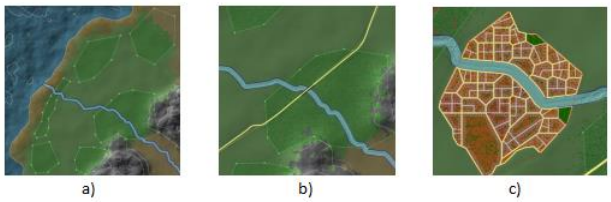

Fig. 11 Procedural sketching session: a) river flowing towards the sea, b) road feature crossing the river, c) city created along the river banks by R.M. Smelik (2011) [12]

2.7 Generation of textures

They can be generated using height maps obtained at the terrain generation. [4] Another method is used for generating texture image such as surfaces of houses or roads. For this, random or fractal functions are used. This method uses two types of image. One is a material image and the other is a weathered one. The material image is generated by placing some fundamental patterns at random. The more patterns are used, the more varieties are generated. On the other hand, the weathered image is generated using fractal functions. Intended images, such as a texture that represents water drop track on the wall, can be realized by selecting the fractal seeds. Also, combining both material and weathered images can make more realistic images for texture mapping. The generated images look like very natural and enable the modelling data of landscape simulation very realistic. [20]

2.8 Generation of other elements

The most important of these elements are animals such as spiders, snails, peacocks, sea urchins and starfish, but the generation of stalagmites and stalactites and crystals is also possible. [4]

ISBN: 978-1-61804-313-9

3 Conclusions

This paper presents ways to generate as many classes of objects on the basis of equations, possibly with minimal data, such as a seed. Items that can be generated include: terrain, terrain textures, clouds, vegetation, soil and water.

Acknowledgments:

The research carried by the first author was supported by Electronic Arts Romania S.R.L.

References:

[1] Benoit Mandelbrot, The Fractalist: Memoir of a Scientific Maverick, *Knopf Doubleday Publishing Group*, 2012

[2] Lewis Fry Richardson, Oliver M. Ashford, Philip Gerald Drazin, *The Collected Papers of Lewis Fry Richardson*, Cambridge University Press, 2009

[3] NOVA, Hunting the Hidden Dimension, PBS, 2008, http:// www.pbs.org/ wgbh/ nova/ physics/ hunting-hidden-dimension.html, accessed 26.05.2015

[4] NVIDIA, GPU Gems, NVIDIA Corporation, 2007, https:// developer.nvidia.com/ gpugems/ GPUGems/ , accessed on 26.05.2015

[5] Jerry Tessendorf, Simulating Ocean Water, Computer Graphics Laboratory, 2005

[6] Yaohua Hu, Luiz Velho, Xin Tong, Baining Guo, Harry Shum, Realistic, Real–Time Rendering of Ocean Waves, *Computer Animation And Virtual Worlds*, vol. 17, issue 1, pp. 59-67, DOI: 10.1002

[7] Krista Bird, Thomas Dickerson, Jessica George, Techniques for Fractal Terrain Generation, HRUMC, 2013

[8] David P. Feldman, Chaos and Fractals: An Elementary Introduction, Oxford University Press, 2012

[9] Michael F. Worboys, Matt Duckham, GIS: A Computing Perspective, Second Edition, CRC Press, 2004

[10] David S. Ebert, Texturing & Modeling: A Procedural Approach, ISBN-13: 978-0122287602, 1994

[11] Costin-Anton Boiangiu, Bogdan Raducanu. "3D Mesh Simplification Techniques for Image-Page Clusters Detection". *WSEAS Transactions on Information Science, Applications*, Issue 7, Volume 5, pp. 1200 – 1209, July 2008

[12] R.M. Smelik. "A Declarative Approach to Procedural Generation of Virtual Worlds" , 30 november 2011, thesis

[13] Ryan L. Saunders. "Terrainosaurus: Realistic Terrain Synthesis Using Genetic Algorithms", December 2006

[14] O. Deussen, C. Colditz , L. Cocunu , H.C. Hege. "Efficient modelling and rendering of synthetic landscapes"

[15] F. Belhadj, P. Audibert. Modeling Landscapes with Ridges and Rivers, November 2005, *Proceedings of the 3rd international conference on Computer graphics and interactive techniques*, Australasia and South East Asia, pp. 447-450

[16] https://classes.yale.edu/fractals/IntroToFrac/SelfSim/SelfSim.html accessed 27.05.2015

[17] L. Strudwick. Infinite Space and Self-Similar Form in Alchemy and Fractal Geometry, *Mythological Studies Journal*, Vol 1, No 1 (2010)

[18] Magdy M. Ibrahim and Robert J. Krawczyk. Generating Fractals Based on Spatial Organizations. College of Architecture, Chicago, IL USA

[19] M.M. de Ruiter (Ed.). "Advance in Computer Graphics III". EurographicSeminars

[20] N. Mukaia , Y. Sakaguchia, M. Kosugia. "A Method for Generating Texture Images used on Landscape Simulation". Electronic & Computer Engineering, Musashi Institute of Technology, Tokyo, Japan