The Modelling of Natural Phenomena

Alain Fournier
Department of Computer Science
Sandford Fleming Building
University of Toronto
Toronto, Ontario
M5S 1A4
alain@dgp.toronto.edu

Abstract
The main activity of physical sciences is to establish models of reality which can be used for measurements, predictions of new phenomena, and possibly for the falsification of current theories. Modelling in computer graphics has many of these characteristics, and this makes it a particularly interesting activity. The modelling of natural phenomena is especially challenging, because these are complex, numerous and familiar.

I propose in this paper a taxonomy of the types of models which have been used in computer graphics, with an analysis of their good and bad qualities, and illustrative examples. I distinguish empirical, physical, structural, morphological, impressionist and self-models.

In addition to models, some basic principles and concepts have proven to be useful for the effective modelling of natural phenomena. I review a somewhat arbitrary list: database amplification, levels of details, stochastic elements, time in models, particle systems, textures and fractals, and illustrate with some examples.

Résumé
L'activité principale des sciences physiques est la création de modèles qui peuvent être utilisés pour des mesures, pour la prédiction de nouveaux phénomènes, et pour l'éventuelle falsification des théories établies. La modélisation en infographie a beaucoup de ces caractéristiques, et ça en fait une activité particulièrement intéressante. La modélisation des phénomènes naturels est spécialement difficile, parce qu'ils sont de forme complexe, ils sont très nombreux et ils sont très familiers.

Je propose dans cet article une taxonomie des modèles qui sont utilisés en infographie, accompagnée d'une analyse de leurs qualités et de leurs défauts et d'exemples. Je distingue les modèles empiriques, physiques, structuraux, morphologiques, impressionnistes et auto-modèles.


1. Introduction

Bed and Board is a François Truffaut film in the series chronicling the life of Antoine Doinel. It opens as the hero’s job is to dye carnations by dipping them into various colouring mixtures. A basic question we can ask when modelling natural phenomena in computer graphics is whether we are like Antoine Doinel, peddling old flowers dipped in colour, or whether we grow our own.

Most of the early modelling tools were designed for the relatively simple forms of artificial objects. When it comes to natural objects, such as land, trees, clouds, water, fire, animals, and the trees in Figure 1 require 40 or one of us humans), those simple modelling tools are not enough. I will try in this paper to define what modelling means in this specific context, propose a taxonomy of the types of models which can be used for natural objects, and examine the general concepts and principles which can guide us to model complex objects,

I will deal in this paper only with shape modelling. Other aspects of modelling, are important when dealing with natural objects. One is the modelling of the interaction between light and matter. This can be subdivided into two categories, atmospheric effects, where the matter interacting with light has no shape itself, and illumination models for surfaces. Illumination models can in turn be subdivided into local illumination models, and global illumination models. The book by Hall, Illumination and Color in Computer Generated Imagery (Hall88) is a good reference for both kinds. Yet another aspect is the problem of modelling and animating moving living bodies or part of bodies, articulated or not. These are fascinating topics, but ones which require much different tools and methods (see [Magna87, Whil87] for a bibliography and a survey).

2. How complex is the world

For once we have a question with a simple answer: the word is very complex. After all it is made of $10^6$ to $10^7$ particles, each one being a fairly complex object itself. Even if we modestly limit ourselves to modelling a small portion of the world, let us say what we see on Figure 1, we have to deal with about $10^6$ particles. Maybe there is a simpler way than to model each particle. After all we are not necessarily interested in modelling all the properties of the objects. In fact, in most cases the appearance of the objects is all we care about. In this case a reasonable model of the surface might be enough. This is precisely why most of the traditional modelling primitives are surfaces. But that does not yet save us from trouble. Many objects have very complex surfaces. Look at the material of your clothes, your skin, the surface of the oceans. Polygons or even higher order parametric surfaces will not work very well here.

To go back to the example of Figure 1, we see about 200 trees, each one has roughly 10,000 leaves. For an ordinary leaf 20 triangles might suffice if we model it with triangles. That means each tree require 200,000 triangles, exclusive of trunk and branches, and the trees in Figure 1 require 4 or 1 million triangles. Clearly the time necessary to design, to store, to transform and to render the trees that way is prohibitive. Techniques are necessary to reduce any or all of these tasks.
2.1. General Characteristics of Natural Objects

Briefly, what makes the modelling of natural objects especially challenging is:

- Most objects have very complex shapes. Some objects cannot even be well described by surfaces. Examples are turbulent water, fire and smoke. In these cases very different types of models will be necessary.

- Natural objects can be needed in very large quantities. When modeling mechanical parts, for instance in a CAD system, they are only needed in small numbers. Even in the case of a VLSI CAD system, where there can be thousands of primitives, each primitive is very simple (a few rectangles). To model a landscape, thousands of trees, each with thousands of leaves, each leaf with a complex shape, are needed.

- Natural objects move in complex ways. Most artificial objects move rigidly, and basic transformations in computer graphics can easily take care of that. But consider the motion of clouds for instance. As a result the motion has to be built into the model, when the model is designed, for any hope of realistic motion.

- And finally, we are very familiar with the look of natural objects, and our visual system is finely tuned to their characteristics. So any discrepancy between the look of our models and the look of the "real thing" will be easily detected.

2.2. Some Inspiring References

This paper ends with a respectable (but not exhaustive) bibliography, and many papers will be specifically referred to below. But before we go further, I want to point out a few works which have to be read, because they are both highly informative and inspirational. Of Growth and Form, by D'Arcy Thompson [Thom56], is a remarkable book, an attempt to stress the importance of geometry and basic physical laws in determining the shape of living things (as opposed to the internal phylogenetic causes).

It has many wrong or suspicious conclusions, but has enough fascinating material left to be of great value to us. A more recent book by Peter S. Stevens, Patterns in Nature [Stev74] is also a marvelous walk (with abundant illustrations) through the variety of natural shapes, with many insightful explanations of their possible origins. He also points out many unexpected similarities. Benoit Mandelbrot developed the concept of fractal, and his book The Fractal Geometry of Nature [Mand82] (preceded by a French language version in 1975 and an English language version in 1977) explains and illustrates the relevance of fractals to natural phenomena with striking pictures, challenging concepts and exciting prose. Finally, Light and Color in the Open Air by M. Minnaert [Minn54], is a very useful and charming book on many atmospheric effects such as fog, rainbows, haloes, etc.

3. Types of models

Traditional modelling techniques use simple primitives to model points, edges or, more commonly, surfaces. In particular we use parametric surface representations for objects whose surface is the main factor in design.

These include car bodies, plane fuselage and ship hulls. The shape is primary in these cases. The book The Modelling of Shape and Form [Lord84] gathers together in one place information about the diverse mathematical ways to model shape and form. For objects where the volume, and the machining operations to fabricate them are the most important factors, solid object modelling techniques, and in particular constructive solid geometry were the answer. In both cases there were important properties to maintain, such as continuity and smoothness for surfaces, and consistency and integrity for solids. While modelling natural objects, the look is the overriding consideration. But, and it is an interesting verification of the old adage "form follows function", it is sometimes necessary to use the functional characteristics of the objects to be modelled to obtain a successful model. In practice, computer graphics has seen a very wide range of models, from models directly derived from the physical situation to models without any known connection with the natural forces at work. What follows is an attempt at a taxonomy of the models we can use. It is always dangerous to establish (and even more to publish) a taxonomy, mostly because it is always difficult to put things in different categories, but a lot more difficult to prove that the categorizing is of any use. I will let you be the judge of that. Some measures of the usefulness of a taxonomy are whether everything falls easily into each category, whether the categories point at similarities and differences which might not be otherwise obvious, and, mainly, whether they help in selecting or eliminating possibilities when designing or choosing a model. Even if the taxonomy fails to convince, the exercise is still useful.

It is important to note that the types of the models used and the primitives used to build the models are rather different issues. I will give examples to illustrate this later. And, finally, this taxonomy is not restricted to the modelling of natural phenomena, but applies to computer graphics models in general, even though the distinction between types of models is less important in most other applications.

The types of model I will use in this taxonomy are: empirical, physical, morphological, structural, impressionist and self-models.

3.1. Empirical models

The most straightforward approach to modelling a natural object is to basically "digitize" an instance of it. For example real terrain can easily be modelled thus since there is an abundance of data coming from direct measurements "on the terrain", or from various satellite data. One can obtain from various geographic information systems the three-dimensional positions of a fairly dense grid of points. These points can be triangulated, and then rendered using traditional techniques.

Another type of primitive which is often found in the context of terrain data is contour. In this case, if surfaces are required for rendering, algorithms to generate them from contours are necessary. You should consult a survey of the possibilities in [Stoc87].

The same approach has been used for objects as complex as trees. J. Bloomenthal, combining a procedurally created branching pattern, a polygonal representations for the surfaces of trunks, limbs and leaves, and texture mapping of digitized bark, obtained excellent models for maple trees [Blo085] (Figure 2). Note that the primitives used in this case, generalized cylinders are especially interesting. Bloomenthal later explored the possibilities of implicit surfaces [Blo087] for the modelling of natural forms.

There are several advantages to empirical models. The main one is that often the data is readily available, as for terrains, or could be obtained by automatic or semi-automatic digitization. Given the data, the choice of modelling primitives is relatively free, and therefore they can be easily integrated into a conventional rendering system. Another advantage is that the range of objects amenable to principle to such an approach is enormous.

Drawbacks of such techniques are numerous too. A lot of tedious modeling work is involved if the data has to be acquired. This is not too bad if the model has a long "life", since that has to be done only once, and will be amortized. More seriously, these models can yield very large databases, and they are not very flexible. Here flexibility has two aspects. One is to "customize" the model. This is clearly impossible with terrain data. The other aspect is the problem of levels of detail. Since the data is all there, what has to be done to obtain a less detailed representation is to merge primitives, or to select good representatives. This is a very difficult task for most primitives. It is sometimes done by manually defining the various levels of representation that will be used (more about this later). And finally, the object has to exist and to be available to model that way. This type of model excludes the creation of objects or variants of objects not already found in nature.

3.2. Physical models

A physical model is where one uses directly the equations given by the relevant discipline (physics, chemistry, fluid mechanics, etc.) to generate the object.

The advantages of such a solution is that they require little effort from the modeler (besides understanding the equations), and that they include some truth about the phenomenon modelled. They are also often easy to parametrize by changing the "constants" involved. A very strong advantage is that almost always time is built into the equation, and therefore the solution can directly be used for animation. Of course it might be that the
original motivation to produce pictures is to illustrate the model. In this case, it falls under the category of self-models (see below). The whole area of illustrating physical models for their own sake is now known as scientific visualization. The line between the latter and modelling for computer graphics is sometimes very thin indeed, but I think the distinction is important to make when discussing models.

The drawbacks of physical models are numerous. For any complex phenomenon the equations most likely do not have a closed-form solution, and one is usually confronted with a system of differential equations to be solved numerically at a large number of points in 2-D or 3-D space. The range of phenomena thus modelled is often narrow. To obtain manageable equations, often simplifications have been made which mean that the range of conditions where the results are applicable is limited. For example in the case of waves there are equations for "deep" waves, "shallow" waves and waves in-between. It is difficult to go smoothly from one solution to another. The scale of the solution, that is the size of the phenomena that it computes, might not be sufficient for a realistic display. Either it is too small (it gives the details by not the overall shape) or too big (the rough outline but not the details). Sometimes that can be "fixed" by adding invented details (by stochastic interpolation, by various texture mapping techniques, see sections below), but that is not always easy. And finally physical models give answers to questions that are not asked in computer graphics, such as the pressure, the temperature, the energy balance; and indeed that is why they have been developed. But the cost of obtaining these answers might be much higher than the cost of what we are really after: the look of the phenomenon. For instance, Nelson Max, discussing a suitable model for waves in computer graphics [Max88] rejects the "cycloid" model and adopt Stokes model, apparently partly because the latter generates irrational flow, and thus corresponds better to reality. In this case it obviously is better to have an model which fails to be rotational but looks like real waves, since it is fairly certain that this property has no visual impact.

An illustration of the cost of a physical model solution can be found in a paper by Miyata [Miyas86] where he computes numerically the solution of a model for breaking waves. The velocity and pressure are computed as a function of time in a two-dimensional grid. In typical computations, grid points are 10 mm apart in the X direction and 5 mm apart in the Z direction, and the total dimension is 2.00 m. by 0.40 m, to give 200x78 cells, or 15600. The time slice used, determined by the equations to be solved and the needed precision is about 1.5 ms, that is 10 time increments are needed for each frame at 60 frames/s. Each computation has an inner loop to solve the differential equations by iteration, and this loop is executed about 200 times. Thus the cost of this solution is about 2000 times the cost of a similar closed-form solution if it were available (the cost of the 200 iterations per time step times the 10 extra time steps per frame). It is also one dimension short of what is needed, and would be hard to merge with what happens in deep water on one side and in very shallow water on the other.

A cloud model
Jim Kajiya and Von Herzen [Kaji84] used a model for clouds that was directly based on the differential equations for the water vapor content of the air as a function of altitude, wind velocity and temperature. The advantages of such a model are that the equations are already known and available, and that as we solve them as a function of time the clouds can be realistically animated, since the evolution of the shape of the clouds is "built-in". Another advantage is that the results are in terms of vapor density, which, if a realistic illumination model is available, can be rendered very convincingly. The drawbacks are that the computations involve solving the differential equations, which has to be done numerically on a four dimensional grid (space + time), and is either a low resolution grid or is very costly (or both). Another drawback is the flip side of an advantage, namely that not giving traditional primitives as output, special rendering techniques have to be found or developed for it. A look at the pictures accompanying the paper shows that a fairly low resolution cloud takes over two hours of an IBM 3431 to compute and render. Less obviously, the interesting shape of the cloud is not given by the equations themselves, but by the initial wind pattern given to the model. So the initial shape is in effect modelled "by hand".

The future of physical models
The balance sheet will change as more powerful computers and physical models become available. Computational cost will become less of a drawback, but some of the other problems will remain. The true complexity (how does the work increase as a function of the size of the problem) will then become a more important issue as we attempt to use physical models at the level of details suitable for realistic rendering.

Another factor is the development of massively parallel computers (such as the connection machine [Hilli85]) that conceivably will change the way physical models are stated (away from analytical differential equations, and directly in terms of interactions between "elements"). Then more computer graphics models might be developed in a similar way.

3.3. Morphological Models
Morphological models are models which gives directly the shape of the model. A trivial example is a sphere for a soap bubble. Most traditional models are morphological models. When we use a bicubic B-spline surface to model the hood of a car, there is no claim to model anything about the part except its shape. To model the hood of a car, there is no claim to model anything about the part except its shape.

An old (by computer graphics standards) and fascinating book is the aforementioned Of Growth and Form by D'Arcy Thompson [Thomp51]. In it, the author proposed, and often succeeded, to explain many of the forms invented details (by stochastic interpolation, by various texture mapping techniques, see sections below), but that is not always easy. And finally physical models give answers to questions that are not asked in computer graphics, such as the pressure, the temperature, the energy balance; and indeed that is why they have been developed. But the cost of obtaining these answers might be much higher than the cost of what we are really after: the look of the phenomenon. For instance, Nelson Max, discussing a suitable model for waves in computer graphics [Max81] rejects the "cycloid" model and adopt Stokes model, apparently partly because the latter generates irrational flow, and thus corresponds better to reality. In this case it obviously is better to have an model which fails to be rotational but looks like real waves, since it is fairly certain that this property has no visual impact.

An illustration of the cost of a physical model solution can be found in a paper by Miyata [Miyas86] where he computes numerically the solution of a model for breaking waves. The velocity and pressure are computed as a function of time in a two-dimensional grid. In typical computations, grid points are 10 mm apart in the X direction and 5 mm apart in the Z direction, and the total dimension is 2.00 m. by 0.40 m, to give 200x78 cells, or 15600. The time slice used, determined by the equations to be solved and the needed precision is about 1.5 ms, that is 10 time increments are needed for each frame at 60 frames/s. Each computation has an inner loop to solve the differential equations by iteration, and this loop is executed about 200 times. Thus the cost of this solution is about 2000 times the cost of a similar closed-form solution if it were available (the cost of the 200 iterations per time step times the 10 extra time steps per frame). It is also one dimension short of what is needed, and would be hard to merge with what happens in deep water on one side and in very shallow water on the other.

A cloud model
Jim Kajiya and Von Herzen [Kaji84] used a model for clouds that was directly based on the differential equations for the water vapor content of the air as a function of altitude, wind velocity and temperature. The advantages of such a model are that the equations are already known and available, and that as we solve them as a function of time the clouds can be realistically animated, since the evolution of the shape of the clouds is "built-in". Another advantage is that the results are in terms of vapor density, which, if a realistic illumination model is available, can be rendered very convincingly. The drawbacks are that the computations involve solving the differential equations, which has to be done numerically on a four dimensional grid (space + time), and is either a low resolution grid or is very costly (or both). Another drawback is the flip side of an advantage, namely that not giving traditional primitives as output, special rendering techniques have to be found or developed for it. A look at the pictures accompanying the paper shows that a fairly low resolution cloud takes over two hours of an IBM 3431 to compute and render. Less obviously, the interesting shape of the cloud is not given by the equations themselves, but by the initial wind pattern given to the model. So the initial shape is in effect modelled "by hand".

Figure 3. Parameters affecting the shape of some shells (according to D'Arcy Thompson).
Pickover of IBM T. J. Watson, among others, generated excellent pictures of shells, using these or similar parameters. It is interesting to note that the same range of shapes can be expressed as transformations: tapering, twisting, bending (see Figure 4) have been used also for the modelling and rendering of artificial objects [Barr84].

Y. Kawaguchi, also partially inspired by D'Arcy Thompson, has built a remarkable series of images and animations around these techniques to create interesting shapes. While his goals are not to reproduce reality, but to emulate the variety and complexity of natural forms, his work fits well within this section [Kawa82] (Figure 5).

These types of models are fairly easy to implement, and can easily be made to use traditional primitives. Most of the "naive" models of computer graphics fall under that category. There are however limited to a small range of shapes and phenomena, those with regular features. There are also hard to animate, and therefore more suited for static pictures.

D'Arcy Thompson also defines an ingenious series of non-linear transformations to go from one shape to another related one. These transformations are underused in computer graphics, and should be investigated as an additional modelling tool (see Figure 6).

Similar transformations are described and applied in [Sede86].

3.4 Structural Models

In structural models, as the name implies, only the structure of the object is given. The rest, that is how this structure is actually realized in our 2-D or 3-D space, and what is the surface appearance, is usually left to "free" interpretation. Models such as grafels [Smith86] and most grammar-based models belong to that category. Maybe less obviously a graph description of a polyhedron (as in edge/vertex adjacency graphs) belongs to this category. In their simplest form structural models give only the topology of the object modelled. While in other applications this is the main feature of the object (when modelling relationships with graphs, for instance), in some it is only the bare bones.

The advantage of these models is especially obvious when the objects modelled have a strong structure, such as for trees, and plants in general. Sometimes the grammar can be coaxed into helping in the geometric interpretation. That is what Prusinkiewicz did with turtle interpretation for L-systems [Prus86] and Oppenheimer did with a fractal process for trees [Opp86].

The drawback is mainly in the fact that this geometric interpretation is necessary at all.

Plant models from botanical knowledge of structure

Based on the work of a botanist/agronomist, Philippe de Reffye [Reff79], a group of researchers [Reff88] designed and implemented remarkable plant models. At the core of the growth of plants are the meristems, specialized tissues of a bud. De Reffye's approach is to characterize and simulate the functions of the meristems. Their activity can be resumed into three stages: growth, ramification and death (no more activity). The knowledge of the rates of growth and of the timings of the ramification and death events (both are in general associated with some probability distribution) leads to a very versatile model of growth for plants, and therefore static models of plants at various stages of their development as well. In fact most of the known architectural types for plants, some repertoried for tropical trees in [Hall78], can be generated by this technique. Figure 7 shows the variety of results from the simulation.

The same approach has been applied to leaves and flowers. In this case of course the primitives should be surfaces instead of branch segments. But P. Lienhart and J. Francon [Lien87, Lien87a] reduced the problem to the generation of the topology of the object by operations on graphs which are then translated into the proper geometry.

Prusinkiewicz and Lindenmayer developed further the use of L-systems to simulate the mechanisms of herbaceous plant growth [Prus88]. The modelling still has two stages, the structural stage controlled by L-systems, and the geometric stage controlled by a geometric interpretation of the structure. Knowledge about the mechanisms controlling growth in real plants has been used to build the relevant grammars.

3.5 Impressionist Models

There are model without any claim to a physical connection with the phenomenon modelled, or to a close geometric rendition of the shape of the surface of the object modelled, but whose only concern is with the impression they make. We will call them impressionists because of the similarity with the aims and methods of the Impressionist painters, and because G. Gardner likes the analogy1.

1. Actually many impressionist painters claim adherence to scientific principles, especially as they apply to light and vision; we will not get here into a debate about subjectivism vs objectivism, or the nature of truth. This is only computer graphics.
Quadric surface models
Clouds and trees have been modelled by G. Y. Gardner [Gard84, Gard85], (see Figure 8).

Figure 8. Clouds from [Gard85] (see colour pictures)

Gardner's models are made of simple primitives giving the general shape, generally a quadric (ellipsoid, hyperboloid or paraboloid), and procedurally generated textures to vary both the colour and the transparency. Thus even though the underlying volume is very smooth and regular, and allows fairly easy geometric transformations and visibility computations, the overall impression is of an object complex in silhouette and colour. The technique has so far been used to model clouds, trees and terrain. The advantages of the techniques are numerous. Since the geometric primitives are few and relatively simple, geometric operations are not too much of a burden. Since the details are assumed by the synthesized textures, they can be easily varied for flexibility, and can be "prefiltered" to avoid aliasing problems. The drawbacks are that the quadrics used are not common primitives, and special scan conversion and visibility algorithms had to be developed and have to be implemented. It is also difficult to generate recognizable species of trees. But overall it is a very efficient method, and while it does not yet allow real-time display, it permits the computations of realistic scenes in minutes of Vax 11/780 time.

A similar technique, with cheaper primitives, has been used by Fournier and Grindal [Four86]. The geometric primitives are convex polyhedra, and the textures are three dimensional texture maps. The algorithms are designed for fast modelling and rendering at the level of the frame buffer.

Impressionist models are good when the appearance is the only concern. Evans & Sutherland CTS, trees are represented by a few polygons, whose shape is modulated by texture masks, whose edges are softened by transparency texture, and whose colour are determined by textures. It should be noted that these models are especially successful when they are accompanied with good texture modelling and texture mapping techniques.

Their drawbacks are that they have to be designed by trial and error, they have limited capabilities for dynamic range (achieved mainly by filtering the texture, or substituting different models, except in the trees of Fournier and Grindal, where dynamic range was one of the motivations), and they are hard or impossible to animate (think of how to animate the bending of branches or the evolution of clouds with these models). Their flexibility is also limited, as mentioned in the case of trees. The parameters of the primitives and the textures are mostly unrelated to the model, and therefore do not help in generating useful variants.

3.6. Self-Models
Self models are where the model just stands for itself. That is the case for all mathematical objects which can be embedded in some geometry. In this regard computer graphics has made possible the display of many mathematical objects that could only be imagined before. The most striking of these are Julia sets, and the Mandelbrot set. The book The Beauty of Fractals, by Peitgen and Richter [Peit86] should be consulted for more details and extraordinary images. I leave it to you to decide whether these objects are "natural" or not.

3.7. An Exercise
Choosing the right type of model is only partly a function of the object being modelled. The intended use of the model also will orient our choice. As an illustration, assume we want to model a sugar cube. For this object, an empirical model is obtained by taking a real sugar cube and determining the coordinates of its corners, using these to generate polygons. A physical model might be derived from information about the crystalline structure of sugar. A morphological model is for instance representing it as a cube (that is our favourite regular polyhedron). A structural model might be using a winged-edge structure to model it as a six-faced polyhedron. An impressionist model might be a superquadric, looking like a cube with rounded corners [Barr81]. We cannot use a self-model, of course, because a sugar cube is not itself a mathematical object.

You should now attempt to choose the best model (within or without the above list) under the following circumstances:

a) an animation of the top view of a breakfast table
b) a documentary on how sugar is made
c) an illustration of the dissolution of a sugar cube immersed in water
d) an animation of the building of a "sugar-cube castle"
e) an explanation of why sugar comes in cubes (as opposed to balls, tetrahedra, cylinders, etc.).

Give the type of model which seems the most appropriate in each case, discuss its advantages and disadvantages, implement it and render it with your favourite renderer.

3.8. Mixed Models
Clearly not every model can fit neatly under the above categories. Most of the time (I hope all of the time) it is because it is made of "sub-models", each one belonging to different types. For example there are models that have some physical basis, but would not be satisfactory within the discipline studying the underlying phenomenon. Thus any model for wave has to have periodic terms, and can be expressed in the form of sum of trigonometric functions. If one of the goals is to simulate refraction, then the known equations related the depth of water to the wavelength for some types of waves can be used. But putting those two together does not necessarily make a valid wave model for hydrodynamics, even though the surface obtained can be very convincing for computer graphics. A structural model for trees can use an empirical model for the leaves, and vice-versa. A texture map from an empirical model (i.e. a real texture digitized) can be used to "bump" map a morphological model, or a structural texture can be used with an impressionist model.

An interesting example of an ambiguous model is when terrains are modelled by a stochastic process such as fractional Brownian motion (see below). This process has only a tenuous relationship with the physical processes which generated terrains, and should qualify as an impressionist model, but one can model different aspects of terrains by changing the dimension parameter, and therefore this goes beyond ordinary impressionist model since one captures some basic property of the terrain.

4. Basic principles and techniques
In the following section we will look at some of the basic principles and tools which have been found useful in the modelling of natural phenomena. These are relevant to all of the types of models described above. First we will look at general principles: data base amplification and variable levels of details. Then we will look at the role of time in a model, and at the use of stochastic elements, that is stochastic modelling. Finally we will look at specific tools: textures, particle systems and fractals.

4.1. Data base amplification
Data base amplification refers to a group of techniques where the original model is rather compact, but allows to generate large amounts of geometric or display primitives. It can be seen as reversing the work in natural sciences, where we try to extract the fewest number of laws and parameters that explain the large diversity of nature. Of course the idea is already apparent in the standard representations of curves and surfaces, where a few coefficients are sufficient to generate a large number of points (most of the time an infinite number of points). The alert reader will object that the problem is precisely that these standard representations generate simple looking objects, smooth and continuous surfaces, the very thing we are trying to avoid. But consider Figure 9. The tree on the right was obtained starting with the two types of segments on the left, and successive applications of two substitution rules. At each step, every segment is replaced by the branching pattern given, suitably scaled, rotated and translated so that their extremities coincide. We have here a simple initial pattern, a simple rule, very easy to describe in a data base, and after a few applications of the rule we obtain a fairly complex looking object. The same techniques can be used to generate textures in two or three dimensions. All these examples can be given as formal grammars, that is a set of symbols and a set of rules that can be applied to transform the symbols. The only new element here is that we want a pictorial interpretation for our string of symbols. There are many ways to achieve this. One is to have the symbols be picture elements to begin with. This is the case in the texture example. The symbols manipulated are patterns of pixels, and these
Another way to generate a picture from a grammar is to add an arbitrary fractal, case are the line segment, and the sequence of four line segments that replace it at each step. The rules implicit in the substitution are that the patterns by themselves make the picture. Another example in the real plane geometric interpretation symbol F is interpreted as a line segment, scaled by 0.5 at each step, the rate a very wide range of textures [Fu80] and remarkable pictures of plants and trees [Smit84, Aon084]. A. R. Smith coined the word graftal, similar to fractal, to name this type of model. The example of Figure 1 is a fractal, and many fractals have an easy representation using grammars (see Figure 9. Rule generated tree [Smi84].

An important condition, not always met, is that at each stage of the application of the rules the object obtained is a "simplified" version of the fully evolved object. This will depend on the grammar. This is true for Koch curve, but not true for the texture grammar given earlier. This technique is often used in association with a stochastic process, where the generated details include a random element (see below and Figure 11).

4.2. Parametrization
A simple example of the second method is with the representation of a circle with polygon. In this case it is easy for the system to determine the number of sides necessary for the polygon by the size of the circle in screen pixels. The parameters of the procedure to create the circle include then the number of sides as a "level of detail" parameter. This is commonly used to compute parametric curves and surfaces by setting the step size in incremental techniques.

4.2.2. Levels of detail
From the figure showing a "Small portion of the world", it can be seen that there is a very wide range of scales in the picture. Some trees can be miles away, and some a few feet away. It is obvious that it is not practical to render all of them at the same level of detail. It is therefore important that the models we use generate primitives at a controllable level of detail.

Models can achieve this flexibility in four main ways; controlling the number of applications of productions in a grammar model, parametrization in a procedural model, separate discrete representations in a fully expanded model, and automatic extraction of representations from the full model.

4.2.1. Controlled generation in a grammar model
An example of the first method is obvious from grammar-based models. An important condition, not always met, is that at each stage of the application of the rules the object obtained is a "simplified" version of the fully evolved object. This will depend on the grammar. This is true for Koch curve, but not true for the texture grammar given earlier. This technique is often used in association with a stochastic process, where the generated details include a random element (see below and Figure 11).

Figure 11. Various level of details by controlled generation (see colour pictures) (at top left is the last "patch" computed, to its right is the total data).

4.2.3. Separate models
The third technique involves the generation of separate databases one for each level of detail needed. This is generally done "manually" (examples can be seen in a paper by Frank Crow [Crow82]). A problem then is to ensure the smooth transition between two levels. Usually the two transformed models are blended, that is the final picture is a weighted sum of both. This has long been the favorite technique in flight simulators.

4.2.4. Automatic extraction of details
The fourth technique, the automatic extraction of different levels of detail, is the most difficult, and still not satisfactorily done for most representations. It would be especially useful in conjunction with empirical models, since it is the only way to obtain variable levels of details with them.

In all techniques the correct levels can be determined globally, that is set for the whole scene (or even the whole animation sequence), or determined adaptively on a local basis. The second method is often necessary, since within the same scene, even the same object, the needed levels can be very different, especially because of perspective. This is all of course strongly reminiscent of filtering, and what we are trying to accomplish is indeed filtering, except that we operate on higher level primitives than pixels.

To really determine which level of detail is sufficient we have to take the characteristics of our visual system into account, and we cannot even start to discuss this here. A challenging problem when implementing this concept is to ensure a smooth transition between the different levels used. This becomes especially acute when different techniques such as texture mapping, bump mapping, displacement mapping, various illumination models (including anisotropy) are used together. Work has been done on various aspect of the integration of these [Kaji85, Bilm78, Pou89, Crow82], but much remains.

2. The same techniques would apply to limit the amount of details necessary at the periphery of the display, taking advantage of the fact that our visual acuity is greater at the center of the visual field, and much less at the periphery.
4.3. Time and Models

It is a truism that if a model is intended for animation, time should be a parameter of the model. But this has to be said anyway, because many proposed models cannot be reasonably animated since their designers ignored this principle.

One example is a wave model. We can obtain from oceanography formulae to represent the frequency or power spectra of certain types of waves [Kim84]. Martin, Waterberg and Macedonia [Mas87] used one of these to generate models of fully developed waves by Fourier synthesis. The still images are very convincing and realistic, but the authors go on to animate the model by shifting frequencies in Fourier space at a speed which are related to the frequencies by a formula which actually should apply to the wave number of "real" waves, not to the frequencies in the Fourier domain. The effect in this case is not necessarily visibly bad, since we are not very good at the visual analysis of this type of motion, but it would be very difficult to model wave trains (which are an important feature of the surface of the ocean) in this way.

Another common example of models where time is absent from the start, and therefore hard to animate are the textures used in Gardner's cloud display, and use the extra dimension as time. This is of course valid only if Time (or more exactly animation) is also very unkind to models. Models of posed models cannot be reasonably animated since their designers ignored parameter of the model. But this has to be said anyway, because many proposed models cannot be reasonably animated since their designers ignored this principle.

The presence of random elements in the model can pose special problems. A stochastic process is any process that generates random variables. Therefore the model for any phenomenon which includes random elements will have to include the simulation of a stochastic process. Fournier and Fussell called stochastic modelling the group of techniques which involve blending ordinary models and stochastic processes [Four82]. The most popular stochastic processes in computer graphics are fractals (see below). The presence of random elements in the model is a major problem. The most important issues are called internal and external consistency in [Four82]. In brief it refers to the necessity to ensure consistency of the model between parts and between invocations at different time and/or different scales. The common answer is to tie the pseudo-random numbers generated to some fixed attributes of the objects. Efficient representation of different levels of detail, and animation at various speeds is useful to keep pseudo-random number generators such that sequences of numbers can be skipped while maintaining repeatability of the overall sequence.

4.5. Textures

A texture is simply a function of space, \( F(x,y) \) in two dimensions, or \( F(x,y,z) \) in three. The function can have a scalar value (a single value) or a vector value, like a three dimensional colour vector. In most applications the coordinates \( x,y \) or \( x,y,z \) are discrete coordinates (so-called texture space), and we can look at the texture as a discrete array of values, indexed by the coordinates.

The use of textures in computer graphics in general, and in the modelling of natural phenomena in particular, has two aspects: the modelling of textures, and their application, the latter usually known as texture mapping. Since we are mainly discussing modelling, we will refer you to an excellent survey paper by Heckbert [Heck86] on the topic of texture mapping. When textures are used to model natural phenomena, the whole range of techniques used for other kinds of objects also applies to them. In this particular case, digitized pictures of real objects, or real textures, are easy to use as source of texture, since both they and the texture are two-dimensional images. This is of course just an easier version of an empirical model. The ubiquitous mandrill is a classic example of this technique.

Procedurally generated textures are also very useful. Their main advantage is that they can be generated at controllable levels of detail, they can be band-limited to avoid aliasing problems, and they usually require few parameters. The papers by Perlin [Per85] Peachy [Peach85] Norton al [Nort82] and Gardner [Gard85] should be consulted for more details. Recent work has concentrated on solid textures, with spectacular results on "furry" textures [Kaji89, Per89].

An important and powerful variation of the empirical approach consists in using a parameterized texture model, and applying a fitting procedure to simulate a given real texture with some minimum error criteria. This was especially developed by Ma and Gagalowicz [Gaga86] They first assume (from experimental evidence about our ability to discriminate textures) that a small set of grey levels is enough. Then they assume that for most textures the domain where the various statistics have to be approximated is rather small (in terms of field of vision, corresponding to a few degrees of solid angle). They use statistics such as the autocovariance, the histogram and the various moments of the distribution of grey-scale values. Once those statistics are computed for a given texture, the procedure is to generate white noise with the required histogram, and then iteratively modify each pixel to minimize the error on each of the statistics used. The models are very successful for most textures without strong macroscopic features. The main drawback is that the number of parameters is rather large (from 300 to 2000 according to the models).

4.6. Particle systems

Particle systems were introduced by W. T. Reeves, first to model fire, then to generate some of the most realistic images of forest produced so far [Reev85, Reev83]. Particle systems are thus a new a powerful type of modelling primitives. In particle systems the basic primitives are points (particles). The creation, destruction and trajectory of these points is controlled according to the characteristics of the objects to be modelled. An object is then represented by these particles, either from their position at a given time, as in the case of fire, or fireworks, or as part of their trajectories, as in the case of grass and trees. For most applications stochastic elements will be added to introduce the necessary variations. In unstructured applications, there will be no deterministic parameters. In the case of trees, there are between 50 parameters, governing branching angles, lengths of branches, etc. One important advantage of particle systems, beside the database amplification property they share with other methods, is that since the geometric primitives are points, they are easy to transform, and it is easy to filter, both in space and time, their trajectories. Thus antialiasing and motion blur can be applied to them fairly easily. Economically the main drawback is the relatively large number of primitives needed at any given time, and the fact that designing with them is a trial and error process, and shading calculations have to be specialized. As a timing example, it took 10 hours of a Vax
The fact that particle systems have been used in most of the types of models described above is a good illustration of the interplay between types of models and primitives.

4.7. Fractal models
As mentioned in the discussion about levels of detail, some natural objects seem to have details with details, and so on to (apparently) infinity. Another way to look at the phenomenon is to consider most classical mathematical functions. The definition of the derivatives of a function involves taking the limit of the tangent to the curve or surface representing the function. If we try this process with a curve representing a coastline, for instance, the "limit" does not seem to go anywhere in particular. From such observations, and from the existence of rigorously defined mathematical objects that shared this paradoxical properties, Benoît Mandelbrot [Mandel82] introduced the concept of fractal, and in so doing a whole new way to look at some mathematical objects and some natural phenomena that can be described by these objects.

The universe of fractals is by now very large, but for our purposes here they can be divided into deterministic fractals, such as the Koch curves of Figure 16, and stochastic fractals, where the fractal properties apply to the various characteristics of random variables. We will only discuss briefly one such stochastic fractal process, *fractional Brownian motion* (fBm). It has been introduced by Mandelbrot and Van Ness [Mandel86] as a generalization of Brownian motion, which itself has fractal properties. It is relevant here because, as a first approximation, it is a useful model for terrain. How can we show this? There are two possible ways, and they again illustrate the different kinds of models. We can compute samples of fBm, and display them rendered to simulate terrain colour and shading, and check visually if they are satisfactory. We can confirm this check by measuring the statistical properties of real terrain, and see if they correspond to fBm characteristics. The other way is to build a model for the creation of the surface in terms of fBm. For the first way, we will let you be the judge. For the second one, the results are mixed at best. Terrain certainly has fractal characteristics, but not through the whole measurable range. It is not very surprising after all. There are many forces and phenomena at work in the shaping of terrain, from plate tectonics to rain fall, and it is too much to ask that one single mathematical process can model the sum of their effects. As for the third one, there are no satisfactory model to date, for exactly the reasons just mentioned: there are too many factors at work. An interesting model, however, was given by Mandelbrot as a sum of randomly distributed faults.

There are various algorithms to compute approximations to fBm. One class, introduced by Mandelbrot and Voss [Voss85], consists in generating white noise, and filtering it so the results has the frequency distribution characteristic of fBm. The other, used by Fournier, Fussell and Carpenter [Fourn82], uses recursive subdivision to successively add details with the required distribution (see Figure 13).

Fractal processes have been used not only to model terrain, but also to model clouds [Kaji84, Voss85], water, texture for trees, fire, etc. This has been done without necessarily verifying first by analysis if the statistics of the natural objects to be modelled are those of a fractal object^3.

The main advantage of fBm as a model of terrain is a remarkable compactness of representation. Depending on how much deterministic data is included, the data base can be from two numbers to a few hundreds, to represent terrain that ultimately contains thousands or million of polygons. The second big advantage, due to its fractal nature, is that unlimited amounts of details can be generated. The disadvantages include the fact that to generate a surface pure recursive subdivision is not sufficient, and that will complicate somehow the subdivision algorithms, and that it has limited flexibility, with basically only one parameter to be adjusted to generate different terrains. As an example, the terrain of Figure 14 has been generated with recursive subdivision in 20mn of Vax 11/780 time.

Methods for Stochastic Subdivisions and Applications
A straightforward way to generate approximations of stochastic fractals (and of other stochastic processes) is stochastic subdivision. There have been numerous methods published for stochastic subdivisions. The criteria to evaluate them have to depend on what they claim to accomplish and the use they are to be put. The techniques to approximate specifically fractional Brownian motion [Four82, Voss85] have been well described. Two papers [Pipe84, Four85] describe "cheap" ways to implement stochastic subdivision and to render the resulting data in a "near-real-time" environment. Some of the problems of stochastic subdivision (creases, "bubbling" in animation) have been addressed, and largely solved by John Lewis [Lewis86, Lewis87]. In Lewis' generalized stochastic subdivision, each new interpolated value is computed by adding noise of known variance to the weighted sum of the current values in a neighbourhood of modest size. The method permits the approximation of a wide range of processes, Markovian as well as non-Markovian, and even oscillatory. The choice of the correct size of neighbourhood is related to the process to be approximated, and the papers should be consulted for the details. Lewis also has a recent paper [Lewis89] which contains useful methods for the generation of three-dimensional textures by stochastic interpolation.

Another paper by Miller [Mill86] proposes another solution (while slightly misrepresenting the problem) which can be interpreted as smoothing over the already generated values with a small filter. Voss [Voss85] has also developed a recursive algorithm which involves modifying the values computed in the preceding steps to maintain the desired statistics.

---

3. Also in the class of fractals are objects such as Julia sets and Mandelbrot sets, already mentioned.
5. A Case Study: A Simple Wave Model

By wave model we mean a model for the surface of the ocean. As mentioned before, a simple model without much physical motivation can be made with one or more sine functions. It is clearly unsatisfactory because it fails to represent two essential aspects of the ocean. The first is that there is a lot of randomness on the surface of the ocean. Even when individual waves are visible, there are generally short crested and short lived. The other is that the surface is characterized not only by the overall shape, but by its transformations under the influence of the wind, the depth and the shape of the bottom.

A model that includes most of these aspects was developed by Fournier and Reeves [Four86a]. It uses an old model of waves which describe the surface through the motion of the molecules of water. They are assumed to describe stationary orbits which are circular or elliptical. The parameters of the orbits and the speed of motion around the orbits are affected by the depth and the wind. The necessary randomness is introduced through wave trains, with large variations between trains and small random variations within trains. The model generates a grid of points for each given time, and these can be rendered using most traditional rendering methods. The model can determine the position and direction of spray and foam, and these are rendered with particle systems. The picture of Figure 15 shows an example of waves and foam rendered by this model.

![Figure 15. Waves at the beach (Folke86) (see colour pictures)](image)

The advantages of such a system are that the original database is quite small, with a few dozen numbers per wave trains, and the output is compatible with most rendering methods. The animation is also "built-in", since time is part of the orbital equations. The main disadvantage is that it requires too much computation real-time applications with current hardware.

The whole model and its associated rendering techniques embody many of the concepts discussed in this paper. It has aspects of empirical modelling: the relationship between wind field and wave heights is from empirical observation. It has several physical aspects: the original model is derived from particle motion by Geestner [Gero99] and Rankine [Rank63], and the foam and spray, modelled by particle systems, follow physical laws of motion under the influence of gravity and viscosity. The "wind effect" on the shape of the wave is purely morphological, and the clouds in the background are impressionist, having been painted by hand. A bump-mapped texture was used for the small scale waves, and is purely impressionist, having been "recycled" from a texture for stained glass used in the "Young Sherlock Holmes". There are stochastic elements in the distribution of the wave heights, wavelengths and periods, and there are even fractal elements, since the terrain (a morphological model) was designed interactively and then "fractalized" by stochastic interpolation. The structural type is represented by the wave trains, which serve as "boxes" of waves to design the overall look of the sea.

6. Conclusion

To recapitulate the wide range of models available, the following tables gather several models that have been used for trees and ocean waves. A fairly subjective evaluation of their qualities is given (on a scale of 0 to 5, where 0 is "no way", and 5 is "can't be beat"), in addition to the type and basic characteristics of the models. For mixed models I indicated the type which seems to be dominant. Of course this only represents the opinion of the author.

Two questions can be asked in conclusion. When will computer generated images pass the "Turing test"? That is, when will computer generated pictures be indistinguishable from pictures of real scenes? The answer of course must depend on what is in the scene. For artificial objects, some computer pictures already can "fool" must of us. For pictures of some natural objects, such as plants, some trees and some waves, we can fool some of the people some of the time. For scenes with humans or animals in them, we are not even close. An educated guess is that it will take five to ten years for convincing natural scenes on a grand scale, and longer for convincingly animated live figures. It is important to note that even when

---

### Table 1. A comparison of various tree models

<table>
<thead>
<tr>
<th>Reference</th>
<th>Type</th>
<th>Realism</th>
<th>Easy</th>
<th>Flexible</th>
<th>Compact</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Max81]</td>
<td>Physical</td>
<td>3.5</td>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>[Peac86]</td>
<td>Morphological</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>[FoRe86]</td>
<td>Physical</td>
<td>4.5</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>[Scha80]</td>
<td>Impressionist</td>
<td>3.5</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>[TsBa87]</td>
<td>Morphological</td>
<td>3</td>
<td>3.5</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>[MaWM88]</td>
<td>Empirical</td>
<td>4</td>
<td>2.5</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2. A comparison of various wave models

<table>
<thead>
<tr>
<th>Reference</th>
<th>Type</th>
<th>Realism</th>
<th>Easy</th>
<th>Flexible</th>
<th>Compact</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Four86a]</td>
<td>Empirical</td>
<td>4.5</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>[ReBI85]</td>
<td>Empirical</td>
<td>4.5</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>[Gard84]</td>
<td>Impressionist</td>
<td>3.5</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>[BloC85]</td>
<td>Empirical</td>
<td>4.5</td>
<td>2.5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>[Bblo85]</td>
<td>Empirical</td>
<td>4.5</td>
<td>2.5</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

---

Acknowledgements

This paper is the result of wondering about what I have been doing these past few years. I gratefully acknowledge the support of NSERC, which through operating and other types of grants, let me wonder.
Figure 1. A small portion of the world

Figure 2. The mighty maple [Bloo85]

Figure 5. An image by Kawaguchi

Figure 7. Some plants simulations from [Reff88]

Figure 8. Clouds from [Gard85]

Figure 11. Various level of details by controlled generation

Graphics Interface ’89
Figure 12. Andre’s forest [Smit84]

Figure 14. A fractal terrain model

Figure 15. Waves at the beach [FoRe86]