



Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

H.A. Lauwerier, J.A. Kaandorp

Fractals (mathematics, programming and applications)

Computer Science/Department of Interactive Systems

Report CS-R8762

December

The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

Fractals (Mathematics, Programming and Applications)

H.A. Lauwerier

*Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands*

J.A. Kaandorp

*Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands
Current address:
Department of Computer Science, University of Amsterdam,
Kruislaan 409, 1098 SJ Amsterdam, The Netherlands*

This tutorial will consist of three subjects. The first subject is the mathematics of fractals, several classes of fractals (Cantor sets, Koch's curve, Levy's curve, Mandelbrot and Julia sets) are discussed together with self-similarity and fractal dimension. The second subject is the generation of fractal objects, in which several methods are discussed for creating fractals (formal languages, Iterated Function Systems, geometric construction of fractals, non-linear complex mappings). The last subject is the application of fractals for modelling objects from nature. In this subject the estimation of the fractal dimension and simulation of objects from nature are discussed.

1980 Mathematics Subject Classification: 58F13, 69K30, 69K34.

Key Words & Phrases: fractals, self-similar sets, non-linear complex mappings, fractal growth.

Note: This paper has been submitted for publication elsewhere and was presented as a tutorial on fractals at Eurographics '87.

1. INTRODUCTION

Fractals find their origin in the second half of the 19th century, when G. Cantor laid the foundation of the modern concept of numbers. He gave an example of a curious point set nowadays called "Cantor dust" or the Cantor fractal. It is the prototype of a large class of fractals consisting of an infinite number of loose points. At the beginning of the 20th century, H. von Koch produced a geometrical construction of a continuous curve, for which there is no tangent in any of its points. Such examples were long considered as monstrosities, good for intimidating beginning students in mathematics. But in the present time they are considered useful objects, with a wide range of applicability. B.B. Mandelbrot did pioneering work. He defined a fractal as a geometrical object with a fractal dimension and coined the word "fractal" in his book "The fractal geometry of nature" [22].

Many objects out of nature, in contrast with man-made objects, show a high degree of irregularity, non-smoothness and fragmentation. These objects cannot easily be described with traditional modelling techniques using spheres, lines, circles etc. The fractal dimension can be demonstrated for several of those objects (for example the length of a coastline, surface of the brain, the lung, the vascular system and trees). Fractals form a powerful tool generating "objects" which show a high resemblance with natural phenomena. In many other areas of science than computer graphics and mathematics, like physical geography, chemistry, biology, physics, fractals may be applied for modelling complex natural objects.

The object of this tutorial is to give some mathematical background helpful for understanding the geometry of fractals, to discuss some methods for the generation of fractals and to give some simple examples of the application of fractals.

Report CS-R8762
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

2. MATHEMATICS OF FRACTALS

We define a fractal as a self-similar set with respect to a class of similarity transformations. We consider mainly deterministic fractals in contrast to random fractals. In a deterministic fractal the similarity transformations T have the property that they transform the fractal into a part of itself. If T_1 and T_2 are such transformations then also T_1T_2 and T_2T_1 belong to the similarity class. In many cases the class can be generated by two similarity transformations, say L (left) and R (right). They produce compound transformations such as $RL^3R^2LR^3L^2$. Obviously then the similarity class has the structure of a binary tree and its elements can be represented by binary fractions like $.001110110001$ ($L = 0, R = 1$).

2.1. The Cantor set

The Cantor fractal is formed in the following way. From a closed interval, say the point set $[0, 1]$, we delete the open (excluding end points) interval $(1/3, 2/3)$ the so-called middle third. Next we delete the middle thirds of the remaining segments. We go on in this way indefinitely. Eventually a set of endpoints is obtained with remarkable properties. It is totally disconnected, uncountable and yet of zero measure. Fig. 1 gives the corresponding geometrical construction as "Cantor's comb".

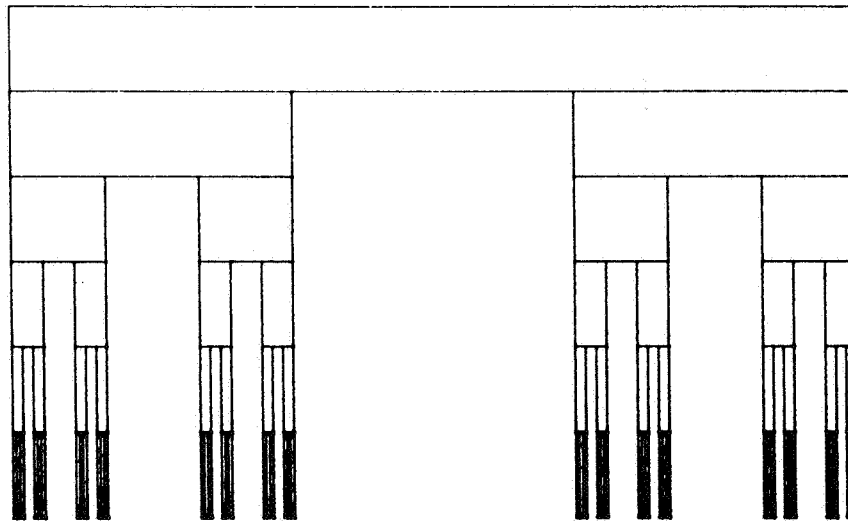


FIGURE 1. Geometrical construction of the Cantor set ("Cantor's comb").

The Cantor set can be described by fractions x written in base 3 as:

$$x = a_1(1/3) + a_2(1/9) + a_3(1/27) + \dots + a_n(1/3^n) + \dots$$

where a_i is one of the three values 0, 1 or 2. The principle of deleting the middle thirds is translated by $a_1 \neq 1, a_2 \neq 1, a_3 \neq 1, \dots$. Thus x is written without 1's. A typical number is:

$$.020020202200020222\dots \text{ (base 3)}$$

The elements of the Cantor set can be brought in a one-to-one correspondence with the binary representation of all real numbers between 0 and 1. For the example given above:

$$.0100101100010111\dots \text{ (base 2)}$$

Thus there are as many points in the Cantor set as there are points on a continuous line, an uncountable collection.

The Cantor set satisfies the two similarity transformations:

$$\begin{cases} L: x' = \frac{1}{3}x \\ R: x' = \frac{1}{3}x + \frac{2}{3} \end{cases} \quad (1)$$

If x is written in base 3 we have:

$$x = .a_1 a_2 a_3 a_4 \dots$$

$$Lx = .0 a_1 a_2 a_3 \dots$$

$$Rx = .2 a_1 a_2 a_3 \dots$$

For a compound transformation like

$$RL^3R^2LR^3L^2$$

(to be read from the right to the left) we have:

$$RL^3R^2LR^3L^2x = .200022022200 a_1 a_2 \dots$$

In this way all points of the Cantor set can be obtained starting from a single starting point such as 0 or 1.

2.2. Koch's curve

Next we consider the curve invented by von Koch. Its construction is similar to that of the Cantor set. Again we start from a single line segment, delete the middle third part and replace it by the two sides of an equilateral triangle as shown in fig. 2.

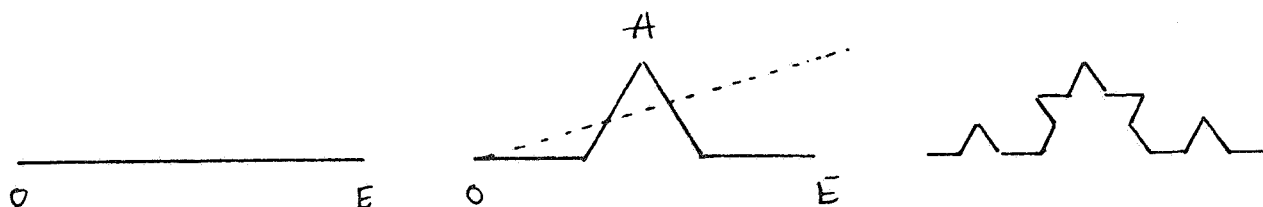


FIGURE 2. Geometrical construction of the Koch's curve.

The construction is continued indefinitely and the result is a continuous line without tangents. In order to describe the self-similarity we may use complex coordinates $z = x + iy$, $\bar{z} = x - iy$. If the line has the endpoints $z = 0$ and $z = 1$, we have of course the similarity transformations (1) of the Cantor set, but there is more. The Koch line satisfies the scaled reflection $z' = c\bar{z}$ where $c = (3 + i\sqrt{3}) / 6$. The axis of reflection passes through O , making 15° with the x -axis. The reflection transformation transforms $E(1)$ into $A(c)$. We note that $|c| = 1/\sqrt{3}$ is the scaling factor of the transformation. There is a similar transformation with respect to E . Together:

$$\begin{cases} L: z' = c\bar{z} \\ R: z' = \bar{c}z + c \end{cases} \quad (2)$$

We note that $L^2: z' = z/3$ and $R^2: z' = z/3 + 2/3$ the scaling transformation of (1). The two similarity transformations of (2) generate a binary tree of such transformations. Starting from

$z = 0$ or $z = 1$ we obtain an infinite number of images by applying all transformations formed by combinations of L and R . Closure of the set, i.e. adding all its limit points, gives the complete Koch curve (fig. 3).

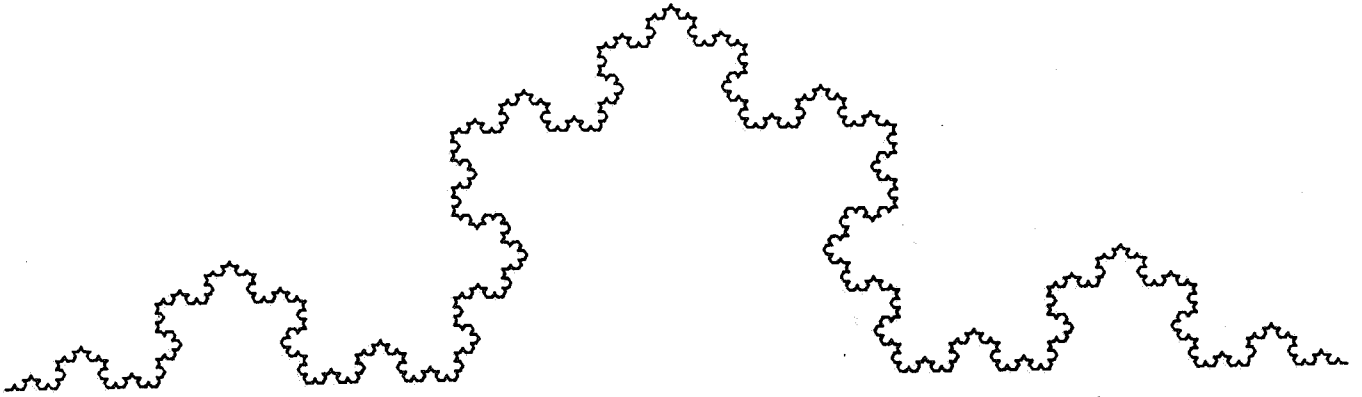


FIGURE 3. Koch's curve.

2.3. Levy's curve

A still simpler example is Levy's curve. Its construction is shown in fig. 4.

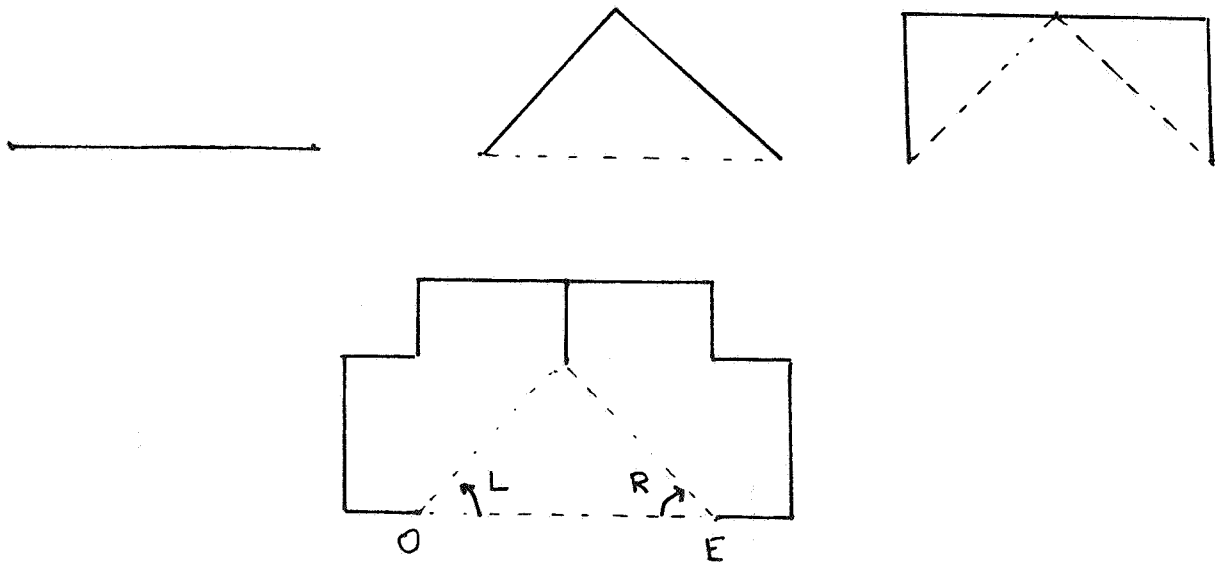


FIGURE 4. Geometrical construction of the Levy's curve

A more advanced stage is shown in fig. 5. Its self-similarity is described by the transformations generated by:

$$\begin{cases} L: z' = cz \\ R: z' = \bar{c}z + c, c = (1+i)/2 \end{cases} \quad (3)$$

both rotations with the scaling factor $1/\sqrt{2}$.

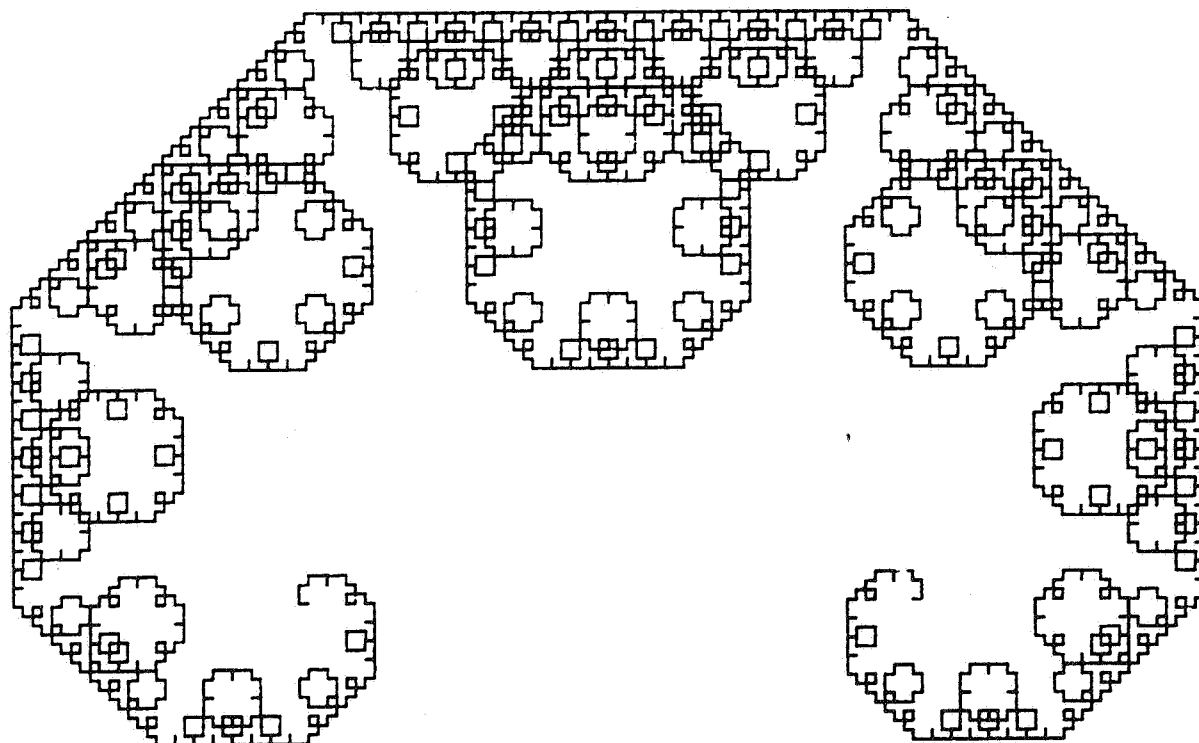


FIGURE 5. Levy's curve

2.4. Self-similarity

These simple examples suggest the following general rule for defining fractals. Let L be a contraction, i.e. a transformation which decreases distances, with the fixed point $z = 0$ and R a similar transformation with the fixed point $z = 1$. Take $z = 0$ (or $z = 1$) as the starting point and subject it to all transformations formed by any combination of L 's and R 's. This defines a point set. After addition of its limit points it becomes a self-similar fractal. L and R may be linear transformations, say:

$$L: \begin{cases} x' = ax + by \\ y' = cx + dy \end{cases}$$

This includes the case of a rotation with scaling ($d = a$, $c = -b$, $a^2 + b^2 < 1$) and a reflection with scaling ($d = -a$, $c = b$, $a^2 + b^2 < 1$). In the general case L is a contraction, provided the eigenvalues of the matrix:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

are inside the unit circle. The way is open to construct a wide variety of fractal forms. (cf. Hata [14]). We give two examples:

$$L: \begin{cases} x' = ax - by \\ y' = bx + ay \end{cases} \quad (4)$$

$$R: \begin{cases} x' = ax - by + 1 - a \\ y' = bx + ay - b \end{cases}$$

L and R are scaled rotations with $a = 0.6$, $b = 0.4$. They are identical but their fixed points are different. The resulting fractal appears to be a space-filling dragon shape (see also next sections fig. 13 and fig. 23).

$$L: \begin{cases} x' = ax - by \\ y' = bx + ay \end{cases} \quad (5)$$

$$R: \begin{cases} x' = ax - by + 1 - a \\ y' = -bx - ay + b \end{cases}$$

L is a scaled rotation with $a = -0.3$, $b = 0.5$. R is a scaled reflection. Further generalizations are possible. We may start with more than two generating transformations and the condition of contracting may be relaxed somewhat.

2.5. Fractal dimension

Mandelbrot and others paid much attention to the notion of the fractal dimension or the capacity of a fractal. It is a number that tells us something about the overall structure of a fractal. It can be shown that the fractal dimension of the Cantor set is 0.6309. For the Koch curve we have 1.2619. A fractal with a fractal dimension slightly less than 2 would mean a "very thick" or "almost plane-filling" fractal line. The fractal dimension (D) may be calculated for a complete self-similar set made up of N equal sides of length r . (r is the similarity ratio) by using the formula:

$$D = \log N / \log (1/r) \quad (6)$$

In most cases the fractal dimension of a given fractal is difficult to obtain by theoretical methods. However, simple computer experiments may give good approximations. The computer method follows closely its mathematical definition. We try to cover the fractal by small squares with the side ϵ . Let $N(\epsilon)$ be the lowest possible number of squares for which this can be done. We repeat the experiment of a sequence of shrinking squares. If ϵ goes to zero the number $N(\epsilon)$ goes to infinity. However, as a rule, the limit:

$$D = \lim_{\epsilon \downarrow 0} \frac{\log N(\epsilon)}{\log (1/\epsilon)} \quad (7)$$

exists and defines the so-called capacity or fractal dimension of the fractal. For a set of a finite number of points the definition gives $D = 0$ the same value of its topological dimension. For a smooth line, such as a circle the definition gives $D = 1$, again in agreement with its topological dimension. For the Cantor set we would find $D = \log 2 / \log 3$, a broken dimension, a true fractal value.

2.6. Julia sets

An active field in the mathematical theory of fractals is the investigation of Julia sets and their corresponding Mandelbrot sets. Those sets may yield colourful illustrations as computer art and they are a source of inspiration in computer graphics. The theory was founded in 1918 by a French mathematician Gaston Julia [17] and has undergone a revival in recent years, due to the efforts of Mandelbrot, who profited from the possibilities of computer graphics (see also [3]). Julia considered an iterative conformal mapping in the plane:

$$z_{n+1} = F(z_n) \quad (8)$$

where $F(z)$ is an analytic function, say a quotient of two polynomials. Such a map preserves angles but its scaling depends on the value of z . Locally the transformation is a scaled rotation with the scaling factor $|F'(z)|$. The standard example is:

$$z_{n+1} = z_n^2 + c \quad (9)$$

where $c = a + ib$. In real notation this can be written in the form:

$$\begin{cases} x_{n+1} = x_n^2 - y_n^2 + a \\ y_{n+1} = 2x_n y_n + b \end{cases} \quad (10)$$

Of special importance are the fixed points and the periodic points of (8). A fixed point z is determined by $z = F(z)$. If $|F'(z)| < 1$ it is stable. If z_0 is close to a stable fixed point z , this means that its orbit $z_0, z_1, z_2, z_3, \dots$ converges to z . We say that z is an attracting fixed point or simply an attractor. If $|F'(z)| > 1$ the fixed point is said to be unstable or repelling. A fixed point for which $|F'(z)| = 1$ is called neutral. An orbit $z_0, z_1, z_2, z_3, \dots, z_{m-1}$ for which $z_m = F(z_{m-1})$ coincides with z_0 is called periodic. If m is the lowest integer for which this happens the orbit is called an m -cycle and z_0 is called a periodic point of order m . For the m -times iterated map $z \rightarrow F(F(F(\dots))) (z)$ an m -cycle becomes a set of m fixed points. So an m -cycle behaves very much like a single fixed point. The m -cycle z_0, z_1, \dots, z_{m-1} is stable when:

$$|F'(z_0) F'(z_1) F'(z_2) \dots F'(z_{m-1})| < 1 \quad (11)$$

and unstable for the corresponding condition with the $>$ sign. For (9) or (10) we have the following examples.

- 1) There are two fixed points $(1 \pm \sqrt{1-4c})/2$. If $c \neq 1/4$ one of them is always unstable. The stability condition is simply $|2z| < 1$. If $c \neq 3/4$ the points $(-1 \pm \sqrt{-3-4c})/2$ form a periodic cycle. The cycle is stable if $|c + 1| < 1/4$.
- 2) In particular for $c = -3/4$ the fixed points are $-1/2$ and $1/2$. The first fixed point is neutral, the second point is unstable. If $c = i$ we consider the orbit of $z = 0$. We find:

$$0 \rightarrow i \rightarrow -1 + i \rightarrow -i \rightarrow -1 + i \rightarrow \dots$$

which gives the unstable 2-cycle $-i \leftrightarrow -1 + i$.

The Julia set of (8) is defined as the collection of all unstable periodic points supplemented with all limit points. It can be proved (see Devaney [9]) that the Julia set $J(F)$ of (8) has the following remarkable properties:

1. $J(F)$ is totally invariant, i.e. if $P \in J(F)$ then all successors and predecessors of P belong also to $J(F)$.
2. If $P \in J(F)$ then all its predecessors give a dense cover of $J(F)$, i.e. any point of $J(F)$ is either a predecessor of P or it is a limit point of such points.
3. $J(F)$ is either a totally disconnected set, or it is a connected set of arcs, or (rarely) the entire plane.
4. $J(F)$ is an attractor of the inverse iterative process $F(z) \rightarrow z$

With the exception of a few very special cases $J(F)$ is a fractal like a Cantor dust, a Koch curve or a dendrite. For the quadratic iterative map (9) and (10) the inverse transformation is two-valued. In complex form:

$$\begin{cases} L: z' = \sqrt{z - c} \\ R: z' = -\sqrt{z - c} \end{cases} \quad (12)$$

We note the striking resemblance in structure with systems like (4). In fact (12) shows that $J(F)$ is a fractal according to our definition. The form (12) can be used in computer programs to obtain effortless approximations of the Julia set of F . There are various techniques available, the simplest being taking a random choice of L and R at each iteration step. More systematically the binary tree

formed by L and R may be followed up to a certain level by backtracking. According to the properties of $J(F)$, it is best to start from the single unstable fixed point $z = (1 + \sqrt{1 - 4c})/2$. The full Julia set is obtained in this way although there may be technical complications. In fig. 6 a few illustrations are given of Julia sets for the quadratic case (9) (cf. Peitgen [27]):

6a) $c = -0.12375 + 0.56508i$, J is a simple closed non-smooth curve forming a fractal island.

6b) $c = i$, J is a dendrite.

6c) $c = 0.32 + 0.043i$, a more complicated but still connected Julia set.

6d) $c = 0.11031 - 0.67038i$, J is disconnected like Cantor dust.

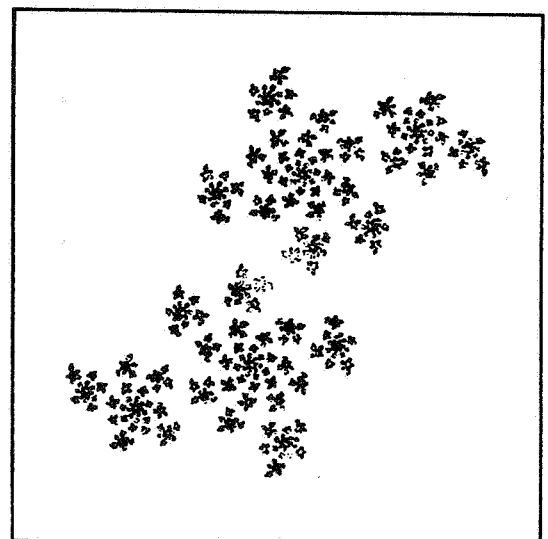
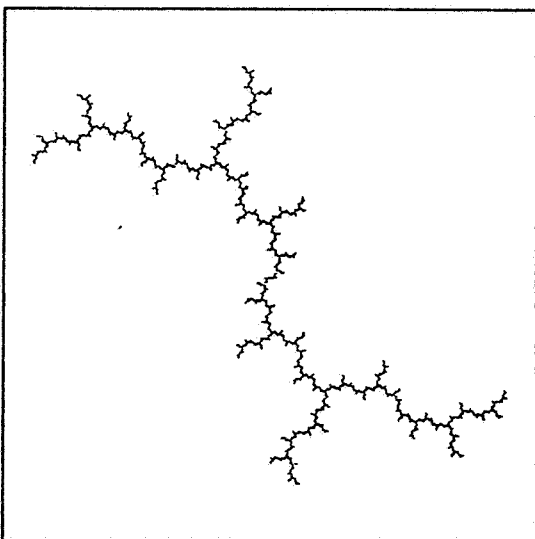
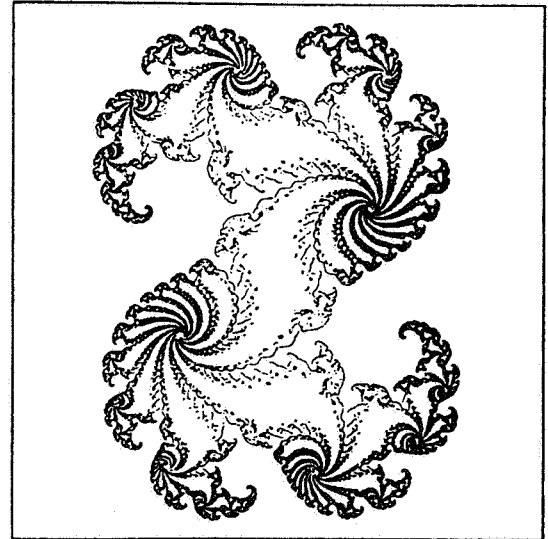
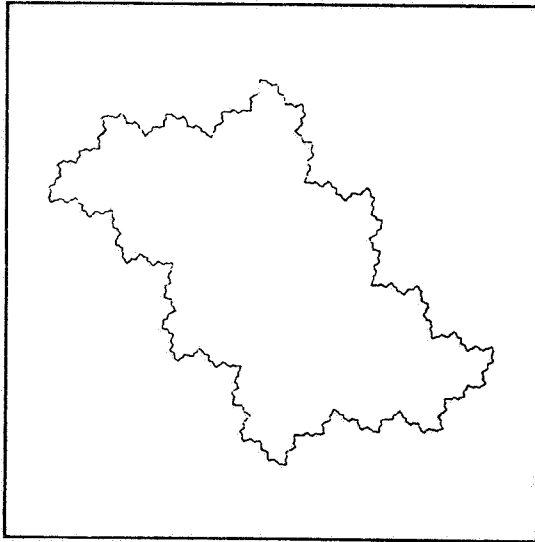


FIGURE 6. Julia sets (see formula (9)), in which the value of c is varied. In the picture in the upper left corner $c = -0.12375 + 0.56508i$; lower left corner $c = i$; right upper corner $c = 0.32 + 0.043i$; lower right corner $c = 0.11031 - 0.67038i$.

2.7. The Mandelbrot set

Mandelbrot had the fortunate idea of making a plot of the points c ($c = a + ib$), for which the Julia set of $z \rightarrow z^2 + c$ is connected. According to a theorem by Julia there exists a simple test. If the orbit of $z = 0$ goes to infinity the Julia set is disconnected otherwise it is connected. The Mandelbrot set is sketched in fig. 7.

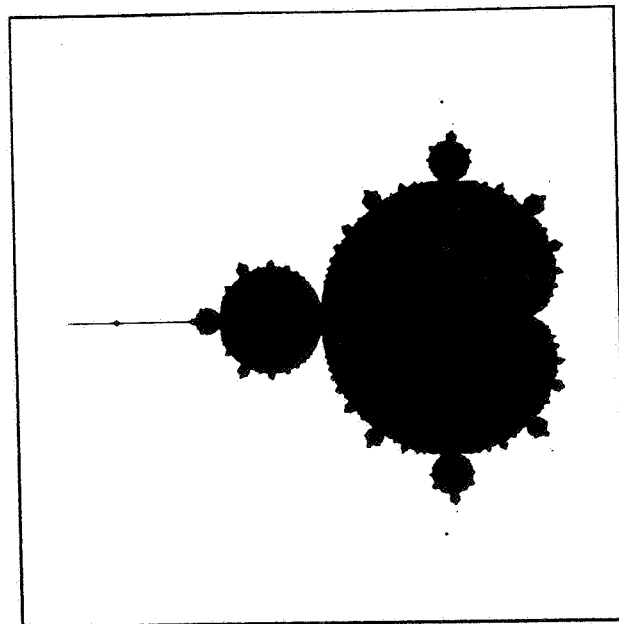


FIGURE 7. The Mandelbrot set

The main body is a kidney-shaped region surrounded by an infinite number of circular regions of decreasing size. A few things can be understood by simple mathematics. The kidney-shaped region corresponds to all c values, for which $z = (1 - \sqrt{1 - 4c})/2$ is an attracting fixed point. At the boundary of the region, the fixed point is neutral, so that we may set $2z = \exp(i\theta)$, $0 \leq \theta < 2\pi$. An elementary calculation gives $c = 1/2 \exp(i\theta) - 1/4 \exp(2i\theta)$ which defines a cardioid. This line intersects the real axis at the points $a = -3/4$ and $a = 1/4$. A still simpler calculation defines the circle at the left-hand side of the cardioid. Inside this circle there exists a stable two-cycle. We have seen before that the stability condition is $|c + 1| < 1/4$. Clearly this defines a circle intersecting the real axis at $a = -3/4$ and at $a = -5/4$. Other details can be given a similar mathematical explanation. However, the Mandelbrot set is still a very complicated mathematical object not yet completely understood. Using computer experiments one believed that the Mandelbrot set was like a continent surrounded by an archipelago of islands, miniatures of the continent. But it could be proved by modern advanced mathematical techniques that the Mandelbrot set is totally connected. Actually the islands are attached to the continent by thin filaments almost undetectable by computer experiments. The Mandelbrot set is a striking example of the interplay between pure mathematical theory and computer experiments. Magnification of small regions close to the boundary of the Mandelbrot set reveal fractal structures with all kinds of self-similarity. In the book by Peitgen and Richter [27], many beautiful pictures in black and white or in colours are shown.

3. THE GENERATION OF (FRACTAL) OBJECTS

In literature several techniques are described for the generation of fractal objects or closely related objects. Basically all these techniques are using feedback processes (fig. 8) in which the output of one iteration is used as input for the next one.

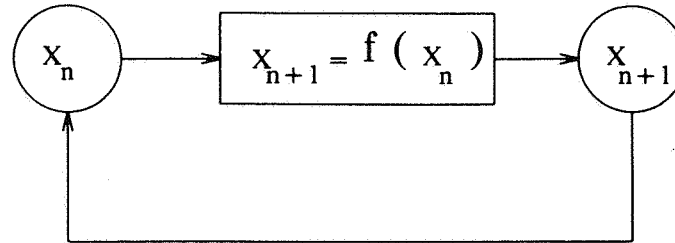


FIGURE 8. Diagram of a feedback process, in which the output of one iteration is used as input for the next one.

The relation between input and output may be linear or non-linear, the result is not necessarily an object with a fractal dimension. The process may also deliver “normal” geometric figures or result in points in infinity or single points of attraction.

In the sections below some of the techniques for creating fractal objects, which are encountered most frequently in literature, are discussed. There are some fundamental differences between these methods, although in many cases these techniques may result (sometimes in a quite surprising way) in the same fractal objects. Probably in a number of cases these techniques can be considered as alternative approaches.

3.1. Generation of fractals using formal languages

Objects, with a high resemblance to plants, may be generated by using formal languages. These languages, which are used for biomorphological description, are known in literature as L-systems ([15], [20]). With L-systems development and branching patterns of filamentous organisms can be formalized and described. In the L-systems strings are generated in a feedback loop. The strings itself do not contain geometric information; in order to translate the strings into a morphological description, additional drawing rules are necessary.

Two examples in which L-systems are applied to the generation of (fractal) objects are the generation of two different branching patterns: monopodial and dichotomous branching. Both branching patterns may be defined as L-systems by using a triple K denoted by $\langle G, W, P \rangle$, in which G is a set of symbols, W is the starting string and P is the production rule. In L-systems brackets are used to denote branches, the brackets contain a branch which is attached to the symbol left of the left bracket.

Monopodial branching may be represented by the following L-system:

$$KM = \langle G, WM, PM \rangle$$

$$G = \{ 0, 1, [,] \}$$

$$WM = 0$$

$$PM = \{ 0 \rightarrow 1[0]0, 1 \rightarrow 1, [\rightarrow [,] \rightarrow] \}$$

$$\langle \text{iteration} \rangle \quad \langle \text{generated string} \rangle$$

1:	0
2:	1[0]0
3:	1[1[0]0]1[0]0
4:	1[1[1[0]0]1[0]0]1[1[0]0]1[0]0

One possibility to visualize the generated strings, for monopodial branching, is to use the drawing rule showed in fig. 9. The generated string, from level 5, is visualized in fig. 10.

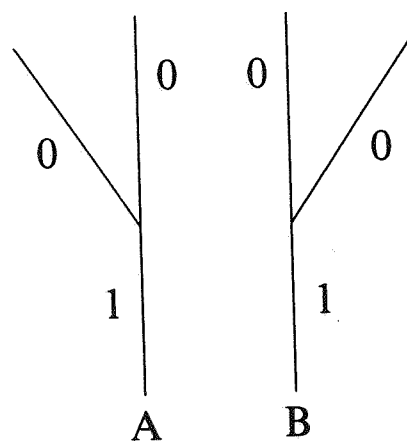


FIGURE 9. Drawing rule for monopodial branching, the string 1[0]0 is visualized, alternating, in shape A and B.

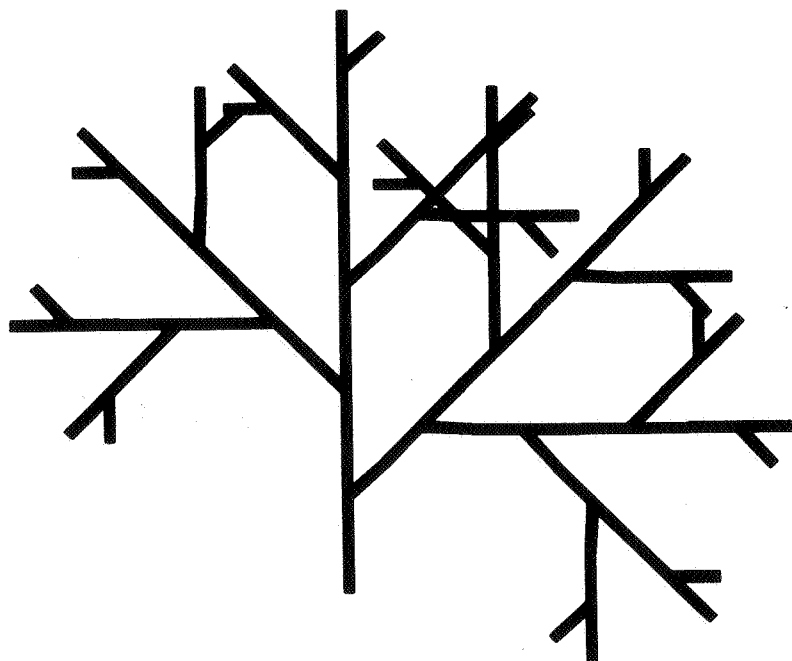


FIGURE 10. Visualization of a generated string, for monopodial branching, from level 5.

Dichotomous branching may be represented by the L-system:

$$KD = \langle G, WD, PD \rangle$$

$$G = \{ 0, 1, [,] \}$$

$$WD = 0$$

$$PD = \{ 0 \rightarrow 1[0][0], 1 \rightarrow 1, [\rightarrow [,] \rightarrow] \}$$

$\langle \text{iteration} \rangle \quad \langle \text{generated string} \rangle$

1:	0
2:	1[0][0]
3:	1[1[0][0]][1[0][0]]
4:	1[1[1[0][0]][1[0][0]]][1[1[0][0]][1[0][0]]]

The generated string may be visualized using the drawing rule showed in fig. 11, the string (from level 5) is visualized in fig. 12.

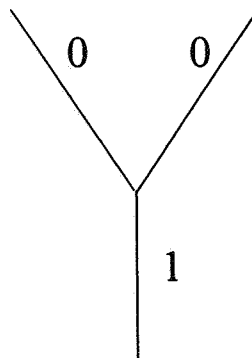


FIGURE 11. Drawing rule for dichotomous branching, visualization of the string 1[0][0].

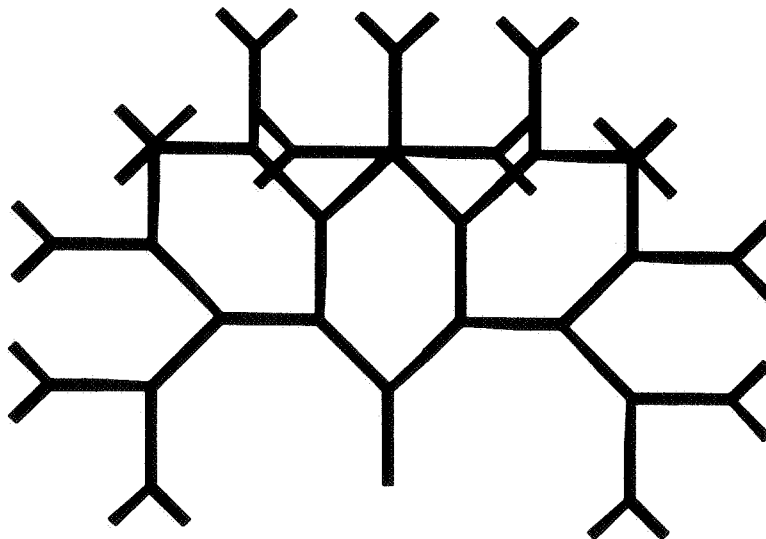


FIGURE 12. Visualization of a generated string, for dichotomous branching, from level 5.

With L-systems growth patterns can be described using string notation, one example of an application of L-systems is given by Aono and Kunii [1], in this study L-systems were supported by 3-D geometric models. Other studies in which L-systems were applied in order to generate fractal (or related objects) are: [31] and [15].

3.2. Generation of fractal objects using Iterated Function Systems

Another method for the calculation and specification of fractal objects is the method based on Iterated Function Systems (IFS, see Barnsley [2]). Iterated Function Systems may be considered as a generalization of formula (2) and (3). With this method a large class of fractal objects may be generated and it may also be applied for modelling objects out of nature.

The first component of an IFS consists out of a finite set of mappings of a d-dimensional space into itself:

$$M = \{ M_1, M_2, \dots, M_n \}$$

The second component is a set of corresponding probabilities:

$$P = \{ P_1, P_2, \dots, P_n \}$$

, in which:

$$\sum_{i=1}^n P_i = 1$$

In a number of cases fractal objects may be generated by randomly choosing mappings out of M . The random walk starts with a point z_0 , a mapping M_i (with probability P_i) is chosen out of M , resulting in $z_1 = M_i(z_0)$. The result of the IFS is calculated in a feedback process. In case the mappings are contractions and the random walk doesn't result into infinity, there is an attractor of the IFS, which might be a fractal object.

One example of the generation of a fractal object is the Dragon Sweep from Mandelbrot [22] This object may be generated using two mappings:

$$M_1(z_{n+1}) = sz_n + 1 \tag{13a}$$

$$M_2(z_{n+1}) = sz_n - 1 \tag{13b}$$

In these mappings z is a complex variable and s a complex parameter:

$$s = \frac{i}{2} + \frac{1}{2}$$

The corresponding set with probabilities is: $P = 0.5, 0.5$. The resulting attractor, the Dragon Sweep, is shown in fig. 13.

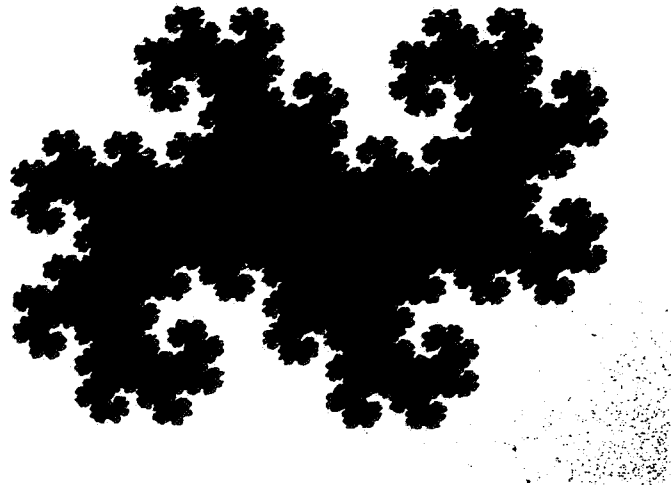


FIGURE 13. Dragon Sweep generated with Iterated Function Systems, using the two mappings from formula (13).

The IFS method may be applied for modelling certain objects from nature, in which a “fitting” attractor is searched for some object. For this purpose it is necessary to find mappings which cover the original object with affine images of itself. After having found the right mappings (which is not always a trivial task) and selecting corresponding probabilities, an attractor may be generated which approximates the original object. An example of this is a leaf of a Black Spleenwort Fern approximated by an IFS, shown in [8].

3.3. Geometric construction of fractals

Many fractal objects may be generated with geometric constructions. The fractals of this class are also known as linear fractals [22], the relation shown in fig. 8 is a linear mapping. In this paper this class will be described as linear fractals or linear mappings; another name used in literature is the, confusing, name self-similar set. The name self-similar set will be avoided, because also non-linear mappings may deliver self-similar sets.

A number of linear fractals can be represented by an initial polygon (the initiator) and a polygon (the generator), which replaces the sides of the initiator (see Mandelbrot [22]). With those two components production rules for some linear fractals may be formulated. In order to enter those production rules in an algorithm generating the fractal objects, the geometric information should be stored in data structures. The data structures discussed below for representing production rules refer to data structures defined in 2D C-GKS ([16], [12]). In this section only the data structures

for representing production rules and the algorithm for generating fractal objects out of the class of linear fractals are discussed. Some more information about the generation of linear fractals can be found in [18].

An advantage of geometric construction of fractal objects is that geometric production rules can be designed interactively in a, relatively, easy way. This may be a great advantage for designing models, which describe fractal growth. In a later section this aspect will be discussed.

3.3.1. Production rules and data structures necessary for representing fractals

One of the quadric Koch curves is represented by the production rule in fig. 14. The data structure necessary for representing the initiator (and all succeeding iteration steps) and the generator is very simple and consists only of two fields: the number of world coordinates and a pointer to the world coordinates (the third component base element will be discussed later).




initiator	generator	base element
		

FIGURE 14. Production rule of a quadric Koch curve. The resulting fractal is shown in fig. 15.

The linear fractals can be classified on the basis of the data structures minimally necessary for representing all components of the production rule. In fig. 15 a possible classification is shown of some linear fractals. In the classification diagram of fig. 15 the quadric Koch curve is the most primitive fractal.

An additional rule may be introduced in the production rule. By defining some of the sides of the initiator "fertile" or "not-fertile", this rule controls which of the sides participate in the next iteration. In the production rule for a simple tree (fig. 16) it is necessary to define the fertilisation state of the sides of the initiator and generator. In the diagram (fig. 15) the data structure for representing the simple ramiform fractal is extended with a new field: a pointer to boolean values, which determine the fertilisation state of the sides of the two components of the production rule.

A new class of ramiform fractals may be created by introducing functions, which randomize the original generator in the production rule. A random processing function which will generate a ramiform fractal with a more irregular appearance is, for example, a function which allows the original generator to make random movements between two limits (see fig. 17). The data structure of the generator is extended by a field for a random processing function. With the introduction of random influences in the construction of fractal sets, the so-called non-deterministic fractals may be created.

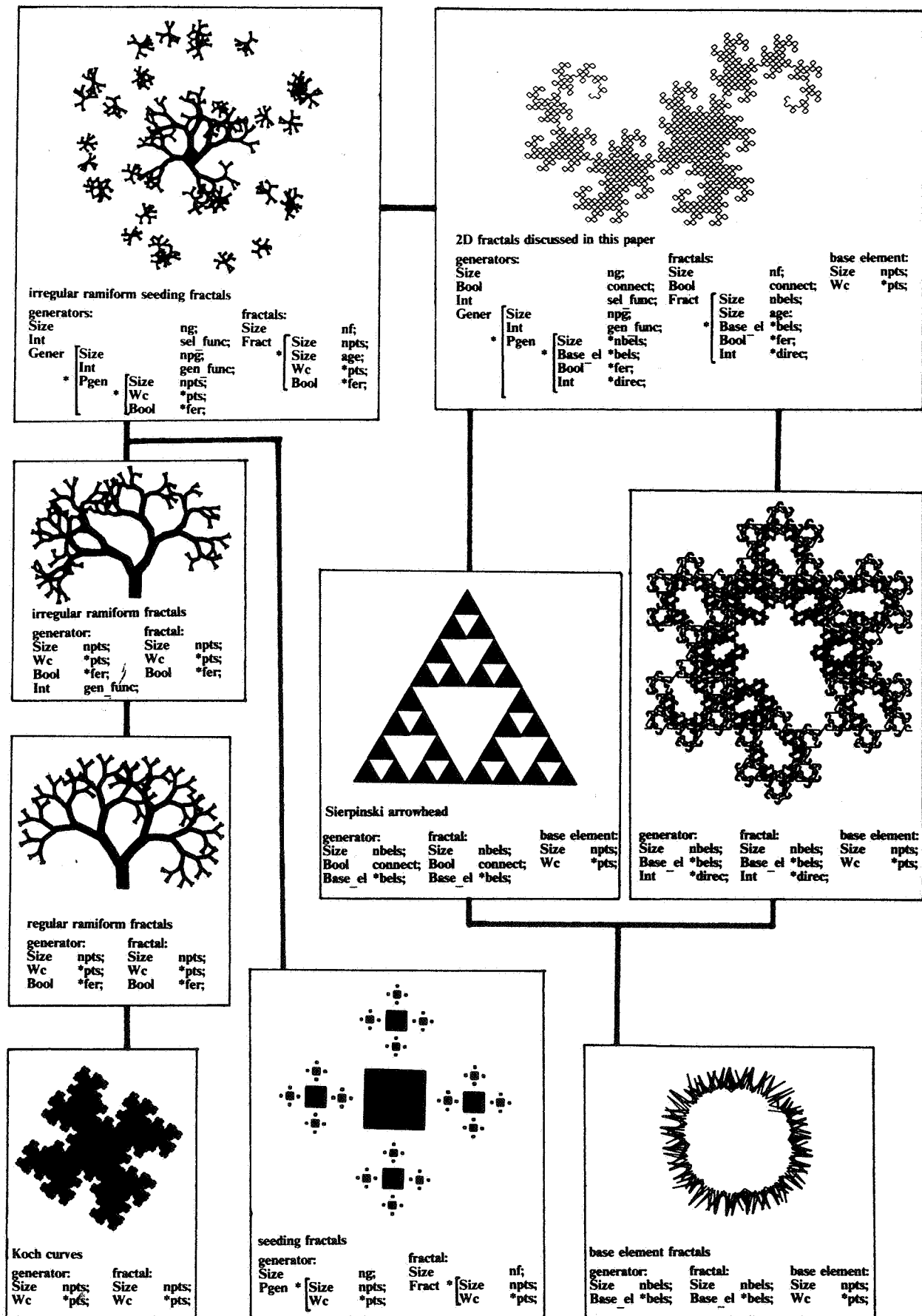


FIGURE 15. Classification diagram of linear fractals based on minimal data structures necessary for representing all components of the production rules. The data structure on top of the classification can be used for representing all fractals discussed in the section "Geometric construction of fractals".


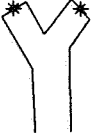

initiator	generator	base element
		

FIGURE 16. Production rule for a simple tree (Pythagoras tree). Fertile sides (sides of the preceding fractal approximant which will be replaced in next iteration steps, are marked with asterisks. The resulting fractal is shown in fig. 15.





initiator	generator	base element
	 	

FIGURE 17. Production rule for a tree in which the original generator is processed by a function which allows random movements of the generator between two limits. The generator processing function is described in the right part of the generator component. The resulting fractal is shown in fig. 15.





initiator	generator	base element
	 	

FIGURE 18. Production rule for a self-seeding square. The generator consists of two parts and is seeding new fractals during each iteration. The resulting fractal is shown in fig. 15.






initiator	generator	base element
	<div style="display: flex; justify-content: space-between;"> <div style="width: 30%;"> <p>if (age == 3) gen1 else gen0</p> </div> <div style="width: 40%; border-left: 1px dashed black; border-right: 1px dashed black; padding: 0 10px;"> <p>gen0</p>  <p>gen1</p>  </div> <div style="width: 25%; border-left: 1px dashed black; padding: 0 10px;">  </div> </div>	

FIGURE 19. Production rule for a vegetation of fractals. The growing generator (*gen0*) is chosen by the selection function (left part of the generator component) as the age of the fractal does not equate 3 iterations, on the third iteration the fractal starts seeding (*gen1*). The resulting fractal is shown in fig. 15.




initiator	generator	base element
		

FIGURE 20. Production rule for a simple base element fractal. The resulting fractal is shown in fig. 15.


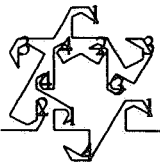

initiator	generator	base element
		

FIGURE 21. Production rule for the Monkeys tree shown in fig. 15. The direction of the base elements is denoted in the first two components with numbers: direction 1, normal base element; direction 2, y-coordinate has been reflected; direction 3, x-coordinate has been reflected; direction 4, x- and y-coordinate has been reflected. The resulting fractal is shown in fig. 15.

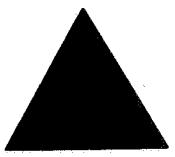


initiator	generator	base element
		

FIGURE 22. Production rule for constructing a Sierpinski arrowhead. The resulting fractal is shown in fig. 15.

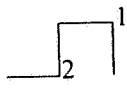
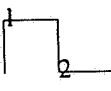
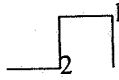

initiator	generator	base element
	<div style="display: flex; align-items: center;"> <div style="flex: 1;"> <p>if odd(i) gen0 else gen1</p> </div> <div style="flex: 2; border-left: 1px dashed black; padding-left: 10px;"> <p>gen0</p>  <p>gen1</p>  </div> </div>	

FIGURE 23. Production rule for constructing the Dragon Sweep, odd base elements of the preceding fractal approximant are replaced by gen0, even base elements by gen1. The resulting fractal is shown in fig. 15.

Another class of linear fractals are the fractals (more accurately: approximants of fractals), which are able to seed new fractals. The data structure of the generator has to be extended to an array of different parts of the generator. The data structure of the initiator and succeeding approximants, is an array of fractals. The production rule for a self seeding square is shown in fig. 18.

The possibilities of the ramiform fractal from fig. 18 may be enlarged by using an array of generators, consisting of an array of parts of the generator. A selection function is used to determine which generator should be used for replacing line-pieces of the preceding fractal approximant by a new set of line-pieces. In this example the growing generator (*gen0*) is chosen as long as the "age" of the fractal approximant does not equate 3 iterations. On the third iteration, the fractal starts seeding a new set of fractals (age equates 0), using the seeding generator (*gen1*). The production rule for a self reproducing ramiform fractal (see fig. 15), in which two generators are used is shown in fig. 19. The original ramiform fractal has changed into a vegetation of fractals.

In all fractals discussed so far fertile sides, consisting of one line piece, of the preceding approximant of the fractal were replaced by a new set of line pieces. It is possible to create a new class of fractals by the replacement of a polygon by a set of polygons. This polygon is called a "base

element". The production rule for a simple fractal, built from base elements, is shown in fig. 20. The data structures for representing the production rule for this class of fractals have to be enlarged by base elements as a new component.

The next step in creating new fractal objects is to add a new field in the data structure of the generator and fractal approximants, which defines the direction of the base elements in both components. In this paper for the linear fractals four different directions have been used: direction 1, normal base element; direction 2, y-coordinate of base element is reflected; direction 3, x-coordinate of base element is reflected; direction 4, x- and y-coordinate of base element are reflected. As a result it is possible to construct, for example, the Monkeys tree of Mandelbrot [22] (see diagram fig. 15, the production rule is given in fig. 21).

So far all base elements or line pieces of the fractals were connected. It is easy to define fractals with disconnected base elements by adding a boolean to the data structure of the generator and initiator, which indicates whether the base elements are connected or not. An example of a fractal using disconnected base elements is to construct a Sierpinski arrowhead, using triangles as base elements. It is shown in fig. 15 (production rule is given in fig. 22).

When all different rules to generate fractal objects are put together in three data structures, it is possible to cover all production rules given above. The resulting data structures are shown at the top of fig. 15. With the resulting data structures it is possible to describe all, interactively defined, production rules in the fractal system. With these data structures, production rules of still more complex fractal objects can be defined. For example the components of the production rule (fig. 23) of the Dragon sweep [22] may be represented by these data structures. The Dragon sweep (see fractal at the top of fig. 15) is created by using an array of two generators which are alternating. The odd base elements of the preceding fractal approximant are replaced by *gen0*, the even base elements by *gen1*. The corresponding fractal object generated with Iterated Function Systems is shown in fig. 13.

3.3.2. An algorithm for the generation of fractal objects, using geometrical constructions

The three components out of the production rule may be passed as arguments to an algorithm for calculating the fractal objects. The fractal object is generated in this algorithm in a feedback process, in which the calculated object (new fractals) is used as input (old fractals) of a next iteration. The algorithm, suitable for 2D fractals, is described in a summarized form using pseudo code below:

```

fractal( old_fractals, new_fractals, generator, base_element ) {
  A: if ((no_random_generator_is_used) && (no_selection_function_is_used))
    a_local_copy_of_the_generator_is_made_and_the_translation_vectors_are_derived_from_the_generator;
  B: next_fractal_from_old_fractals_is_taken {
    C: next_base_element_from_current_old_fractal_is_taken {
      D1: if (current_old_base_element_is_fertile) {
        D1a: first_point_is_taken_from_the_current_old_base_element_and_temporarily_stored,_this_value_is_the_initial_value_used_in_G;
        D1b: if ((random_function_is_used) || (selection_function_is_used))
          In_case_a_selection_function_is_used_a_generator_is_selected_from_the_array_of_generators,_in_case_a_random_processing_function_is_used_a_local_copy_(randomly_changed_by_a_function_)_is_made_of_the_original_generator,_translation_vectors_are_derived_from_the_local_copy;
        D1c: the_distance_between_the_last_and_the_first_point_of_the_current_old_base_element_is_calculated;
        D1d: the_orientation_of_the_line_through_last_and_first_point_of_the_current_old_base_element_is_calculated;
        E: next_part_is_taken_from_the_generator {
          F: next_base_element_is_taken_from_the_generator {
            G: next_point_is_taken_from_base_element_of_the_generator {
              G1: translation_is_performed_using_the_vectors_from_A / D1b_and_the_result_of_D1c;
              G2: Rotation_is_performed_using_results_of_D1d;
              G3:
                if (fractal_is_seeding) calculated_value_is_used_for_jumping_to_new_point;
                else new_point_is_added_to_new_fractals;
            }
            F1: if (fractal_is_not_seeding)
              fertilisation_status_of_new_base_element_(equal_to_fertilisation_status_of_current_base_element_generator)_is_added_to_new_fractals,_direction_of_the_new_base_element_is_evaluated_from_direction_current_base_element_generator_and_current_old_base_element_and_added_to_new_fractals.
          }
          E1: if (fractal_is_seeding)
            A_new_fractal_is_added_to_new_fractals,_the_value_from_D1a_is_used_as_initial_value;
        }
      }
    }
  }
  D2: if (current_old_base_element_is_not_fertile)
    old_base_element_and_its_fertilisation_and_direction_state_is_added_to_new_fractals;
}
}
}

```

3.4. Fractal objects generated using non-linear complex mappings

Many famous fractal objects belong to the class of fractals, in which the relation between input and output (fig. 8) is non-linear, using complex variables and parameters. Well-known fractals out of this class are the Mandelbrot and Julia sets (see: [22], [21], [27], [28]).

In order to visualize fractal objects, the easiest way is to decompose the complex mapping:

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

into a real and a imaginary part:

$$x_{n+1} = f(x_n, y_n)$$

$$y_{n+1} = g(y_n, x_n)$$

where $F = f + ig$ (see also formulas (8), (9) and (10)).

For points in the complex field it can be determined how many iterations it takes until a point escapes towards infinity or to other points of attraction. When for a certain mapping one or more stopping criteria are formulated, it is possible to count for each point how many iterations are necessary until the stopping criteria have become TRUE and the feedback process should be interrupted. The number of iterations before interrupting the loop may be used as an argument for a colour function.

There are 7 components which can be entered into an algorithm to visualize the fractal object. The first four components are the real part of the complex mapping ($f()$), the imaginary part ($g()$), the stopping criteria and the colour function. With a 5th argument it is specified which part of the complex field is considered. The 6th argument determines the size of the steps which are taken through the investigated part of the complex field. For real time results, it is possible to generate objects relatively quickly, by using large steps. For high quality pictures the step can be decreased until the pixel size of the graphical workstation is reached. The last argument determines the maximum number of iterations; in case different colours are used this argument will be in most cases equal to the number of available colours. In order to attain a sharp contrast, regions with points (a basin of attraction) which exceed the maximum number of iterations are often not coloured.

An algorithm for generating fractal objects out of this class is described using pseudo code below:

```

attractor( f( ), g( ), stopping_criteria(), colour_function(), window, step, max_iterations ) {
  next point ( x0, y0) is taken from window making steps with size step{
    n = 0;
    while( (stopping_criteria( xn, yn) != TRUE) && (n < max_iterations) ) {
      xn+1 = f(xn, yn);
      yn+1 = g(xn, yn);
      n++;
    }
    current point ( x0, y0) is coloured using colour_function( n );
  }
}

```

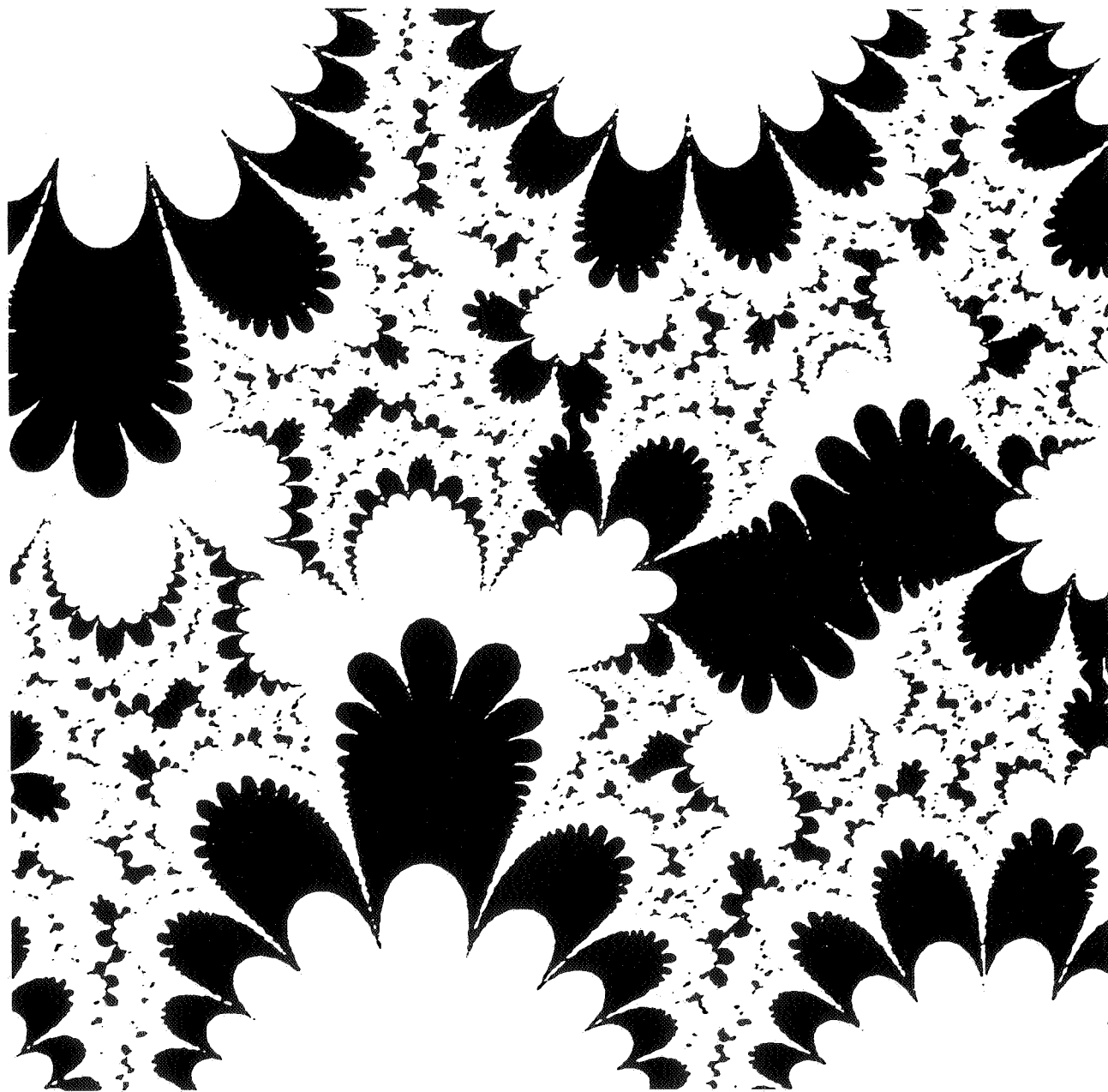



FIGURE 24. Basins of attraction of the complex mapping (14) and the stopping criterion (15). The investigated part of the complex field was: $(-\pi, -\pi, \pi, \pi)$.

In fig. 6b a Julia set is shown, using the complex mapping (10) ($a = 0, b = 1$). The arguments used for generating this object in the algorithm described above are:

- 1) $f(x_n, y_n) = x_n^2 - y_n^2 + a$
- 2) $g(x_n, y_n) = 2x_n y_n + b$
- 3) stopping_criteria: $x_n^2 + y_n^2 > \epsilon$ ($\epsilon = 100.0$)
- 4) colour_function: only basins of attraction in which n exceeds max_ iterations are coloured.
- 5) window: -1.5, -1.5, 1.5, 1.5
- 6) step: depending on resolution graphical workstation.
- 7) max_ iterations: 100

In fig. 24 the basins of attraction are shown of the complex mapping:

$$z_{n+1} = a \sin(z_n) + b \quad (14)$$

For the complex parameters a en b the values $a = b = 1 + i$ were chosen. The stopping criterion used was:

$$|y| < \epsilon, \quad (\epsilon = 0.01) \quad (15)$$

In the colour function only the points, in which the feedback loop was interrupted after an even number of iterations, were coloured. The investigated part of the complex field was: $(-\pi, -\pi, \pi, \pi)$.

4. APPLICATION OF FRACTALS

4.1. Estimation of fractal dimension

The fractal dimension (D) can be shown for several biological objects, for example the surface of the brain, lung, vascular system (all examples from Mandelbrot [22]), surfaces of proteins [19], coral reefs [4], vegetation [24] etc.

D can be determined analytically only in few special cases, such as some of the linear fractals using formula (6). In cases the self-similarity is disturbed by random processing functions (see irregular ramiform fractals in fig. 15), D can be determined because the set is statistically self-similar (see for statistically self-similar sets [10]). In many other cases D (for example from an object from nature) can only be demonstrated in an experimental way. D related with an object in a plane can for example be estimated with formula (7). (See for methods for estimating D of objects also: [29]).

In applications a fractal dimension is only useful when objects out of a same class are compared. Many different objects may have an equal D , but have a completely different appearance. D may be used for the plane-filling properties of a fractal. An example out of the class of regular ramiform fractals is shown in fig. 26. The production rule for those fractals is related to the rule shown in fig. 16.

The object at the top of fig. 26 has a D of almost 2.0, the fractal is almost plane filling. At the bottom D is nearly 1.0. The middle object is a transition case. D may be also used to demonstrate the degree of fragmentation. An example of this is shown in fig. 27. The production rule for the objects shown in fig. 27 are based on the production rule shown in fig. 18.


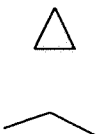

initiator	generator	base element
		

FIGURE 25. Production rule for constructing fragmenting fractals. The resulting fractals are shown in fig. 27.

The generator shown in fig. 18 may be considered as consisting of two parts: a "coastline" part and an "island" (or seed) part. In fig. 25 the production rule is shown for the objects in fig. 27. In this case a "coastline" is used, which is making random movements; together with an "island", which is also making random movements. The degree of fragmentation is controlled in the production rule of fig. 25 by a parameter, which determines the probability that an "island" is seeded. In fig. 27 the degree of fragmentation (D) is increasing from bottom to top.

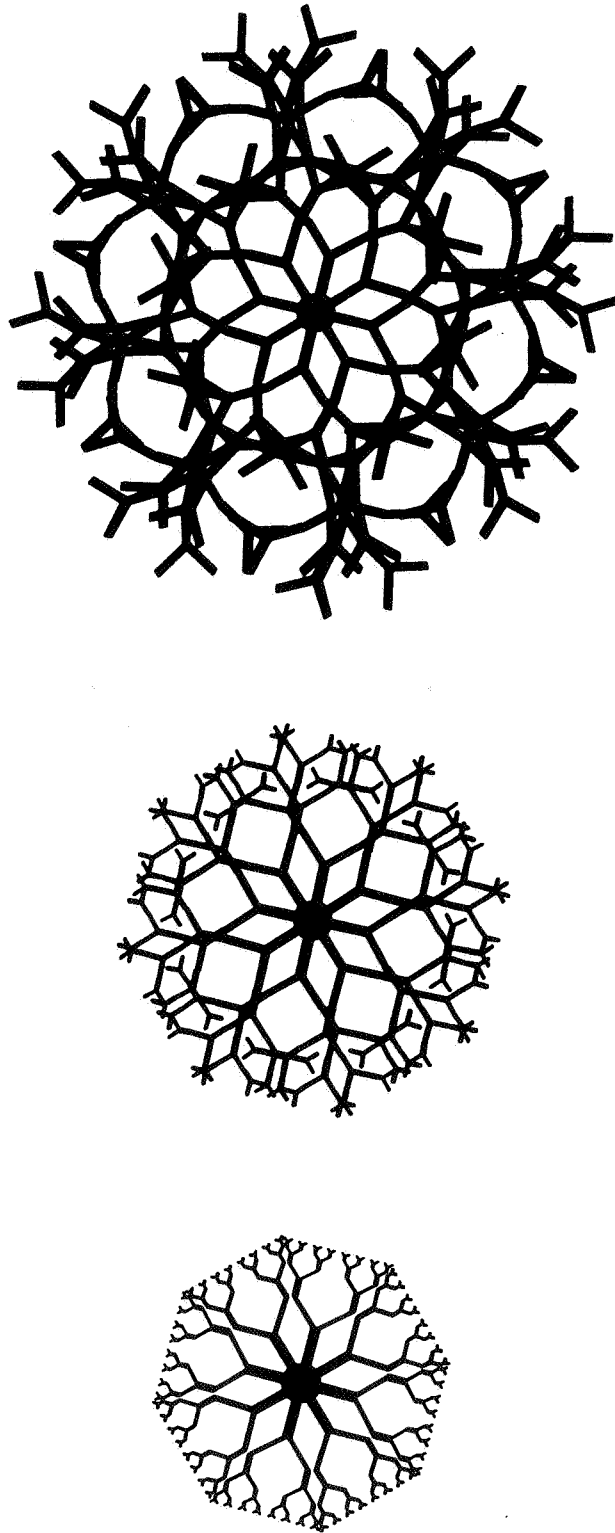


FIGURE 26. Series of ramifying fractals in which D is decreasing from top to bottom. At the top D is nearly 2.0, at the bottom D is almost 1.0.

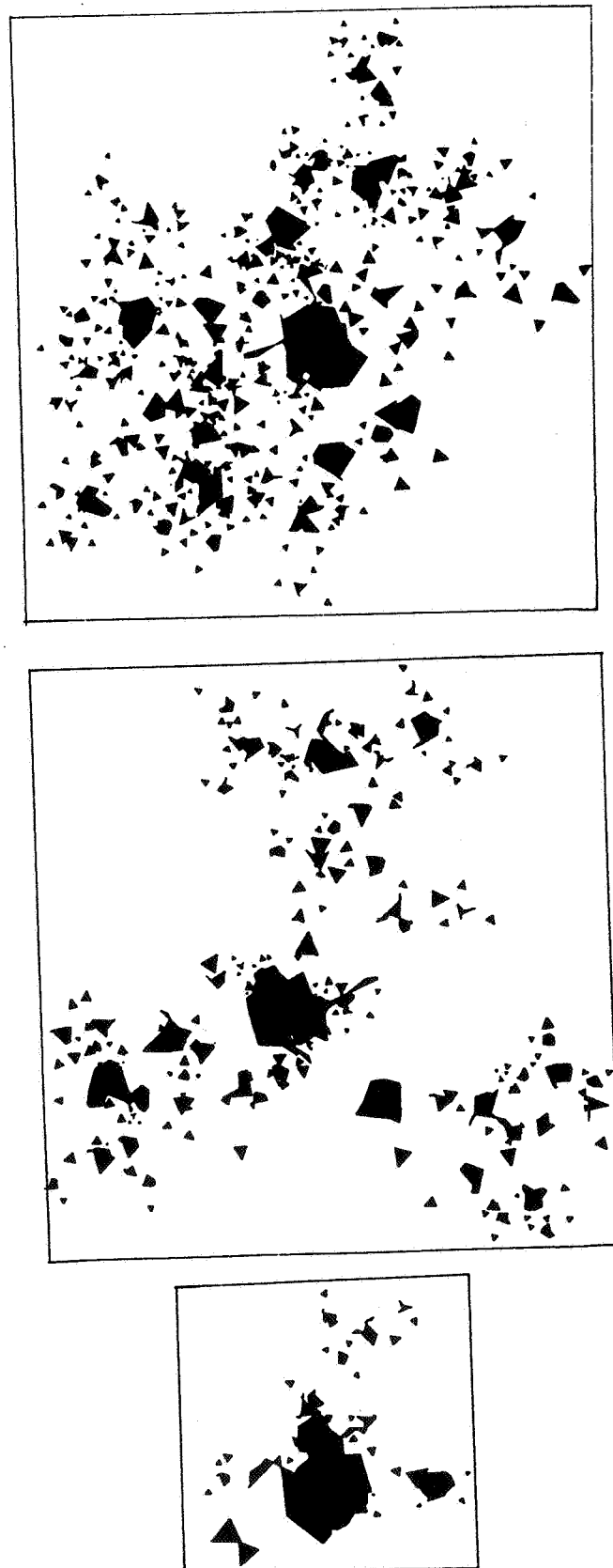


FIGURE 27. Series of fragmenting fractals in which D is decreasing from top to bottom. The production rule for those fractals is shown in fig. 27.

4.2. Fractal growth

Many processes in biology and in physics, for example growth processes, may be described as feedback processes; in which the last growth stage will serve as input for the next growth process. In a number of cases fractal objects are created during these processes. Fractal patterns occur during the growth of so-called viscous fingers, when water is pushed through a fluid of higher viscosity [25] ; when copper is electrodeposited, dendritic growth patterns occur [5] ; during chemical dissolution [6] . An overview of fractal growth phenomena in physics can be found in [32] .

Several models have been developed in order to simulate those growth patterns ([26] ,[30] ,[23]). One model, which seems to describe a wide range of growth processes in nature is the so-called diffusion limited aggregation model [30] . In this model particles or cells stick to certain parts of an object. New particles or cells will stick especially to the protruding points of the growing object, those points have the highest probability to grow. Schematically, an object growing in discrete stages, may have the form as shown in fig. 28. The growth velocity is minimal in the holes of the object and is increasing towards the tips, where it attains its maximum. An object, generated with geometrical construction, in which those growth rules are applied, is shown in fig. 29

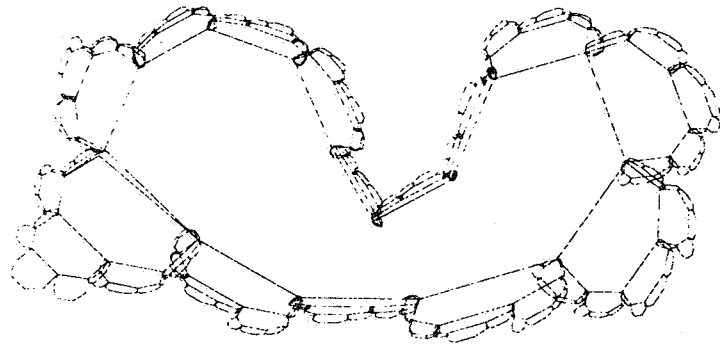


FIGURE 28. Scheme of an object, which is growing in discrete stages. The protruding points of the growing object have the highest probability to grow.

A growth model, which can be used to describe growth forms of sponges and corals (see for example: [11]), can use the same principles as above: growth velocities over the surface of the object, to form the next growth stage, are determined by the form of the preceding growth stage. A microscopic view of a longitudinal section through a tip of the branching sponge species *Haliclona oculata* (see Hartman [13]) might appear as shown in fig. 30.

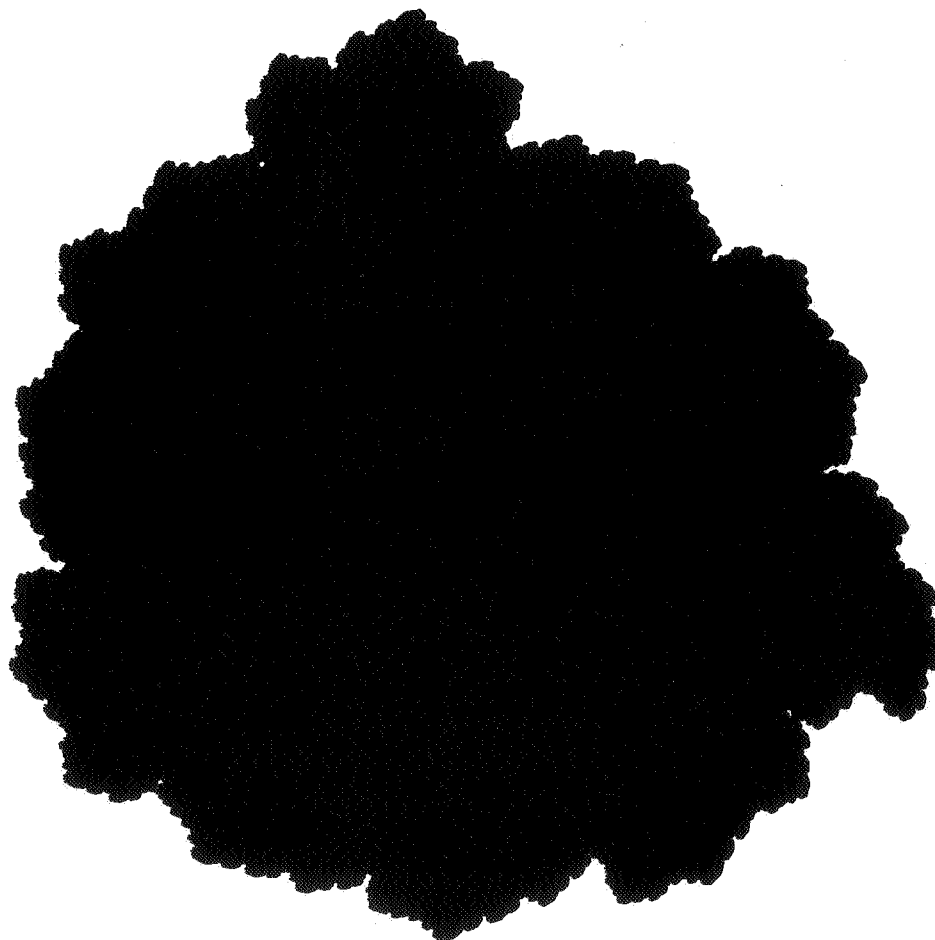


FIGURE 29. Fractal growth of an object, in which the growth rules described in fig. 28 are used.

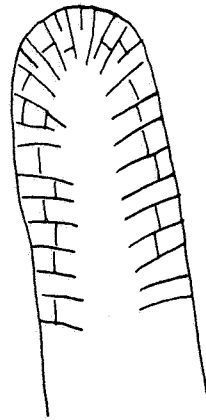


FIGURE 30. A microscopic view of a longitudinal section through a tip of the branching sponge species *Haliclona oculata*.

The stripes in fig. 30 represent the bundles of skeleton elements (the *spiculae*), which are connected with the proteine *spongine*. Some bundles of *spiculae* are arranged longitudinally, between the longitudinal bundles there are again transverse connections. Growth of the top of the sponge can be described schematically by a model, in which growth velocity attains its maximum at the top of the sponge and decreases to a certain minimum level laterally. A simple version of this growth model can be represented by the geometric construction of the generator, shown in fig. 31. In this construction bundles of *spiculae* are represented by lines.

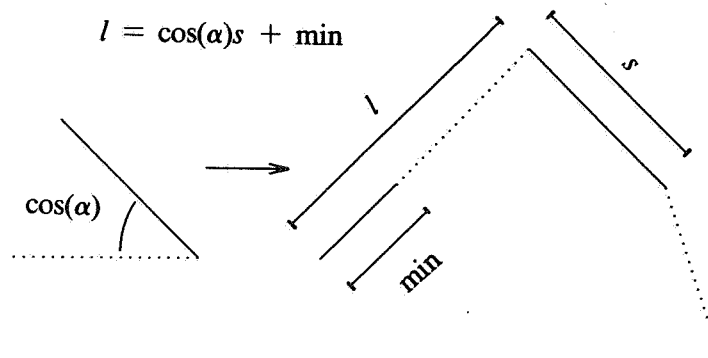


FIGURE 31. Geometric construction of a generator, in which the growth of the top of a sponge is described schematically.

The generator adds during each iteration a new longitudinal line, the length l of this line is determined by the cosine of the angle $\cos(\alpha)$ between the original transverse line and the x-axis. During each iteration also a new transverse line is added to the object, when there is enough space between two neighbouring transverse lines, an extra new transverse line is added. The initial form consists of 16 transverse lines, arranged in a hemicycle. Some growth stages of the sponge top are shown in fig. 32.

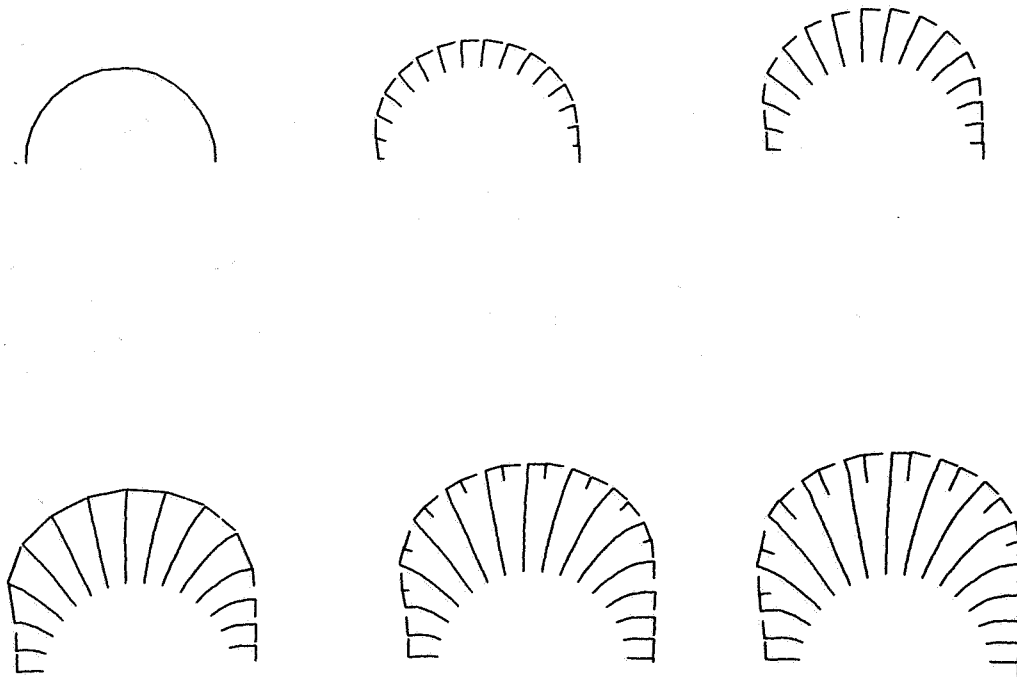


FIGURE 32. Some growth stages of a sponge top, using the production rule shown in fig 31.

The sponge model is, of course, a strong simplification of reality. In reality this sponge species shows all kind of branching patterns, which range from irregular forms to almost perfect dichotomous forms. The objects shown in fig. 32 are, although generated in a feedback process, not of a fractal dimension. When branching would also be included in the model, then also in this model fractal objects would be generated too.

5. CONCLUSION

In this tutorial we introduced some main aspects about the subject of fractals. The mathematics of fractals and all its applications in computer graphics and many other areas of science is a rapidly developing topic. Literature about fractals is quite scattered over the different sciences. For further reading about fractals we can recommend the following books: "The fractal geometry of nature" [22] ; "The beauty of fractals" [27] , especially for those interested in beautiful pictures; "On growth and form: fractal and non-fractal patterns in physics" [32] , for those interested in fractal growth; "The geometry of fractal sets" [10] , which is more about the mathematical background of fractals. The paper entitled "On the structure of self-similar sets" [14] is a good mathematical introduction to the aspects of self-similarity (see also: [7]).

ACKNOWLEDGEMENTS

Dr. N.M. Temme (Centre for mathematics and Computer Science (Amsterdam), department of Applied Mathematics) is thanked for reviewing this paper

References

1. M. Aono and L. Kunii, "Botanical tree image generation," *IEEE Computer Graphics and Applications* **8**, pp. 10-34 (1984).
2. M.F. Barnsley and S.G. Demko, *Chaotic dynamics and fractals*, Academic Press, New York (1986).
3. Blanchard, "Complex analytic dynamics on the Riemann sphere," *Bull. Am. Math. Soc.* **11**, pp. 85-141 (1984).
4. R. H. Bradbury and R. E. Reichelt, "Fractal dimension of a coral reef at ecological scales," *Mar. Ecol. Prog. Ser.* **10(2)**, pp. 169-172 (1983).
5. R.M. Brady and R.C. Ball, "Fractal growth of copper electrodeposits," *Nature* **309**, pp. 225-229 (1984).
6. G. Daccord and R. Lenormand, "Fractal patterns from chemical dissolution," *Nature* **325**, pp. 41-43 (1987).
7. F.M. Dekking, "Recurrent sets," *Advances in Mathematics* **44**, pp. 78-104 (1982).
8. S. Demko, L. Hodges, and B. Naylor, "Construction of fractal objects with iterated function systems," *Computer Graphics* **19(3)**, pp. 271-278 (1985). SIGGRAPH '85 proceedings.
9. Devaney, *An introduction to chaotic dynamical systems*, Benjamin (1986).
10. K.J. Falconer, *The geometry of fractal sets*, Cambridge University Press, Cambridge (1985).
11. R.R. Graus and I.G. Macintyre, "Variation in growth forms of the reef coral *Montastrea annularis* (Ellis and Solander): a quantitative evaluation of growth response to light distribution using computer simulation," *Smithsonian contributions to the marine sciences* **12**, pp. 441-464 (1982).
12. P.J.W. ten Hagen and M.M. de Ruiter, *C-GKS the C implementation of GKS, the graphical Kernel System user guide*, Centre for Mathematics and Computer science, Amsterdam (1986).

13. W.D. Hartman, *Natural history of the marine sponges of southern New England*, Pea poddy museum of natural history, Yale University, New Haven, Connecticut (1958).
14. M. Hata, "On the structure of self-similar sets," *Japan J. Appl. Math.* **2**, pp. 381-414 (1985).
15. P. Hogeweg and B. Hesper, "A model study on biomorphological description," *Pattern Recognition* **6**, pp. 165-179 (1974).
16. F.R.A. Hopgood, D.A. Duce, J.R. Gallop, and D.C. Sutcliffe, *Introduction to the Graphical Kernel system (GKS)*, Academic Press, London (1983).
17. Julia, "Memoire sur l'iteration des fonctions rationnelles," *J-Math.*, pp. 47-245 (1918).
18. J.A. Kaandorp, *Interactive generation of fractal objects*, Eurographics '87 proceedings. 1987.
19. M. Lewis and D.C. Rees, "Fractal surfaces of proteins," *Science* **230**, pp. 1163-1165 (1985).
20. A. Lindenmayer, "Mathematical models for cellular interactions in development," *J. Theoret. Biol.* **18**, pp. 280-299 (1968).
21. B.B. Mandelbrot, "Fractal aspects of the iteration $z \rightarrow lz(1 - z)$ for complex l and z ," *Annals of the New York Academy of Sciences* **357**, pp. 249-259, Non Linear Dynamics, Ed. R.H.G. Helleman (1980).
22. B.B. Mandelbrot, *The fractal geometry of nature*, Freeman, San Francisco (1983).
23. P. Meakin, "A new model for biological pattern formation," *J. Theoret. Biol.* **118**, pp. 101-113 (1986).
24. D.R. Morse, J.H. Lawton, M.M. Dodson, and M.H. Williamson, "Fractal dimension of vegetation and the distribution of arthropod body length," *Nature* **314**, pp. 731-733 (1985).
25. J. Nittmann, G. Daccord, and G. H.E. Stanley, "Fractal growth of viscous fingers: quantitative characterization of a fluid instability phenomenon," *Nature* **314**, pp. 141-144 (1985).
26. J. Nittmann and H.E. Stanley, "Tip splitting without interfacial tension and dendritic growth patterns arising from molecular anisotropy," *Nature* **321**, pp. 663-668 (1986).
27. H. O. Peitgen and P. H. Richter, *The Beauty of Fractals*, Springer Verlag (1986).
28. C.A. Pickover, "Biomorphs: computer displays of biological forms generated from mathematical feedback loops," *Computer Graphics Forum* **5(4)**, pp. 313-316 (1986).
29. C.A. Pickover, "A Monte Carlo approach for ϵ placement in fractal-dimension calculations for waveform graphs," *Computer Graphics Forum* **5**, pp. 203-210 (1986).
30. L.M. Sander, "Fractal growth processes," *Nature* **322**, pp. 789-793 (1986).
31. A.R. Smith, "Plants, fractals and formal languages," *Computer Graphics* **18(3)**, pp. 1-10, SIGGRAPH '84 proceedings (1984).
32. H.E. Stanley and N. Ostrowsky, *On growth and form: fractal and non-fractal patterns in physics*, Martinus Nijhoff Publishers, 1987.

