

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

Table of Contents

[Cover](#)

[Title page](#)

[Copyright](#)

[Preface](#)

[Introduction](#)

[1.1. Fractals for industry: what for?](#)

[1.2. Fractals for industry: how?](#)

[1 The BC-IFS Model](#)

[1.1. Self-similarity and IFS](#)

[1.2. Controlled Iterated Function System](#)

[1.3. Boundary controlled iterated function system](#)

[2 Design Examples](#)

[2.1. Curves](#)

[2.2. Wired structures](#)

[2.3. Surfaces and laces](#)

[2.4. Volumes and lacunar objects](#)

[2.5. Tree structures](#)

[2.6. Form assembly](#)

[3 Surface NURBS, Subdivision Surfaces and BC-IFS](#)

[3.1. Bezier curves and surfaces](#)

[3.2. Uniform B-spline curves and surfaces](#)

[3.3. Generalization](#)

[3.4. NURBS curves](#)

[3.5. Subdivision curves and surfaces](#)

[4 Building Operations, Assistance to Design and Applications](#)

[4.1. Topological consistency and symmetry constraints](#)

[4.2. Topological combination](#)

[4.3. Applications](#)

[Conclusion](#)

[Appendix: Data of Figures](#)

[A.1. Data of figures](#)

[A.2. Subdivision surface in Figure 3.6](#)

[References](#)

[Index](#)

[Other titles from ISTE in Numerical Methods in Engineering](#)

[End User License Agreement](#)

List of Illustrations

Introduction

[Figure I.1. 3D tree built by iterative modeling \(source: project MODITERE no. AN...](#)

Chapter 1

[Figure 1.1. Schematic illustration of self-similarity. The black tree can be see...](#)

[Figure 1.2. An example of a self-similar object composed of five copies of itsel...](#)

[Figure 1.3. The self-similarity property, as shown in Figure 1.2, is symbolized ...](#)

[Figure 1.4. Hausdorff distance. For a color version of this figure, see www.iste...](#)

[Figure 1.5. Example of self-similarity involving non-contractive transformations...](#)

[Figure 1.6. The Cantor set successively represented at the iteration levels from...](#)

[Figure 1.7. Cartesian product of two Cantor sets successively represented at ite...](#)

[Figure 1.8. The Sierpinski triangle successively represented at iteration levels...](#)

[Figure 1.9. The Menger sponge successively represented at iteration levels from ...](#)

[Figure 1.10. Example of Romanesco broccoli consisting of seven self-similar elem...](#)

[Figure 1.11. On the left-hand side, we provide a few examples of self-similarity...](#)

[Figure 1.12. Example of a decomposition of an L-shape into several similar eleme...](#)

[Figure 1.13. Example of self-similarity. The object on the left-hand side has a ...](#)

[Figure 1.14. Lattice structure of the attractors. On the left, the lattice struc...](#)

[Figure 1.15. An example of a connection between two attractors. The green attrac...](#)

[Figure 1.16. The evaluation tree of the attractor of the IFS computed at the thi...](#)

[Figure 1.17. Example of the parameterization of the attractor in Figure 1.13. On...](#)

[Figure 1.18. Example of a transport mapping that defines a morphism of IFS. For ...](#)

[Figure 1.19. Example of mapping between two attractors using the transport map. ...](#)

[Figure 1.20. Attractor defining the parameter space for the Sierpinski triangle ...](#)

[Figure 1.21. Automaton of an IFS \$T = \{T_0, T_1, T_2\}\$. The transition \$i\$ is associated with the tran...](#)

[Figure 1.22. Example of a three-state automaton inducing a restriction of the se...](#)

[Figure 1.23. Both images represent the attractors defined from the same transfor...](#)

[Figure 1.24. Other examples of attractors built from the same automata as thos...](#)

[Figure 1.25. Automaton generating the union of two attractors. Transitions 0 and...](#)

[Figure 1.26. The internal structure of the Menger sponge. On the left, the inter...](#)

[Figure 1.27. Automaton describing the structure of the image on the right-hand s...](#)

[Figure 1.28. Example of a two-state automaton: the \$\square\$ is divided into four \$\triangle\$ and ...](#)

[Figure 1.29. Construction of the sequence converging to the attractor of the aut...](#)

[Figure 1.30. Approximation of the automaton attractor of Figure 1.28 obtained wi...](#)

[Figure 1.31. Evaluation tree developed at level 2, for the attractor of the auto...](#)

[Figure 1.32. Example of a third-degree B-spline surface defined from a grid of c...](#)

[Figure 1.33. The surface, at the top right, is a smooth B-spline surface and has...](#)

[Figure 1.34. Example of a curve constructed based on an FIF. The parallelepipeds...](#)

[Figure 1.35. Barycentric space. On the left, the barycentric space of dimension ...](#)

[Figure 1.36. Cantor set built in the barycentric space \$BI^2\$ using the IFS compose...](#)

[Figure 1.37. Sierpinski triangle built in the barycentric space \$BI^3\$. For a color...](#)

[Figure 1.38. Example of projections of the Sierpinski triangle. The attractor is...](#)

[Figure 1.39. Example of a two-state automaton. The \$\square\$ is divided into three \$\square\$ and...](#)

[Figure 1.40. Three different projections of the attractor described by the autom...](#)

[Figure 1.41. Automaton defining the attractor in the barycentric spaces and perf...](#)

[Figure 1.42. Curve of the "Takagi" type, defined from three control points and t...](#)

[Figure 1.43. Incidence constraints. On the left: Three curves of the "Takagi" ty...](#)

[Figure 1.44. Example of the construction of a connection between the subdivision...](#)

[Figure 1.45. Automaton integrating the cellular decomposition of a curve subdivi...](#)

[Figure 1.46. Tree for a curve. For a color version of this figure, see \[www.iste....\]\(http://www.iste...\)](#)

[Figure 1.47. Quotient graph for a curve. For a color version of this figure, see...](#)

[Figure 1.48. Example of curves generated for different parameter values. For a c...](#)

[Figure 1.49. Attractor built from the IFS \$\{B_0^C, B_1^C\}\$ whose subdivision operators correspo...](#)

[Figure 1.50. Subdivision structure of the tile. For a color version of this figure...](#)

[Figure 1.51. Automaton for the subdivision of a quadrangular surface. For a color...](#)

[Figure 1.52. Cell structure of a quadrangular tile. For a color version of this ...](#)

[Figure 1.53. Example of a quadrangular surface. For a color version of this figure...](#)

[Figure 1.54. Example of a quadrangular surface bordered by Bezier curves with an...](#)

[Figure 1.55. Example of a quadrangular surface structure bordered by Bezier curv...](#)

[Figure 1.56. Example of a surface structure with fractal topology, obtained from...](#)

[Figure 1.57. Example of the quadrangular surface bordered by Bezier curves with ...](#)

[Figure 1.58. Example of curves projected into the modeling space \$\mathbb{R}^2\$, following th...](#)

[Figure 1.59. Example of curves projected into the modeling space \$\mathbb{R}^2\$, following th...](#)

[Figure 1.60. Example of curves projected into the modeling space \$\mathbb{R}^2\$, following th...](#)

[Figure 1.61. Example of a network of control points for a triangular surface tha...](#)

Chapter 2

[Figure 2.1. "Standard" automaton for a curve. Transitions referred to by the sym...](#)

[Figure 2.2. Examples of curves constructed using two transformations. The contro...](#)

[Figure 2.3. A tree illustrating another possibility for connecting the subdivisi...](#)

[Figure 2.4. Comparison of the effect of the two types of connection, standard an...](#)

[Figure 2.5. Examples of curves constructed using three transformations whose ver...](#)

[Figure 2.6. Examples of von Koch curves constructed with two transformations \(to...](#)

[Figure 2.7. Left-hand column: for the first figure, the first subdivision point ...](#)

[Figure 2.8. Automaton describing a CA curve subdivided into a CA-type curve and ...](#)

[Figure 2.9. Example of a curve obtained with two mutually referenced states acco...](#)

[Figure 2.10. Tree illustrating the construction of a connection for a wired stru...](#)

[Figure 2.11. Example of two wired structures built from the incidence and adjace...](#)

[Figure 2.12. Examples of wired structures. The diagrams above each shape represe...](#)

[Figure 2.13. Diagram of the cellular decomposition of a quadrangular surface wit...](#)

[Figure 2.14. Example of subdivision of a quadrangular surface with "non-standard...](#)

[Figure 2.15. Connections to build a Hilbert/Peano curve. The red circles show th...](#)

[Figure 2.16. Example of a curve attempt that satisfies the adjacency relations i...](#)

[Figure 2.17. Subdivision of a quadrangular surface satisfying the constraints of...](#)

[Figure 2.18. Example of a Hilbert/Peano surface, defined from the subdivision of...](#)

[Figure 2.19. The diagram on the right-hand side presents the quadrangular subdiv...](#)

[Figure 2.20. To achieve a quadrangular surface from two subdivisions, we need to...](#)

[Figure 2.21. Automaton symbolizing the subdivision system of a quadrangular surf...](#)

[Figure 2.22. Example of a quadrangular surface with two subdivisions. For this i...](#)

[Figure 2.23. Standard triangular subdivision. The triangular face is subdivided ...](#)

[Figure 2.24. Standard triangular subdivision, but with connections differing fro...](#)

[Figure 2.25. Pentagonal subdivision. On the left, the incidence relations are sy...](#)

[Figure 2.26. On the left, incidence relations are symbolized using red dotted li...](#)

[Figure 2.27. Cellular decomposition and subdivision of the Sierpinski triangle. ...](#)

[Figure 2.28. Cellular decomposition and subdivision of the Sierpinski triangle w...](#)

[Figure 2.29. Example of a Sierpinski triangle whose face, edges and vertices hav...](#)

[Figure 2.30. Example of a Sierpinski triangle whose edges are uniform quadratic ...](#)

[Figure 2.31. Penrose tiling of the "kite" type at iterations 1, 2, 3 and 6. For ...](#)

[Figure 2.32. Topological subdivision diagram of the faces and edges representing...](#)

[Figure 2.33. Example of 3D surfaces constructed from the topological subdivision...](#)

[Figure 2.34. Examples of the Menger sponge, represented with three iteration lev...](#)

[Figure 2.35. Example of the construction of a 2D tree structure, consisting of t...](#)

[Figure 2.36. Subdivision process of the tree structure of Figure 2.35 and cell d...](#)

[Figure 2.37. Automaton describing the iterative construction process of the tree...](#)

[Figure 2.38. Examples of projection of the tree's topological structure, defined...](#)

[Figure 2.39. Subdivision process of a tree structure whose trunk is subdivided i...](#)

[Figure 2.40. Automaton describing the iterative construction process of the tree...](#)

[Figure 2.41. Example of a tree structure whose trunk \(green\) is not subdivided \(...\)](#)

[Figure 2.42. Simplified representation of the incidence and adjacency relations ...](#)

[Figure 2.43. 3D tree built on the principle of space tiling\(source: project MODI...](#)

[Figure 2.44. Example of assembling fractal structures \(built from an octagonal f...](#)

[Figure 2.45. Example of an assembly of triangular surface structures. The basic ...](#)

[Figure 2.46. Examples of assemblies of pentagonal fractal faces following a dode...](#)

[Figure 2.47. Assembly of 10 3D Penrose tilings of the "kite"-type to form the co...](#)

[Figure 2.48. Examples of assemblies built from Menger sponges and manufactured b...](#)

Chapter 3

[Figure 3.1. On the left is an example of a quadratic Bezier curve, defined by th...](#)

[Figure 3.2. C-IFS automaton whose attractor is a Bezier curve. This automaton si...](#)

[Figure 3.3. Illustration of the self-similarity property of uniform quadratic B...](#)

[Figure 3.4. Blending B-spline functions of the second degree. Their support is o...](#)

[Figure 3.5. The control points \(\$P_0, P_1, P_2, P_3\$ \) and the knot vector \(\$u_0, u_1, u_2, \dots\$](#)

[Figure 3.6. Subdivision scheme obtained by duplication of knots](#)

[Figure 3.7. Automaton of the C-IFS representing the iterative process of a secon...](#)

[Figure 3.8. From left to right: The non-uniform quadratic curve defined from fou...](#)

[Figure 3.9. Automaton of the C-IFS representing the subdivision of a third-degre...](#)

[Figure 3.10. The top diagram represents the knot interval vector of a curve of d...](#)

[Figure 3.11. Illustration of the extraction of the knot vectors from the two cur...](#)

[Figure 3.12. Subdivision process of a NURBS surface of degree 2, obtained by the...](#)

[Figure 3.13. Automaton of the C-IFS representing the subdivision of a NURBS surf...](#)

[Figure 3.14. C-IFS automaton whose attractor is a subdivision curve built from t...](#)

[Figure 3.15. Approximations of a uniform cubic B-spline curve for levels 0 to 6:...](#)

[Figure 3.16. Refinement of a regular control mesh. In red: Minimal control mesh ...](#)

[Figure 3.17. Self-similarity of the refined mesh. Each of the four blue sub-mesh...](#)

[Figure 3.18. Automaton of a subdivision surface for the Doo-Sabin scheme](#)

[Figure 3.19. Refinement of an irregular control mesh. In red: Minimal control me...](#)

[Figure 3.20. Self-similarity of the refined mesh. In blue, the four sub-meshes o...](#)

[Figure 3.21. Automaton of a surface subdivision for the Doo-Sabin scheme with an...](#)

[Figure 3.22. Adjacency and incidence constraints. On the left: Example of an adj...](#)

[Figure 3.23. Control points defining one of the irregular edges. The irregular e...](#)

[Figure 3.24. Illustration of incidence and adjacency constraints for an irregula...](#)

[Figure 3.25. Representation of the topological subdivision process of an irregul...](#)

[Figure 3.26. Cell decomposition of an irregular tile. The constraints are repres...](#)

[Figure 3.27. Subdivision of an irregular tile with an irregular face of six side...](#)

[Figure 3.28. Catmull subdivision of a regular tile](#)

[Figure 3.29. The four sub-meshes of \$4 \times 4\$ control points are represented in blue...](#)

[Figure 3.30. Example of an irregular mesh with a vertex of valence five. After r...](#)

[Figure 3.31. Illustration of the Loop subdivision. Figure 3.31\(a\) is an illustra...](#)

[Figure 3.32. Refinement of a regular control mesh for the Loop scheme. The struc...](#)

[Figure 3.33. Self-similarity of the refined mesh for the Loop scheme. Decomposit...](#)

Chapter 4

[Figure 4.1. Example of the user interface of "MODITERE", the iterative modeler d...](#)

[Figure 4.2. Example of orientation constraints of edges for the definition of th...](#)

[Figure 4.3. Example of configuration of edge orientation for the attractor of Fi...](#)

[Figure 4.4. Issues in the orientation of edges. On the left, the connection for ...](#)

[Figure 4.5. Definition and application of the permutation operator. On the left,...](#)

[Figure 4.6. Construction of a connection between two edges of opposite orientati...](#)

[Figure 4.7. The three curves have a single internal dimension, and vertices of d...](#)

[Figure 4.8. The definition of the connections for volume cells can prove complex...](#)

[Figure 4.9. Illustration of the definitions of face permutations: on the left, a...](#)

[Figure 4.10. Examples of construction of the topological tensor product. For a c...](#)

[Figure 4.11. Example of cellular decomposition of a tensor product of two curves...](#)

[Figure 4.12. Automaton of the curve a](#)

[Figure 4.13. Quotient graph of curve a](#)

[Figure 4.14. Automaton of a surface automatically generated from the automata ...](#)

[Figure 4.15. Quotient graph of a surface induced by adjacency relations combin...](#)

[Figure 4.16. Quotient graph obtained from adjacency relations on incidence opera...](#)

[Figure 4.17. Automaton of a volume structure obtained by tensor product of three...](#)

[Figure 4.18. Example of an automatically generated tree, whose leaves will perfe...](#)

[Figure 4.19. First stages of subdividing a tree bordered by a curve](#)

[Figure 4.20. The types of subdivisions of the tree depend on the types of edge s...](#)

[Figure 4.21. Automaton representing a tree structure whose set of leaves is divi...](#)

[Figure 4.22. Double trees bordered by quadrangular surfaces](#)

[Figure 4.23. Example of surface obtained by the tensor product of a NURBS curve ...](#)

[Figure 4.24. Automaton representing the construction of a connection structure b...](#)

[Figure 4.25. Connection subdivision process. The red, green and blue mesh, respe...](#)

[Figure 4.26. Connections built between different pairs of surfaces: a Doo-Sabin ...](#)

[Figure 4.27. Example of a lacunar surface, co-imagined by architects \(IBOIS-EPFL...](#)

[Figure 4.28. Example of the definition of a 2D lacunar structure by tiling the p...](#)

[Figure 4.29. Example of construction of the Menger sponge. In red, the cube is d...](#)

[Figure 4.30. Lacunar structures obtained by removing the central part of a regul...](#)

[Figure 4.31. Menger-Excoffier sponge with walls of the same thickness between ea...](#)

[Figure 4.32. The Menger-Excoffier sponge is built from two subdivision systems o...](#)

[Figure 4.33. For the Menger-Excoffier sponge, the two 3D subdivision systems are...](#)

[Figure 4.34. "Triangle"-type cells have been added to the structure of Figure 4....](#)

[Figure 4.35. Design of a lacunar structure from two types of cell: a tetrahedron...](#)

[Figure 4.36. Two copies of the 3D lacunar topological structure obtained from th...](#)

[Figure 4.37. Prototypes designed by assembly of tetrahedral structures shown in ...](#)

[Figure 4.38. Illustration of the topological subdivision of the lacunar face. Fo...](#)

[Figure 4.39. Lacunar face at iteration levels 3 \(on the left\) and 4 \(in the cent...](#)

[Figure 4.40. Illustration of the cellular decomposition and the subdivision proc...](#)

[Figure 4.41. Example of the design of a lacunar structure, built from a truncate...](#)

[Figure 4.42. Example of filling using porous volumes. On the left, the geometry ...](#)

[Figure 4.43. Examples of the design of lacunar structures through the assembly o...](#)

[Figure 4.44. Assembly according to the structure of a diamond. On the left, an e...](#)

[Figure 4.45. Examples of rough surfaces. For a color version of this figure, see...](#)

[Figure 4.46. Example of a variant of the Von Koch curve filling up almost an ent...](#)

[Figure 4.47. Example of a rough, or even chaotic, surface, designed on the princ...](#)

[Figure 4.48. Both surfaces are designed from the same topological structure as s...](#)

[Figure 4.49. Example of a self-supporting hull, built by wood panel assembly \(IB...](#)

[Figure 4.50. On top, prototype of the thermal exchanger manufactured in aluminum...](#)

[Figure 4.51. Construction of the display primitive linking two levels of the exc...](#)

[Figure 4.52. Internal structure of the switch. For a color version of this figur...](#)

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.

Apart from any fair dealing for the purposes of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act 1988, this publication may only be reproduced, stored or transmitted, in any form or by any means, with the prior permission in writing of the publishers, or in the case of reprographic reproduction in accordance with the terms and licenses issued by the CLA. Enquiries concerning reproduction outside these terms should be sent to the publishers at the undermentioned address:

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

© ISTE Ltd 2021

The rights of Christian Gentil, Gilles Gouaty and Dmitry Sokolov to be identified as the authors of this work have been asserted by them in accordance with the Copyright, Designs and Patents Act 1988.

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

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.

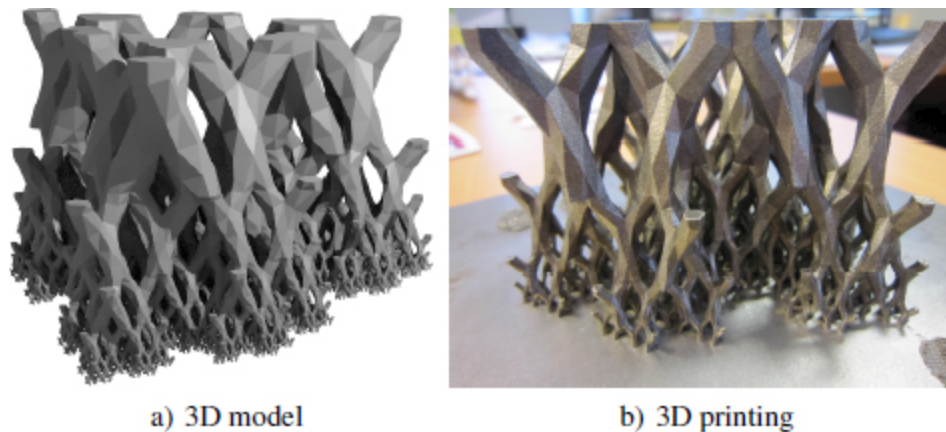


Figure I.1. *3D tree built by iterative modeling*

(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).

1.2. 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.

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

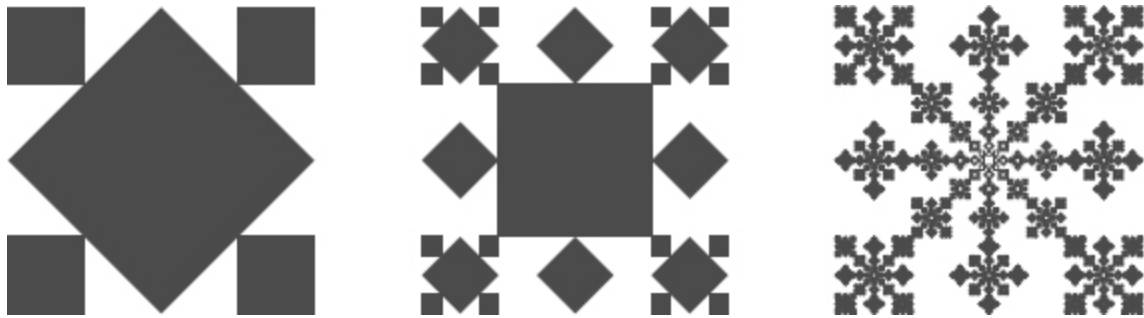


Figure 1.2. *An example of a self-similar object composed of five copies of itself. These five main parts are symbolized on the left by the five squares. In the center, each square has been replaced by the overall form composed of the five parts. On the right-hand side, the same construction process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