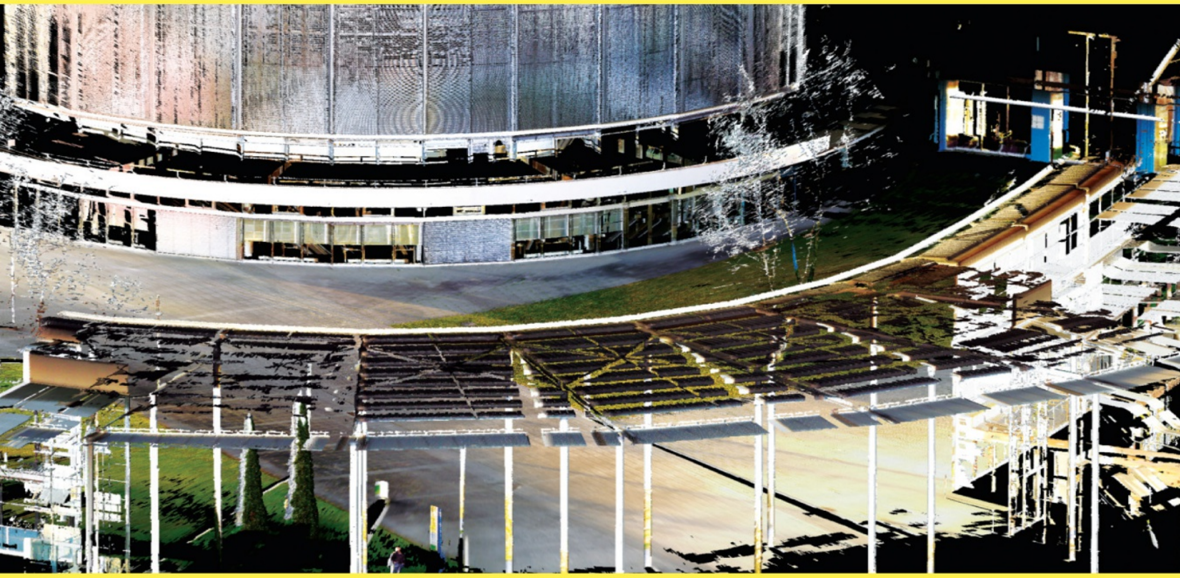
NUMERICAL METHODS IN ENGINEERING SERIES

GEOMETRIC MODELING AND APPLICATIONS SET



Volume 5

# Geometric Modeling of Fractal Forms for CAD

Christian Gentil
Gilles Gouaty and Dmitry Sokolov

iSTE

WILEY

# Geometric Modeling of Fractal Forms for CAD

**Geometric Modeling and Applications Set**

coordinated by
Marc Daniel

Volume 5

# Geometric Modeling of Fractal Forms for CAD

Christian Gentil
Gilles Gouaty
Dmitry Sokolov

iSTE

WILEY

First published 2021 in Great Britain and the United States by ISTE Ltd and John Wiley & Sons, Inc.

ISTE Ltd
27-37 St George's Road
London SW19 4EU
UK

www.iste.co.uk

John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
USA

www.wiley.com

Library of Congress Control Number: 2021932086

British Library Cataloguing-in-Publication Data
A CIP record for this book is available from the British Library
ISBN 978-1-78630-040-9

# Contents

# Preface

This work introduces a model of geometric representation for describing and manipulating complex non-standard shapes such as rough surfaces or porous volumes. It is aimed at students in scientific education (mathematicians, computer scientists, physicists, etc.), engineers, researchers or anyone familiar with the mathematical concepts addressed at early stages of the graduate level. However, many parts are accessible to all, in particular, all introductory sections that present ideas with examples. People with no prior background, whether they are artists, designers or simply curious, will be able to understand the philosophy of our approach, and discover a new universe of unsuspected and exciting forms.

Geometric representation models are mathematical tools integrated into computer-aided geometric design (CAGD) software. They make the production of numerical representations of forms possible. By means of graphical interfaces or programming tools, users can draw and/or manipulate these shapes. They can also test or evaluate their physical properties (mechanical, electro-magnetic, acoustic, etc.) by communicating geometric descriptions to further specific numerical simulation software.

The geometric representation model we present here is based on the fractal geometry paradigm. The principle behind this consists of studying the properties (signal, geometry, phenomena, etc.) at different scales and identifying the invariants from there. The objects are described as self-referential between two scales: each of the object features (namely, the lower scale level) is described as a reference to the object itself (namely, the higher scale). This approach is not conventional and often confusing at first. We come to perceive its richness and power very quickly, however. The universe of forms that can possibly be created is infinite and has still only partially been explored.

In this book, we present the mathematical foundations, so that the reader can access all the information to understand, test and make use of this model. Properties, theorems and construction methods are supplemented with algorithms and numerous examples and illustrations. Concerning the formalization, we have chosen to use precise and rigorous mathematical notations to remove any ambiguity and make understanding easier.

Readers unwilling to be concerned with mathematical formalisms can get to grips with the philosophy of our approach by focusing on the sections found at the beginnings of the chapters, in which ideas and principles are intuitively presented, based on examples.

This book is the result of 25 years of research carried out mainly in the LIRIS laboratories of the University of Lyon I and LIB of the University of Burgundy Franche-Comté. This research was initiated by Eric Tosan, who was instrumental at the origin of this formalism and to whom we dedicate this work.

Christian GENTIL
Gilles GOUATY
Dmitry SOKOLOV

February 2021

# Introduction

## I.1. Fractals for industry: what for?

This book shows our first steps toward the fundamental and applied aspects of geometric modeling. This area of research addresses the acquisition, analysis and optimization of the numerical representation of 3D objects.

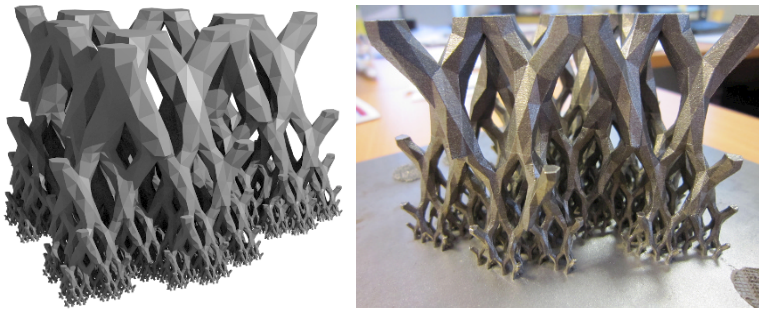

a) 3D model b) 3D printing

**Figure I.1.** *3D tree built by iterative modeling (source: project MODITERE no. ANR-09-COSI-014)*

Figure I.1 shows an example of a structure that admits high vertical loads, while minimizing the transfer of heat between the top and bottom of the part. Additive manufacturing (3D printing) allows, for the first time, the creation of such complex objects, even in metal (here with a high-end laser printer EOS M270). This type of technology will have a high societal and economic impact, enabling better systems to be created (engines, cars, airplanes, etc.), designed and adapted numerically for optimal functionality, thus consuming less raw material, for their manufacturing, and energy, when used.

Current computer-aided design is, however, not well suited to the generation of such types of objects. For centuries, for millennia, humanity has produced goods with axes, files (or other sharp or planing tools), by removing bits from a piece of wood or plastic. Tools subsequently evolved into complex numerical milling machines. However, at no point during these manufacturing processes did we need sudden stops or permanent changes in the direction of the cutting tool. The patterns were always "regular", hence the development of mathematics specific to these problems and our excellent knowledge of the modeling of smooth objects. This is why it was necessary to wait until the 20th century to have the mathematical knowledge needed to model rough surfaces or porous structures: we were just not able to produce them earlier.

Thus, since the development of computers in the 1950s, computer-aided geometric design (CAGD or CAD) software has been developed to represent geometric shapes intended to be manufactured by standard manufacturing processes. These processes are as follows:

– subtractive manufacturing, using machine tools such as lathes or milling machines;

– molding, where molds themselves are made using machine tools;

– deformation-based manufacturing: stamping or swaging (but again, dies are usually manufactured using machine tools), folding, etc.;

– cutting, etc.

Each of these processes imposes constraints, for example, concerning collision issues in milling machines (even a five-axis mill cannot produce any geometry). These manufacturing processes inevitably influenced the way we design the geometries of objects, in order to actually manufacture them. For example, CAD software has integrated these design methodologies by developing appropriate numerical models or tools. Currently, most CAD software programs are based on the representation of shapes by means of surfaces defining their edges. These surfaces are usually described using a parametric representation called non-uniform rational B-spline (NURBS). These surface models are very powerful and very practical. It is possible to represent any volume enclosed by a quadric (cylinders, cones, spheres, etc.) and complex shapes, such as car bodies or airplane wings. They were originally designed for this.

However, the emergence of additive manufacturing techniques has caused an upheaval in this area, opening up the possibility of potentially "manufacturable" forms. By removing the footprint constraint of the tool, it then becomes possible to produce very complex shapes with gaps or porosity. These new techniques have called into question the way objects are designed. New types of objects, such as porous objects or rough surfaces, can have many advantages, due to their specific physical properties. Fractal structures are used in numerous fields such as architecture (Rian and Sassone 2014), jewelry (Soo *et al.* 2006), heat and mass transport (Pence 2010), antennas (Puente *et al.* 1996; Cohen 1997) and acoustic absorption (Sapoval *et al.* 1997).

## I.2. Fractals for industry: how?

The emergence of techniques such as 3D printers allows for new possibilities that are not yet used or are even unexplored. Different mathematical models and algorithms have been developed to generate fractals. We can categorize them into three families, as follows:

– the first groups algorithms for calculating the attraction basins of a given function. Julia and Mandelbrot (Peitgen and Richter 1986) or the Mandelbulb (Aron 2009) sets are just a few examples;

– the second is based on the simulation of phenomena such as percolation or diffusion (Falconer 1990);

– the last corresponds to deterministic or probabilistic algorithms or models based on the self-similarity property associated with fractals such as the terrain generator (Zhou *et al.* 2007), the iterated function system (Barnsley *et al.* 2008) or the L-system (Prusinkiewicz and Lindenmayer 1990).

In the latter family of methods, shapes are generated from rewriting rules, making it possible to control the geometry. Nevertheless, most of these models have been developed for image synthesis, with no concerns for “manufacturability”, or have been developed for very specific applications, such as wood modeling (Terraz *et al.* 2009). Some studies approach this aspect for applications specific to 3D printers (Soo *et al.* 2006). In (Barnsley and Vince 2013b), Barnsley defines fractal homeomorphisms of $[0,1]^2$ onto the modeling space $[0,1]^2$. The same approach is used in 3D to build 3D fractals. A standard 3D object is integrated into $[0,1]^3$ and then transformed into a 3D fractal object. This approach preserves the topology of the original object, which is an important point for “manufacturability”.

The main difficulty associated with traditional methods for generating fractals lies in controlling the forms. For example, it is difficult to obtain the desired shape using the fractal homeomorphism system proposed by Barnsley. Here, we develop a modeling system of a new type based on the principles of existing CAD software, while expanding their capabilities and areas of application. This new modeling system offers designers (engineers in industry) and creators (visual artists, designers, architects, etc.) new opportunities to quickly design and produce a model, prototype or unique object. Our approach consists of expanding the possibilities of a standard CAD system by including fractal shapes, while preserving ease of use for end users.

We propose a formalism based on standard iterated function systems (IFS) enhanced by the concept of boundary representation (B-rep). This makes it possible to separate the topology of the final forms from the geometric texture, which greatly simplifies the design process. This approach is powerful, and it generalizes subdivision curves and standard surfaces (linear, stationary), allowing for additional control. For example, we have been able to propose a method for connecting a primal subdivision scheme surface with a dual subdivision scheme surface (Podkorytov *et al*. 2014), which is a difficult subject for the standard subdivision approach.

The first chapter recalls the notion of self-similarity, intimately linked to that of fractality. We present the IFS, formalizing this property of self-similarity. We then introduce enhancements into this model: controlled iterated function systems (C-IFS) and boundary controlled iterated function systems (BC-IFS). The second chapter is devoted to examples. It provides an overview of the possibilities of description and modeling of BC-IFS, but also allows better understanding the principle of the model through examples. The third chapter presents the link between BC-IFS, the NURBS surface model and subdivision surfaces. The results presented in this chapter are important because they show that these surface models are specific cases of BC-IFS. This allows them to be manipulated with the same formalism and to make them interact by building, for example, junctions between two surfaces of any kind. In the fourth chapter, we outline design tools that facilitate the description process, as well as examples of the applications, of the design of porous volumes and rough surfaces.

# 1
# The BC-IFS Model

In this chapter, in section 1.1, we begin by intuitively introducing the notion of self-similarity. Then, in section 1.1.2, we give its mathematical formulation as proposed by Hutchinson (Hutchinson 1981) using iterated function systems (IFS). Next, we present how this mathematical model can be implemented to calculate and visualize geometric shapes.

In section 1.2, we set out an extension of the IFS, allowing us to move away from strict self-similarity and generate a larger family of forms. This extension is called a *controlled iterated function system* (C-IFS).

Finally, the final step in formalization is to enhance the C-IFS model with the notion of boundary representation (B-rep). This step is fundamental because it will make it possible to describe and control the topology of fractal forms.

## 1.1. Self-similarity and IFS

In this book, when we talk about fractal shapes, fractal objects or simply fractals, it is in the sense of self-similar objects.

There are different definitions of self-similarity. For example, in the field of image processing, an image is considered self-similar when certain parts of that image are identical to (or “look alike”) other parts. This property is exploited, in particular, by *inpainting*, a technique that consists of reconstructing part of a damaged or deliberately subtracted image (for example, to erase a character). The property of self-similarity is often verified with natural images, which is why *inpainting* algorithms are so successful. The principle is to look for an area similar to the area that needs to be filled in, and copy it over the missing area.

In fractal geometry, it has a different meaning.

*Geometric Modeling of Fractal Forms for CAD,*
First Edition. Christian Gentil, Gilles Gouaty and Dmitry Sokolov.
© ISTE Ltd 2021. Published by ISTE Ltd and John Wiley & Sons, Inc.

DEFINITION.– *An object is called self-similar if it is composed of copies of itself.*

This definition is not rigorous and further clarification has to be provided. Nonetheless, it contains the main idea. Rather than describing the complex structure of an object, we describe its parts using a reference to the object itself. It is therefore a self-referential or recursive definition.

For example, we can describe the structure of a tree as being composed of several parts: its main branches (see Figure 1.1). Each of its parts can be considered as a smaller tree.

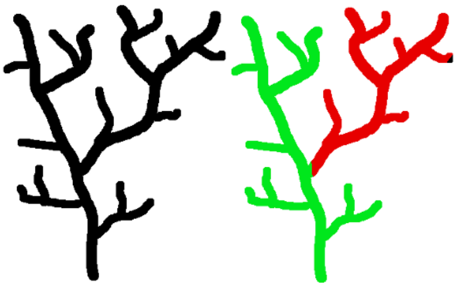


**Figure 1.1.** *Schematic illustration of self-similarity. The black tree can be seen as a composition of two trees (in green and red). For a color version of this figure, see www.iste.co.uk/gentil/geometric.zip*

In this example, we understand how a complex shape, such as that of a tree, can be simply described. Self-similarity provides information about the structure of the object with a different approach from that of standard geometric representations. By referring to the parts of the object, we study the details of the shape, in other words we perform a change of scale. Each detail is then described according to the object itself, namely as if the detail "looked like" a reduced version of the object. The tree is made up of branches. Each branch is defined as a tree. We can then apply thereto the definition of the tree: it is a composition of branches. This reasoning can then be indefinitely iterated.

From this definition and this introductory example, a number of questions immediately arise:

– Is this description relevant?

– Which objects are self-similar?

– What does it mean when a detail "looks like" the object?

– Is the shape of an object completely determined from its self-similarity?

– Knowing the self-similarity property, is it possible to reproduce its shape?

– Can an object possess different self-similarities?

In order to better understand the self-similarity property and provide an early answer to some of these questions, we are going to consider a second example. However this time, we start from an object whose shape is not *a priori* known, but its self-similarity property is. Assume that this object consists of five main parts, each of which is an exact copy of the object on a smaller scale.

The left part of Figure 1.2 shows each part, symbolized by an arbitrarily chosen shape, a square in this case. In this illustration, each part is not identical to the overall structure. To meet the definition of self-similarity, we need to replace each part with the overall composite structure of the five forms. We thus obtain the image in the center of Figure 1.2. However, by adding details to the details, we have also added details to the overall structure that were not initially present. There is always a discrepancy between the details of the overall form and the details of each part. By iterating this construction to reduce this discrepancy, the same effect will then be observed. Consistency between the object and each of its parts can only be achieved if the process is applied an infinite number of times to obtain a result like the image on the right

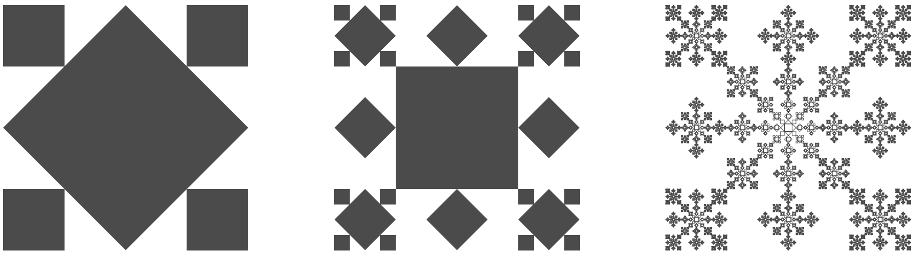


**Figure 1.2.** *An example of a self-similar object composed of five copies of itself. These five main parts are symbolized on the left by the five squares. In the center, each square has been replaced by the overall form composed of the five parts. On the right-hand side, the same construction process has been applied seven times*

From the second example, the following observations can be derived:

– It shows how from the definition of self-similarity alone, a form can be built.

– It brings forward the recursive aspect of the definition.

– To obtain the final form, the construction process must be applied an infinite number of times. From a mathematical point of view, this is not a problem, but it is unfeasible from a practical point of view and in particular, in computer science.

– The shape originally chosen to represent each self-similar part (in this example, the square) is not relevant. Indeed, the details that we add are successively smaller. After a few iterations (seven or eight), they have little influence on the overall shape. When the details are smaller in size than the accuracy of the display medium or the accuracy of the manufacturing device, their initial shape has no influence on the result. Figure 1.3 shows the same self-similar structure, but is symbolized by the image of an elephant instead of a square. The replacement or substitution process remains the same. We can observe that we obtain an identical figure as soon as the details become very small. The final image is less dense than that obtained with the square, but the structure is the same.

– The definition of self-similarity that we propose is very inaccurate. What does "copy of the object" mean? For our example, the copies correspond to the object after applying geometric transformations: change of scale ($<1$), translation and rotation. These transformations are essential because they are the ones that tell us, with each iteration, how to replace each part with its details.

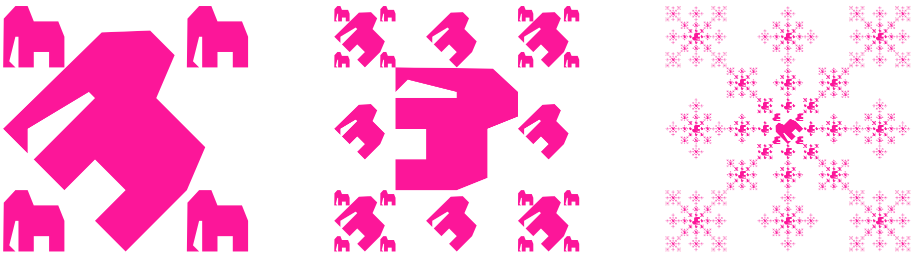


**Figure 1.3.** *The self-similarity property, as shown in Figure 1.2, is symbolized by means of a pink elephant instead of a square. For a color version of this figure, see www.iste.co.uk/gentil/geometric.zip*

COMMENTS ON FIGURE 1.3.– *The same construction process is used. In the right-hand side of the image, we observe that after seven iterations the resulting form is composed of very fine details: a set of very small elephants that we cannot discern. The overall shape, however, is identical to that shown in Figure 1.2.*

### 1.1.1. *Mathematical definitions and reminders*

The concept of self-similarity was formalized by Hutchinson, based on iterated function systems (IFS) (Hutchinson 1981). Subsequently, it was developed and popularized by Barnsley (1988). Because the construction of fractal objects is based on an iterative process, it is then appropriate to choose workspaces with “good properties” that ensure the convergence of these constructions. In this section, we give the definitions of the spaces from which a very general notion of objects is defined and which iterated systems of functions will be able to operate with.

We also recall the concept of contracting transformations and fixed-points. These two concepts justify the representation (or coding) of self-similar fractals by iterated function systems. They make it possible to show the existence and unicity of a fractal associated with an iterated system of functions.

This section is not essential in understanding the rest of this book. An intuitive understanding of the concept of contractive function and of the principle of the fixed-point theorem should be enough.

##### 1.1.1.1. *Working space*

As a work environment, we choose a complete metric space, denoted by $(\mathbb{X}, d)$. The topology associated with the $\mathbb{X}$ space is simply that induced by the metric $d$. In practice, this space corresponds to the set in which fractal objects are designed. Generally, $\mathbb{X} = \mathbb{R}^2$, $\mathbb{R}^3$ and $d$ correspond to the Euclidian distance, hereafter denoted by $d_E$. Subsequently, we shall see that $\mathbb{X}$ can designate a barycentric space. The latter are best suited to represent shapes according to control points.

A complete space is a space in which any Cauchy sequence converges. Intuitively, if the distance between the terms of a sequence can be as small as we wish, provided that the terms are chosen with sufficiently high indices, then we are guaranteed that this sequence possesses a limit in that same space.

In our space $(\mathbb{X}, d)$, an “object” is defined as a non-empty compact subset. The set of the objects is denoted by $\mathcal{H}(\mathbb{X})$. If space $\mathbb{X}$ is a finite-dimensional metric Space (which will be the case in our context), then a compact set is a closed and bounded set. A closed set is a set that contains its boundary. The fact that it is bounded means that it can be included inside a ball of radius $r < \infty$.

The definition that we give an object is very general and does not ensure that the object can be materialized or manufactured. It can have an empty interior and it can be composed of unconnected parts, or both. For example:

– if $(\mathbb{X}, d) = (\mathbb{R}^2, d_E)$, the set reduced to a point $\{(1, 2)\}$ is an object with an empty interior;

– if $(\mathbb{X}, d) = (\mathbb{R}, d_E)$, $\{x_n = \frac{1}{n}, n \in \mathbb{N}\} \cup \{0\}$ is an non-connected object with an empty interior.

It is helpful to associate a distance with the spaces that we work in. This allows us to make use of the topology induced by this distance and to simply manipulate the notion of neighborhood. We equip the set of objects with the Hausdorff distance, induced by that of the working space $(\mathbb{X}, d)$. This distance is defined as follows.

DEFINITION.– *Let $(\mathbb{X}, d)$ be a metric space, the Hausdorff distance between two elements $A$ and $B$ of $\mathcal{H}(\mathbb{X})$ is defined by:*

$$d_{\mathbb{X}}(A, B) = \max\{\sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b)\}$$

The Hausdorff distance is a distance between sets. To intuitively understand this distance, consider two sets $A$ and $B$. If the Hausdorff distance between these two sets is equal to $\epsilon$, it means that for any point of $A$, a point $B$ can be found at a distance less than or equal to $\epsilon$. However, for any point of $B$, a point of $A$ can also be found at a distance less than or equal to $\epsilon$. In other words, if we take a ball of radius $\epsilon$ and we walk the center of that ball over all the points of $A$, the space covered by this ball will cover the set $B$. The same property must be verified when inverting the role of $A$ and $B$ (see Figure 1.4).

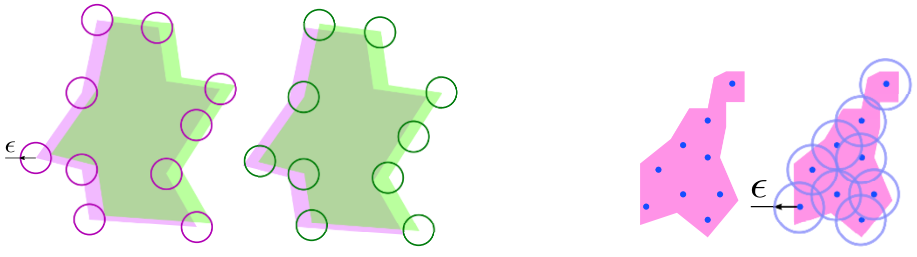


**Figure 1.4.** *Hausdorff distance. For a color version of this figure, see www.iste.co.uk/gentil/geometric.zip*

COMMENTS ON FIGURE 1.4.– *On the left, the Hausdorff distance between the two purple and green sets is less than or equal to $\epsilon$. If the center of a disc of radius $\epsilon$ travels through all the points of the purple set, the set of discs will cover the green set. Conversely, if the center of a disc of radius $\epsilon$ travels through all the points of the green set, the set of discs will cover the purple set. On the right, the pink set is at a Hausdorff distance smaller than $\epsilon$ from the set consisting of blue dots. Indeed, each blue dot belongs to the pink set and the set of discs of radius $\epsilon$ centered at the blue dots covers the pink set. We observe that these last two sets do not share the same topological nature.*

We then define the set of objects in which we shall build our fractal objects using iterative processes.

DEFINITION.– *The space of the objects of $(\mathbb{X}, d)$, denoted $\mathcal{H}(\mathbb{X})$, is the set of the non-empty compact subsets of $\mathbb{X}$ endowed with the Hausdorff distance $d_{\mathbb{X}}$, associated with $d$.*

The following property is fundamental, it makes it possible to show that the mere knowledge of the self-similar property alone is sufficient to determine an object in a unique way.

PROPERTY.– If $(\mathbb{X}, d)$ is a complete metric space, then $(\mathcal{H}(\mathbb{X}), d_{\mathbb{X}})$ is a complete metric space.

This property also ensures the convergence of the iterative construction process of fractal objects (see section 1.1.4).

#### 1.1.1.2. *Geometric transformations*

Geometric transformations play a major role in the formalization of the self-similarity property. As we have been able to understand through the introductory example (see Figure 1.2), they completely determine the geometry of the forms. Here, we recall some definitions and properties of transformations.

DEFINITION.– *A transformation is a function whose definition domain is equal to the starting space.*

DEFINITION.– *A transformation $T : \mathbb{X} \to \mathbb{X}$, with $(\mathbb{X}, d)$ as a metric space, is contractive if there is a real number $s$, $0 \leqslant s < 1$, such that $d(T(x), T(y)) < s \cdot d(x, y)\ \forall x, y \in \mathbb{X}$.*

The concept of contractive transformation appears implicitly in the self-similarity property when we wish to transform an object into one of its parts, which is necessarily smaller than the object itself. We shall see later that this condition is convenient but not necessary.

EXAMPLE.– Contractive transformations:

– $f : \mathbb{R} \to \mathbb{R}$ such that $f(x) = \frac{1}{2}x + \frac{1}{2}$, $(s = \frac{1}{2})$ ;

– $f : \mathbb{R}^2 \to \mathbb{R}^2$ such that $f(x, y) = \left(\frac{1}{3}x, \frac{1}{3}y\right)$ , $(s = \frac{1}{3})$.

Here, we introduce a fundamental theorem on contractive transformations: the Banach fixed-point theorem. In our context, as we shall see hereafter, this theorem justifies the relevance of the definition of self-similarity by proving the existence and unicity of the object associated with a given self-similarity.

THEOREM.– Let $(\mathbb{X}, d)$ be a complete metric space and $T$ a contractive transformation on $(\mathbb{X}, d)$. There then exists a single point $x \in \mathbb{X}$ called a fixed-point of $T$, verifying $T(x) = x$.

EXAMPLE.–

– The function $f(x) = \frac{1}{2}x + \frac{1}{2}$ ($x \in \mathbb{R}$) has as unique fixed point $x^F = 1$ : $f(1) = 1$.

– The function $f(x, y) = (\frac{1}{3}x, \frac{1}{3}y)$ ($(x, y) \in \mathbb{R}^2$) has as unique fixed point $x^F = (0, 0) : f(0, 0) = (0, 0)$.

If a function is not contractive, it may have several fixed points, or none.

– The function $f(x) = x^2$ ($x \in \mathbb{R}$) has two fixed points $x_0^F = 0$ and $x_1^F = 1$ ($f(0) = 0$ and $f(1) = 1$).

– The identity function possesses an infinite number of fixed points.

– The function $f(x) = x + 1$ ($x \in \mathbb{R}$) has no fixed point.

### 1.1.2. *Self-similarity*

DEFINITION.– *Let $(\mathbb{X}, d)$ be a complete metric space and $(\mathcal{H}(\mathbb{X}), d_{\mathbb{X}})$ the space of objects associated with it. $K \in \mathcal{H}(\mathbb{X})$ is self-similar if there exists a finite number $N$ of contractive transformations on $\mathbb{X}$, $T_i, i = 0 \cdots, N - 1$, such that:*

$$K = \bigcup_{i=0}^{N-1} T_i(K)$$

*We will then say that the object $K$ possesses the self-similarity property relating to the transformation set $\{T_i, i = 0 \cdots, N - 1\}$. It is useful to identify transformations using indices to differentiate them and to then explain formulations and algorithms. The order, however, is arbitrary and has no impact on the geometry of objects. We denote by $\Sigma = \{0, \cdots, N - 1\}$, the set of these indices.*

The previous definition specifies the intuitive idea introduced in section 1.1. The objects that we are looking at are compacts subsets of a metric space, that is, closed and bounded sets. The nature of the transformations describing self-similarity expresses what we mean by "composed of copies of itself". These transformations are supposed to be contractive, which seems natural because part of the object must be smaller than the object itself. This condition of contraction is sufficient but not necessary. A part is necessarily included in the object, but locally it could be stretched over certain areas, or in a given direction, and contracted in another direction (see Figure 1.5).

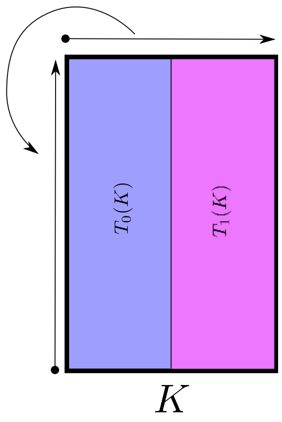


**Figure 1.5.** *Example of self-similarity involving non-contractive transformations, $K = T_0(K) \cup T_1(K)$. For a color version of this figure, see www.iste.co.uk/gentil/geometric.zip*

COMMENTS ON FIGURE 1.5.– *The transformation $T_0$ transforms the overall rectangle $K$ (whose edge is a thick black line) over the blue rectangle by rotating and contracting in one direction and dilating in the other: the edge referred to by the horizontal arrow is transformed into the edge referred to by the vertical edge. The transformation $T_1$ operates analogously for the pink rectangle.*

### 1.1.3. *Examples*

#### 1.1.3.1. *The Cantor set*

One of the simplest examples is that of the Cantor set. It is iteratively defined from the $[0, 1]$ segment. This segment is divided into three segments of equal length and the middle segment is removed. The same process is applied to each of the remaining segments (see Figure 1.6). The operation is repeated endlessly. The transformations of $\mathbb{R}$ describing this self-similarity are: $T_0(x) = \frac{x}{3}$ and $T_1(x) = \frac{x}{3} + \frac{2}{3}$.

**Figure 1.6.** *The Cantor set successively represented at the iteration levels from 1 to 5. For a color version of this figure, see www.iste.co.uk/gentil/geometric.zip*

### 1.1.3.2. *The Cartesian product of two Cantor sets*

The Cartesian product of two Cantor sets can be directly obtained from the definition of the Cartesian product, or by simply defining self-similarity, as presented by Figure 1.7. This self-similarity can be described by a set of four transformations of $\mathbb{R}^2$:

$$T_0\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{3}&0\\0&\frac{1}{3}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix},\quad T_1\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{3}&0\\0&\frac{1}{3}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}\frac{2}{3}\\0\end{pmatrix}$$

$$T_2\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{3}&0\\0&\frac{1}{3}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}\frac{2}{3}\\\frac{2}{3}\end{pmatrix},T_3\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{3}&0\\0&\frac{1}{3}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}0\\\frac{2}{3}\end{pmatrix}$$

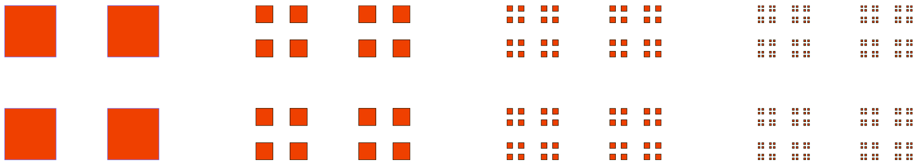


**Figure 1.7.** *Cartesian product of two Cantor sets successively represented at iteration levels from 1 to 4. For a color version of this figure, see www.iste.co.uk/gentil/geometric.zip*

### 1.1.3.3. *The Sierpinski triangle*

The Sierpinski triangle is also a classic example. It is built from a solid triangle, subdivided, and four triangles, from which the central triangle is removed. The operation is iterated for each of the resulting triangles (see Figure 1.8). The three transformations of $\mathbb{R}^2$ for this Sierpinski triangle are homotheties of ratio $\frac{1}{2}$ centered at a vertex of the triangle:

$$T_0\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{2}&0\\0&\frac{1}{2}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}-\begin{pmatrix}\frac{1}{2}\\0\end{pmatrix}$$

$$T_1\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{2}&0\\0&\frac{1}{2}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}\frac{1}{2}\\0\end{pmatrix}$$

$$T_2\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}\frac{1}{2}&0\\0&\frac{1}{2}\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}0\\\frac{1}{2}\end{pmatrix}$$

### 1.1.3.4. *The Menger sponge*

The Menger sponge is built from a square decomposed into nine squares of the same size, removing the central square. Figure 1.9 illustrates the recursive construction. Self-similarity is described by eight homotheties of ratio $\frac{1}{3}$, with the respective centers of the four vertices of the square and the four middles of the sides square.