Procedural Modeling of Natural Phenomena

Marius Dorian Zaharia
POLITEHNICA University Bucharest
Computer Science Department
Splaiul Independentei, 313, Bucuresti – 77206 Romania
E-mail: zaharia@cs.pub.ro

Abstract: The paper describes some methods used to model natural objects like: fire, grass, different types of trees, water or mountain surfaces. This kind of techniques is used in an application program to model natural 3D scenes. The images synthesized through the methods described below could be used as basic frames in animation programs. An evaluation of the modeling algorithms is also presented.
Keywords: realistic image synthesis, fractal, texture, particle systems, fuzzy object, Gouraud shading, Fourier synthesis, data amplification.

1. Introduction

A class of objects from real world that are modeled in realistic image synthesis systems is formed from irregular shaped objects whose surfaces (shapes) may change in time. The main techniques for modeling such fuzzy objects are fractal geometry ([7]), particle systems, ([2],[9]) light scattering ([2],[11]). These approaches of image synthesis are based on a combination between the two main phases of computer graphics scene visualization: object modeling and model rendering.
Procedural models are a very useful way of defining visual representations of natural phenomena; they allow modeling objects and motion in the same time. The models are parameterized. This type of methods can be used to create models with a huge amount of various (stochastically generated) details, more than a human designer could specify.
An interesting method for modeling natural phenomena are the particle systems. These are collections of entities (called particles) that evolve in time. These particles are generated by stochastic processes and they have randomly distributed life times. During their life, the particles move or change attributes according to specific lows.
Another heavily used modeling strategies are based on fractal geometry. A fractal object is characterized by:

- his fractional dimension - that express the infinite variety of details at each point of the geometric fractal space
- the property of self-similarity (or statistical self similarity) which signifies that there are portions of the fractal which could be mapped, through simple geometric transformations, to other portions.

As Mandelbrot has already stated ([7]) the fractal objects could represent the shapes of various natural objects as mountains, dust, sand, or water surface.
The two above-mentioned methods were used to generate images of different natural objects or phenomena.

2. Description and implementation of the modeling methods

2.1. The fire model

The model is based on self-similar patterns generation. The data amplification process starts with a seed formed of 4 rhombs placed as indicated in Figure 1. The base rhomb is placed upon the fire base plane; the other three rhombs have a 60° elevation relatively to the base plane. They are rotated with randomly chosen angles around the normal vector of the base plane. The rotation angles are determined by:

$$u_{rot} = u_{init} \pm random() * \pi \quad (1)$$

where u_{init} has respectively the values: $-\pi/2$, $\pi/6$, $5\pi/6$

The geometric parameters used to completely specify the shape of the seed are:

- the length of the longest diagonal of the basic rhomb
- the aspect ratios of the four above mentioned polygons (all these ratios are relative to the longest diagonal of the basic rhomb).

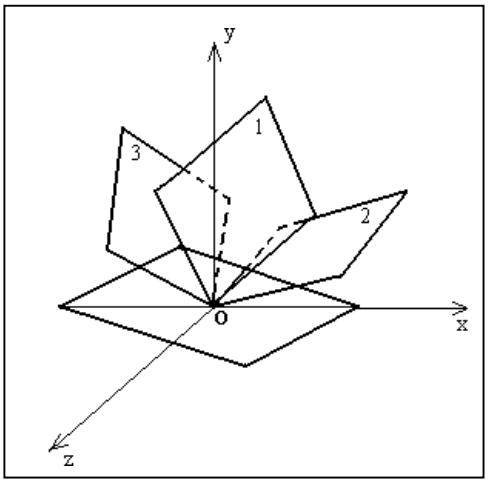

Figure 1 – The seed for the fire generation process

The data amplification is accomplished through a recursive process. Along the longest diagonals of rhombs (1), (2), (3) at parametric coordinates 0.15, 0.33, 0.66 and 1 are generated scaled copies of the seed. These copies have scaling factors of 0.6, 0.6, 0.5 and 0.2 respectively (the actual values of the scaling factors are determining the fire shape). The process stops when the copies become smaller than a predefined threshold value.

An important aspect of the fire visualization is the shading method used to render the fire tongues. The shading color is chosen appropriately depending on the temperatures in different areas of the fire. It is reddish in the areas closed to the center of the basic rhomb and yellowish near the fire boundary. In fact every point of a fire tongue is a light emitter. The basic rhomb is sweeped by a vector having one extremity in the center and the other extremity moving along the polygon boundary. The color of the interior pixels of the sweeping vector is obtained through linear interpolation between the colors of the vector extremities. The other three rhombs of the seed are shaded using a scan line algorithm. All the points placed on one horizontal scan-line have the same color (c). The value of (c) results from linear interpolation between the colors of the two most distanced vertices (measured along the axis normal to the scan-line) of the rhomb.

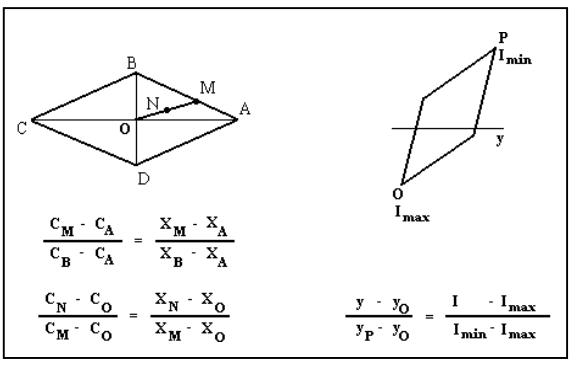

Figure 2 – Shading a fire rhomb

2.2. The grass model

The grass is generated as a collection of tufts. One tuft is a collection of some hundreds of grass blades that are growing starting from closely positioned points. A grass blade is a particle which, during its life time, generate a straight line (in fact a thin triangle). The direction of the grass blade is specified by two rotation angles (around Oy and Oz axes respectively) that are laying the $\vec{k}$ versor along the grass blade.

The parameters characterizing the shape of the grass tuft are:

- the maximal length of the blades composing the grass blade
- the spread factor of the blades
- the maximum number of grass blades in a clump

These parameters are used to compute the information necessary to render each grass blade. They are computed using relations similar to (1). For example:

- The grass blade length: $l = l_{min} + (l_{max} - l_{min})*random()$
- The width at the base of the grass blade: $w = w_{min} + (w_{max}-w_{min})*random()$
- The rotation angles associated to a grass blade: $u_y=\pi/2*random()$; $u_z=2*\pi/random()$

where random() returns a pseudo-random value uniformly distributed between 0 and 1.

In order to render a grass tuft, the particles are set as light reflectors that is why, for every grass blade was applied the Gouraud shading model ([4]). As it was already stated, a grass blade is an extremely thin isosceles triangle (with his base of maximum 1.5 pixels wide); in that case the Gouraud shading has a visual effect similar with drawing an antialiased vector using a Gupta-Sproull algorithm.

The amount of detail created by these algorithms is so large that exact visible surface determination or shading calculations require a computing effort too high to be supported by the available processor (Pentium/200MHz). That is why the particles are processed from back to face (i.e. in the decreasing order of the distances from the viewing point to the center points of the grass blades). The same technique is used to render trees collections in a forest, even if some branches of a more distanced tree could obscure some branches of a tree placed closer to the viewpoint. This last mentioned effect could be ignored due to the huge amount of detail contained in the models of the scene objects.

2.3. Models of different types of trees

There were modeled two types of trees the oak-tree and the fir. The oak has secondary branches that rise around the trunk at different positions along the trunk (without forming “floors” as in case of firs). The main trunk is modeled as a truncated cone whose surface is rendered using a texture mapping technique. The secondary (intermediate) branches are generated recursively and are

placed around the trunk at positions and rotation angles randomly distributed. The terminal branches are covered with leaves that correspond with small, thin ellipsoids, colored using a Fourier synthesis method ([3]).
In case of firs the needles are simply short vectors and the last four levels of branches have greenish shades that are darker for smaller branch levels. The different structural elements (branches, leaves) are placed in space conforming to statistical laws in function of the tree category (conifer or leaf-tree).
The parameters used to individualize a tree are:

- The tree position. It is a 3D point representing the center of the base of the tree trunk
- The age. This parameter influences the values of all the geometric attributes associated to the tree and can be interactively modified by the user
- The number of sub-branches. It is computed with:
  no_branch=min_branch+(max_branch–min_branch)*(age – min_age)/(max_age-min_age)
- the length of the current branch
  l_trunk=length*age/max/age+
  $(-1)^{random(65)}$*Δ*random(no_branch)/no_branch
  l_branch=(length_father–dist)*(adjust_branch+
  (age/max_age))*random()
  where adjust_branch is a parameter whose value increases with the level of branches in order to prevent a too large reduction of the branch length.
- the rotation angles (u_z, u_y) of the current branch relatively to a local reference system whose origin is placed in the starting point of the parent branch. The process of building the current branch uses this local coordinate system. Given a branch (b), the value of the associated u_y angle will determine the drawing order of the subbranches of (b). The branches directed to the scene background (opposed to the viewer) will be processed first.

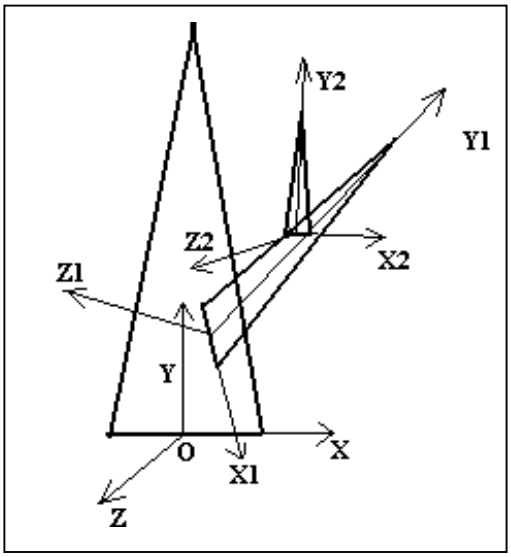

Figure 3. Branch generation for a fir-tree

As mentioned above, the leaves are attached only to the terminal branches of the oak. They are ellipsoids with semiaxes parallel to the axis of the referential associated to the leaf and that have (positive) lengths normally distributed around 3, 1 and 0). The ellipsoids are textured using a Fourier synthesis approach. The same method was used to model terrain texture and will be described below. This coloring method does not take into account the light source position or the multiple reflections that could occur between the surfaces of the leaves.

2.4. The terrain model

The earth surface is modeled by second degree B-spline patches. The patches are generated in two phases. First the geometry of the patches is determined interactively by the user that introduces the control points of those surface patches. Every B-spline patch is approximated by a polyhedral surface. The faces of the approximating polyhedron are quadrilaterals (supposed plane). The vertices are points on the B-spline surface patch, placed at intersections of two sets of iso-parametric curves.
The approximating polygonal mesh is rendered using a combination of two methods: first a Gouraud interpolation shading technique, second a Fourier texture that is applied to every element of the polygonal mesh. This method adds detail through local modulation of the reflected light intensity.
The use of the Fourier synthesis technique for simulation of natural surface textures was originally proposed by Gardener ([3]). It is a powerful modeling method that could be used to model the leaves of the maple-tree, the surface details of a rocky mountain or the texture of a green pasture. Gardener uses a scalar function defined over a three dimensional domain by a linear combination of sinusoidal functions. The parameters of these sinusoids (i.e. the amplitude, period and phase) are selected by the programmer in order to perform various visual effects. In case of a bi-dimensional definition domain, the function proposed by Gardener and used in the present program is:

$$G(x,y)=\sum_{i=1}^{n} c_i(\cos(\omega_{xi}\,x+\varphi_{xi})+A_0)+\sum c_i(\cos(\omega_{xi}\,y+\varphi_{yi})+A_0)$$

The frequencies ω satisfy the equations (specific to Fourier series)
$\omega_{i+1}=\omega_i$ for i>=0, (ω_0 was chosen 12)
The amplitudes c_i satisfy the equations:

$$c_{i+1}=c_i\sqrt{2}$$

(c_0 was chosen to be 1.76)
A_0 is a basic offset and it provides contrast control (it was chosen to be 1).

3. Efficiency of the algorithms

The fire. The recursive algorithm that generates a fire has the following phases:

- computing the geometric parameters that define the seed (an O(1) algorithm)

- visualization of the polygons forming the current seed. There are two shading methods of the fire rhombs. Shading the rhombs (1), (2), (3) (see Figure 1) is cheaper than shading the basic rhomb of the seed, because of the larger amount of incremental calculus in the first of the above mentioned cases.
- computing the position parameters of the new seed (child of the current seed)

Table 1 shows the dependence of the fire rendering time, in function of "cf01" parameter which is the ratio between the length of the longer diagonal of the rhombs (1), (2) or (3) and the length of the longer diagonal of the basic rhomb (parameter "axa"). It is a parameter that heavily influences the amplitude of the fire.

axa	20	20	30	30	30	30	35	40
cf01	0.3	0.5	0.3	0.5	0.7	0.9	0.3	0.3
time	24.6	27.9	79.1	82.3	83.5	82.5	179	297

Table 1

The trees. The phases of the process that generate an oak are:

- determining the position and shape of a branch; that is computing the base and apex of the cone that approximates the branch) and applying to those points the transformations of the visualization pipeline. This process includes also the sort (in increasing order of $\sin(u_y)$)of the child branches of the current branch.
- Rendering a branch consists in applying the Gouraud shading algorithm for a triangle and mapping a Fourier texture to the triangle surface.
- Leaves rendering. It was already mentioned that the leaves are placed only on terminal branches. A typical number of leaves on a branch is between 5 and 10, depending on the branch length. Obviously, rendering two oaks with same age could require considerably different amounts of time. Over 95% of the time used for generating an oak is spent by the leaves rendering process. From Table 2 results that the mean rendering time per leaf is approximately 4.1 hundredths of second.

Age	Total rendering time	Leaves rendering time	No. of leaves	No. of branches
100	31988	31820	7673	1755
120	27941	27786	6727	1707
140	35296	35128	8521	1856
160	64038	62738	15497	2561
180	55931	55599	13467	2731
200	55244	55078	13318	2522

Table 2

In case of the fir, the computation effort is considerably smaller because the rendering process of a leaf is much simpler. The leaves are no more modeled by thin ellipsoids but they are assimilated with simple vectors. As an example, the total rendering time of an 100 years old fir, with a branching factor of 8 is around 75 hundredths of second.

The terrain. The model of the hilly ground is formed from 5 adjacent B-spline patches. The process of interactively specification of the 50 control points can be cumbersome and is not taken in consideration in the efficiency analysis. The other two phases of modeling the surface of the ground are:

- Computation of a 3D polygonal mesh (of maximum 35x35 quadrilaterals) that approximates the B-spline patch
- Rendering the polygonal mesh i.e. applying the visualization pipeline transformations to each polygon vertex, the Gouraud shading procedure and the Fourier synthesis method in order to add details to the polygons in the image space.

For example, rendering the terrain in Figure 3 took 36.80 seconds that corresponds to 0.042 seconds per quadrilateral.

The running times determinations where made on an INTEL Pentium (200Mhz) processor. All the experimental results are specified in hundredths of second. Figure 4 is the visualization of the model of a green (Fourier synthesized) hilly ground with some grass clumps and an automatically generated fir forest (in the upper right corner of the figure). Figure 5 shows five oaks on a grass field.

Conclusions

The above mentioned modeling methods (based on particle systems, fractals or Fourier synthesis) are valuable tools in realistic image synthesis applications. They could be used to produce basic frames (background images) in animation programs. The methods presented above support considerable refinement in order to increase rendering realism. Some methods require a considerable amount of rendering time, that make them suitable to be used only for static scenes visualizations (in case of limited hardware

resources). A future direction of work could consist in studying the behavior of such modeling techniques in case of dynamic scenes (where the modeled objects are moving).

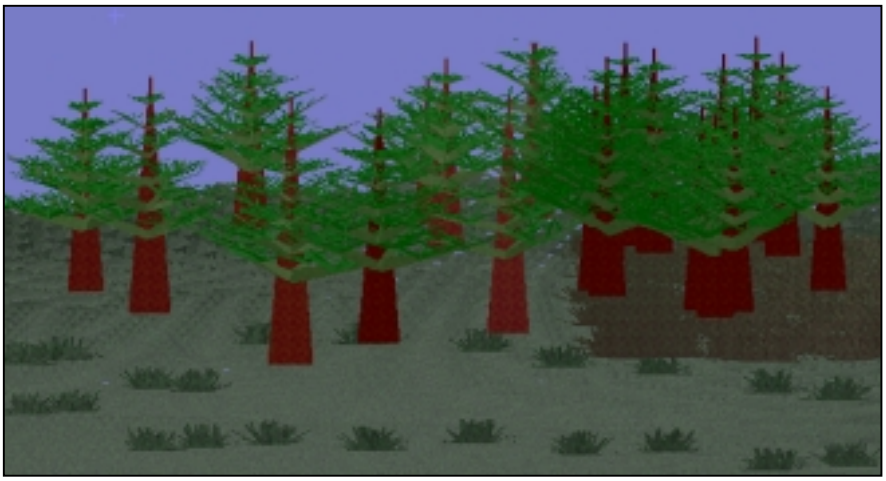

Figure 4 – Fir tree forest

Figure 5 – Five oak trees

References

[1] R. Bukowsky, C. H. Séquin, "Interactive Simulation of Fire in Virtual Building Environments", *Proceedings of SIGGRAPH '97*, pp. 35-44, 1997

[2] J. Foley, A. van Damm, S. Feiner and J. Hughes, *Computer Graphics: Principles and Practice*, Addison Wesley publishing Co., pp. 1031 – 1035, 1992

[3] G. Y. Gardener, "Simulation of natural scenes using textured quadric surfaces", *Computer Graphics*, No. 3, pp. 11-20, 1984

[4] H. Gouraud, "Continous shading of curved surfaces", *IEEE Transactions on Computers*, pp. 623-629, June, 1971

[5] N. Magnenat Thälmann, D. Thälmann, *Image synthesis*, Springer Verlag, pp. 220-330, 1985

[6] N. Magnenat Thälmann, D. Thälmann, *Computer Animation: Theory and Practice*, Springer Verlag, Tokio, 1990

[7] B. Mandelbrot, *The fractal geometry of nature*, W. H. Freeman, San Francisco, 1982

[8] F. Moldoveanu, Z. Racovita, S. Petrescu, G. Hera, M. Zaharia, *Grafica pe Calculator*, pp. 418-439, Editura Teora, 1996

[9] W. T. Reeves, R. Blau, "Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems", Proceedings of SIGGRAPH '85, pp. 313 – 322, 1985

[10] S. T. Todericã, "Modelarea proceduralã a fenomenelor naturale si generarea de imagini realiste ale scenelor 3D", Dyploma Project No. 55/98, Universitatea POLITEHNICA Bucuresti (coordinator: dr. eng. Marius Dorian Zaharia), 1998

[11] A. Watt, *3D Computer Graphics*, Addison Wesley Publishing Company, pp. 377-379, 1993