Simulation of Sierpinski-type fractals and their geometric constructions in Matlab environment

Zhiyong Zhu*
Northwest A&F University
College of Science
Taicheng Road 3, 712100 Yangling
China
yzzhu0412@163.com

Enemi Dong
Northwest A&F University
College of Science
Taicheng Road 3, 712100 Yangling
China
enmei117@sohu.com

Abstract: Study on properties of Sierpinski-type fractals, including dimension, measure, connectedness, Lipschitz equivalence, etc are very interesting. Although there have been some very nice results were obtained, there is still a long way to go to solve all the problems. In order to facilitate understanding of these results and further study, in this paper, we simulate this kind of fractals and their geometric constructions in Matlab environment that is more easily understood and mastered for researcher base on the recursive and iterative algorithms that are used to simulate fractals. Furthermore, our results are also interesting results to enrich the theoretical and applied research of fractal simulation.

Key–Words: Sierpinski gasket-type fractal, Sierpinski carpet-type fractal, Fractal simulation, Recursive algorithm, Iterative algorithm, Matlab

1 Introduction

For an given integer $n \geq 2$, let $\mathcal{D} \subset \{0, \ldots, n-1\}^2$. We shall call the unique non-empty compact subset $E_{\mathcal{D}} \subset \mathbb{R}^2$ satisfying the following set equation

$$E_{\mathcal{D}} = (E_{\mathcal{D}} + \mathcal{D})/n$$

a Sierpinski carpet-type fractal throughout this paper. If the above set $\{0, \ldots, n-1\}^2$ is replaced by $\Delta = \{k_1\alpha + k_2\beta : k_1 + k_2 \leq n-1 \text{ and } k_1, k_2 \in \mathbb{N} \cup \{0\}\}$,where $\alpha = (1, 0)$ and $\beta = (1/2, \sqrt{3}/2)$, then we call $E_{\mathcal{D}}$ a Sierpinski gasket-type fractal. In particular, for $n = 3$, $\mathcal{D} = \{(0,0), (1,0), (2,0), (0,1), (2,1), (0,2), (1,2), (2,2)\} \subset \{0, 1, \ldots, n-1\}^2$, $E_{\mathcal{D}}$ is well-known Sierpinski carpet, and for $n = 2$, $\mathcal{D} = \{(0,0), (1,0), (1/2, \sqrt{3}/2)\} \subset \Delta$, $E_{\mathcal{D}}$ is well-known Sierpinski gasket. Let $I = [0, 1]^2$ or be the following set

$$\{c_1\alpha + c_2\beta : c_1 + c_2 \leq 1 \text{ and } 0 \leq c_1, c_2 \leq 1\}.$$

We define $E^1_{\mathcal{D}} = (I + \mathcal{D})/n$, and recurrently, $E^{k+1}_{\mathcal{D}} = (E^k_{\mathcal{D}} + \mathcal{D})/n$ for $k \geq 1$. Then $E^k_{\mathcal{D}}$ is a union of squares(equilateral triangles) of side $1/n^k$. Clearly, $E^{k+1}_{\mathcal{D}} \subset E^k_{\mathcal{D}}$ and $E_{\mathcal{D}} = \cap_{k=1}^{\infty} E^k_{\mathcal{D}}$.

In recent years, study on properties of Sierpinski-type fractals described as above, including dimension, measure, connectedness, Lipschitz equivalence, etc have aroused wide concern ([2,4,7-10,13-16,18-21]). Although there have been some very nice results were obtained, there is still a long way to go to solve all the problems. It is well know that the in-depth analysis and understanding of topological structure of fractals are very important to study properties of fractals. In this process, simulate the fractal and its geometric construction by computer, will provide us the most intuitive discussion and explanation. The computer simulation of fractal patterns is based on the basic theory of fractal, is one of the most popular area of study in computer graphics at present. Study the algorithms and programs of drawing fractal not only can offer technical support for the study of fractal theory, but also can stimulate creative inspiration, to further enrich the content of computer graphics, and also can drive a lot of old discipline in newborn. For example, the complex analysis don't obtain further development for many years, until the computer graphics display completely the complex structures of Julia set, the mathematical theory of this discipline was once again leap development. At present, although many works(computer programming) have been devoted to the study of the computer simulation of Sierpinski-type fractals, especially the classic Sierpinski carpet and Sierpinski gasket [1, 5, 11, 17], these source codes of computer programs that we saw in these works are mostly written by the programming language: Java, C++, Delphi, Java, etc. Thus these works are usually

not easy to be understood and applied for researchers to engage in mathematics study, unless have the corresponding knowledge of programming language. In addition, since the majority of these works came from the researchers and engineering and technical personnel that engage in the fields of computer graphics and digital image processing etc, thus these works usually tend to ignore mathematical problems in study, and focus on the design and the realization of computer algorithms of drawing fractal. So the different fine structures of $E_{\mathcal{D}}^{k}$ and $E_{\mathcal{D}}^{k+1}$ mentioned as above can't well described by these works on one hand, but on the other hand we also can't easy to obtain any Sierpinski-type fractal pattern is what we need in the study only by modifying a few parameters. But these contents just are the most important for researchers of studying the properties of such fractals. How to use a popular and easy-to-understand way as much as possible, in a real-time information exchange interface, such that the researchers can well-obtain any pattern is what they need in the study of a class of fractal patterns by modifying a few parameters, without need to know much about the knowledge of complex programming language, at the same time complete some relative functions such as color adjustment, graphical comparison and storage etc, and generate fractal patterns with certain theoretical research value, has turned into one of the hot problems concerned by many researchers. For this motive and note that the powerful advantages of Matlab in numerical calculation and graphic visual ability and its programming language is more easily understood and mastered by mathematical researchers. In this paper, we study the simulation of the above-mentioned Sierpinski-type fractals and their geometric constructions in Matlab environment base on the recursive theory and iterative theory that are used to simulate fractals. Our results not only can well-describe the different fine structures of $E_{\mathcal{D}}^{k}$ and $E_{\mathcal{D}}^{k+1}$ in the process of constructing Siepinski-type fractal $E_{\mathcal{D}}$, but also can well present different patterns are what we need in study only by modifying a few parameters.

The rest of this paper is organized as follows. In section 2, some basic facts and known concepts needed in our discussion are described, for details, see [4, 7, 12, 17]. An algorithm and a computer program of simulating the Sierpinski gasket-type fractals and their geometric constructions are given base on iterative and recursive theory in section 3. In section 4, besides an program similar to section 3 is given, we will give the other algorithm and program of simulating the Sierpinski carpet-type fractals and their geometric constructions only by recursive theory.

2 Some basic facts and concepts

2.1 Function M-files

For simple problems, entering your requests at the Matlab command window prompt is fast and efficient. However, as the number of commands increases or trial and error is done by changing certain variables or values, typing the commands over and over at the Matlab command window prompt becomes tedious. To get around this, we enter the commands into a text file (called M-files or Script M-files [12]), and execute them by simply typing filename at the Matlab command window prompt. All function M-file names must end with the extension '.m' (e.g. test.m). Function M-files are like Script M-files but can pass parameters and isolate variables. The structure of a typical function M-file, say my fun.m, is as follows:

function[output arguments]=my fun(input arguments)
code
.....
code
.....

Note that the word function appears at the start of the file. In addition, the output arguments and input arguments and name of the function are listed. If a function only has a single output argument, then the square brackets are not required. If a function does not have any output arguments, then neither the square brackets nor the equals sign that follows are used.

2.2 Recursion Theory

Recursion is the process of repeating items in a self-similar way [5]. In computer programming, recursion is a function that can directly or indirectly call itself. A general syntax of recursion in Matlab looks like this:

function Recur(n)
.....
Recur(m)
.....

Recursion is a method of solving problems based on the divide and conquer mentality. The basic idea is that you take the original problem and divide it into smaller (more easily solved) instances of itself, solve those smaller instances (usually by using the same algorithm again) and then reassemble them into the final solution. In this paper, the recursive algorithm will be used in two places, one is to divide each squares(triangles) in $E_{\mathcal{D}}^{k}$ into n^2 equal-sized squares(triangles) for any k, the other is to fill the squares are removed after each division with white.

2.3 Theory of Iterated Function System(IFS)

We call the finite set $\{f_i : i = 1, 2, \ldots, N\}$ is an iterated function system if each f_i is a contraction mapping on a complete metric space. Hutchinson [7] showed that, for the metric space $\mathbb{R}^d$, such a system of functions has a unique nonempty compact set(closed and bounded) E such that

$$E = \bigcup_{i=1}^{N} f_i(E)$$

and for any nonempty compact subset $A \subset \mathbb{R}^d$, we have

$$E = \bigcap_{k=1}^{\infty} S^k(A) = \lim_{k\to\infty} S^k(A), \qquad (1)$$

where $S(A) = \cup_{i=1}^{N} f_i(A), S^{k+1}(A) = S(S^k(A))$ for $k \geq 1$. There are two iteration algorithms for generating fractals using an iterated function systems. They are deterministic algorithm and random iteration algorithm. The set equation (1) is the basis of constructing fractals using deterministic algorithms. In this paper, we will use deterministic algorithm to computer all the lower-left coordinates of squares(triangles) in $E_{\mathcal{D}}^k$ for any $k \geq 1$.

3 Simulation of the gasket-type fractal and its geometric construction

The geometric construction of the Sierpinski gasket-type fractals $E_{\mathcal{D}}$ mentioned in Introduction is the following: we divide an equilateral triangle with side 1 into n^2 congruent triangles, choose some triangles according to the rule [2] described by $\mathcal{D}$, again divide each chosen triangles into n^2 congruent ones, choose the triangle according to the same rule and repeat the procedure inductively to the infinity, we can obtain the set $E_{\mathcal{D}}$. For any $k \geq 1$, $E_{\mathcal{D}}^k$ here is the union of triangles that are chosen in the step k. Let

$$f_i(x) = \frac{1}{n}(x + a_i), x \in \mathbb{R}^2, \qquad (2)$$

where $a_i \in \mathcal{D}$, $i = 1, 2, \ldots, \sharp\mathcal{D}$. Then $E_{\mathcal{D}}$ is unique invariant set of such a iterated function systems.

3.1 Algorithm and Procedure

The steps of creating the Sierpinski-gasket fractals as following:

① Suppose (x, y) is the lower-left coordinate of an equilateral triangle with side d.

② Divide this equilateral triangle into n^2 congruent triangles and computer all the lower-left coordinates of triangles that chosen according to certain rule by certain transformation equations. For example, we may compute the lower-left coordinates of triangles in $E_{\mathcal{D}}^1$ by (2).

③Assign the above calculated values to the variable of lower-left coordinate of initial equilateral triangle respectively and ready to participate in the calculation.

④ Suppose the iterated depth is k, circularly execute procedure ②∼③.

⑤ Fill all of triangles with side d/n^k and lower-left coordinates obtained by the above calculation with blue.

3.2 Program Design

function Sierpinskitriangle(M, x, y, d, n, k)
% SIERPINSKITRIANGLE, Display the geometric construction process of Sierpinski gasket-type fractal.
% Call format: Sierpinskitriangle(M, x, y, d, n, k).
% $M = [a_1, a_2, \ldots, a_m; b_1, b_2, \ldots, b_m]$. If we type it at the Matlab command window prompt, we can obtain the following $2 \times m$ matrix:

$$\begin{pmatrix} a_1 & a_2 & \ldots & a_m \\ b_1 & b_2 & \ldots & b_m \end{pmatrix},$$

where $m \geq 1$ is an integer, $(a_i, b_i) \in \mathbb{R}^2$, $i = 1, \ldots, m$, are the lower-left coordinates of triangles that chosen according to certain rule after the first division for initial equilateral triangles.
% x is the $x-$coordinate of lower-left corner of initial equilateral triangle.
% y is the $y-$coordinate of lower-left corner of initial equilateral triangle.
% d is the side length of initial equilateral triangle.
% $1/n$ is contraction radio.
% k is iterated depth.
$a = [x, x + d, x + d/2]$;
$b = [y, y, y + sqrt(3) * d/2]$;
plot(a, b) % Draw an equilateral triangle with side d.
hold on
for $j = 1 : k$
$a1 = []$; $b1 = []$; % Two empty arrays, will be used to store the x and y coordinates of lower-left corner of chosen triangles after the kth step construction respectively.
for $i =$ 1: length(x)
$x1 = x(i) + d * M(1, :)$;
$y1 = y(i) + d * M(2, :)$;
trianglegrid$(x(i), y(i), n - 1, d)$ % Divide the equilateral triangle with lower-left coordinate $(x(i), y(i))$ and side d into n^2 congruent triangles.
$a1 = [a1, x1]$;$b1 = [b1, y1]$;
end

$d = d/n$; $x = a1$; $y = b1$;
end
for $i = 1$: length(x)
fill$(x(i) + [0, d/2, d], y(i) + [0, sqrt(3) * d/2, 0],' b')$ % Fill the equilateral triangle with lower left corner $(x(i), y(i))$ and side d by blue.
hold on
end
hold off
axis off, axis equal
set(findobj(gcf,'type','patch'),'edgecolor','none');
function trianglegrid(x, y, r, s)
$t = r$; $c = []$;
while $t >= 0$
for $i = 0 : t$
$a = [x + i * s/(r + 1), x + (i + 1) * s/(r + 1), x + (2 * i + 1) * s/(2 * (r + 1)), x+ i * s/(r + 1)]$;
$b = [y, y, y + sqrt(3) * s/(2 * (r + 1)), y]$;
$c = [c, a]$;
plot(a, b)
hold on
end
$x = c(3)$; $y = b(3)$; $t = t - 1$; $c = []$;
end

The percent-sign (%) denotes a comment, everything following this sign up to the end of the sentence (next sentence break) will be ignored in this program. Saving the above text in a file called Sierpinskitriangle.m in your current directory. For given $\mathcal{D} \subset \{k_1\alpha + k_2\beta : k_1 + k_2 \leq n - 1 \text{ and } k_1, k_2 \in \mathbb{N} \cup \{0\}\}$, let $M_{\mathcal{D}} = [a_1, a_2, \ldots, a_{\mathcal{D}}; b_1, b_2, \ldots, b_{\mathcal{D}}]$, where $(na_i, nb_i) \in \mathcal{D}$ for each i. We can obtain the fine details of the kth step of constructing Sierpinski gasket-type fractal $E_{\mathcal{D}}$ (begin with an equilateral triangle with side 1 and lower-left coordinate $(0, 0)$) by running the following command:

$$\text{Sierpinskitriangle}(M_{\mathcal{D}}, 0, 0, 1, n, k)$$

at the Matlab command window prompt.

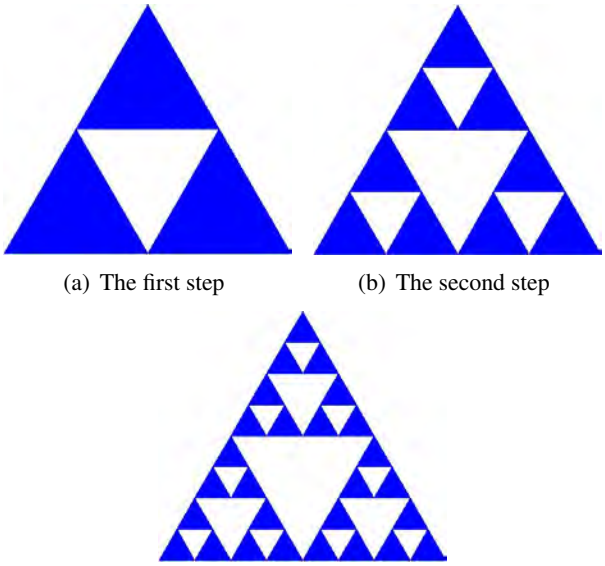
Example 1. *Let* $n = 2$,

$$\mathcal{D}_1 = \{(0, 0), (1, 0), (\frac{1}{2}, \frac{\sqrt{3}}{2})\}.$$

We can obtain the fine details of the first three steps of constructing Sierpinski gasket $E_{\mathcal{D}_1}$ *by running continuous the following commands: Sierpinskitriangle*$([0, 1/2, 1/4; 0, 0, sqrt(3)/4], 0, 0, 1, 2, k), k = 1, 2, 3$, *see Figure 1.*

Example 2. *Let* $n = 3$,

$$\mathcal{D}_2 = \{(0, 0), (1, 0), (\frac{3}{2}, \frac{\sqrt{3}}{2}, (1, \sqrt{3})\}.$$



(a) The first step (b) The second step

(c) The third step

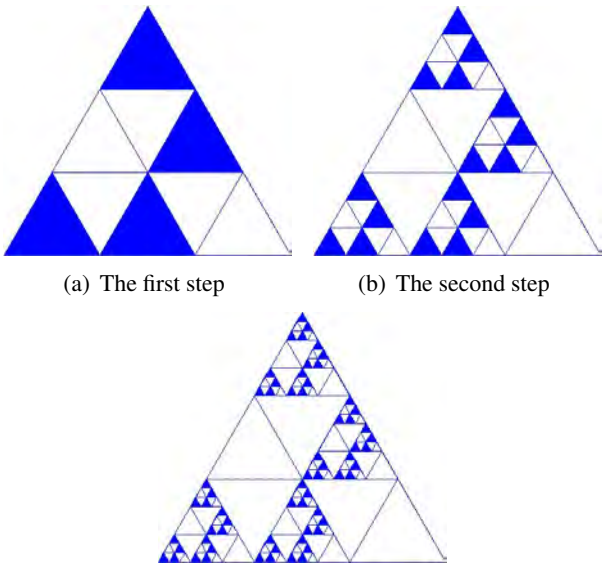
Figure 1: The first three steps of constructing $E_{\mathcal{D}_1}$.

We can obtain the fine details of the first three steps of constructing Sierpinski gsket-type fractal $E_{\mathcal{D}_2}$ *by running continuous the following commands: Sierpinskitriangle*$([0, 1/3, 1/2, 1/3; 0, 0, sqrt(3)/6, sqrt(3)/3], 0, 0, 1, 3, k), k = 1, 2, 3$, *see Figure 2.*

Similarity, take
$\mathcal{D}_3 = \{(0, 0), (1, 0), (2, 0)(3, 0), (\frac{3}{2}, \frac{\sqrt{3}}{2}), (1, \sqrt{3}), (\frac{3}{2}, \frac{3\sqrt{3}}{2})\}$;
$\mathcal{D}_4 = \{(1, 0), (2, 0), (\frac{1}{2}, \frac{\sqrt{3}}{2}), (\frac{3}{2}, \frac{\sqrt{3}}{2}), (\frac{5}{2}, \frac{\sqrt{3}}{2}), (1, \sqrt{3}), (2, \sqrt{3}), (\frac{3}{2}, \frac{3\sqrt{3}}{2})\}$;
$\mathcal{D}_5 = \{(1, 0), (3, 0), (\frac{1}{2}, \frac{\sqrt{3}}{2}), (\frac{3}{2}, \frac{\sqrt{3}}{2}), (\frac{5}{2}, \frac{\sqrt{3}}{2}), (\frac{7}{2}, \frac{\sqrt{3}}{2}), (2, \sqrt{3}), (\frac{3}{2}, \frac{3\sqrt{3}}{2})(\frac{5}{2}, \frac{3\sqrt{3}}{2})\}$;
$\mathcal{D}_6 = \{(2, 0), (\frac{1}{2}, \frac{\sqrt{3}}{2}), (\frac{3}{2}, \frac{\sqrt{3}}{2}), (\frac{5}{2}, \frac{\sqrt{3}}{2}), (\frac{7}{2}, \frac{\sqrt{3}}{2}), (1, \sqrt{3}), (2, \sqrt{3}), (3, \sqrt{3}), (\frac{3}{2}, \frac{3\sqrt{3}}{2}), (\frac{5}{2}, \frac{3\sqrt{3}}{2}), (2, 2\sqrt{3})\}$,
and run the the following commands in turn:
Sierpinskitrigle$(M_{\mathcal{D}_i}, 0, 0, 1, 4, 3)(i = 3, 4)$,
Sierpinskitrigle$(M_{\mathcal{D}_i}, 0, 0, 1, 5, 2)(i = 5, 6)$.
We can obtain the corresponding fine details as Figure 3.

4 Simulation of the carpet-type fractal and its geometric construction

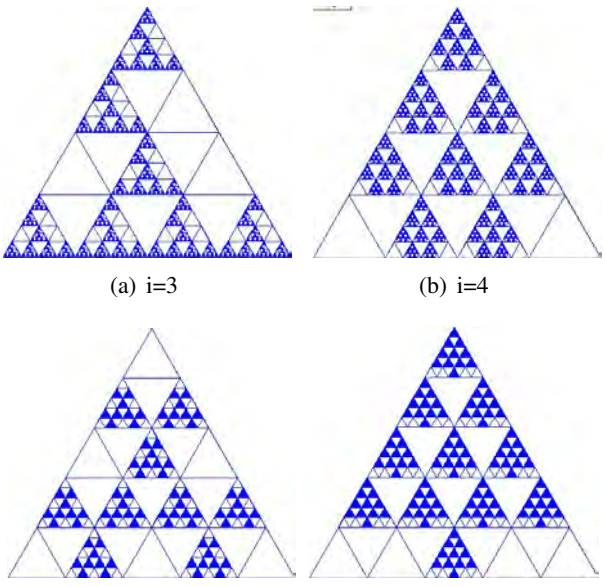
The geometric construction of the Sierpinski carpet-type fractal $E_{\mathcal{D}}$ mentioned in Introduction is the following: we divide a unit square into n^2 congruent squares, choose some squares according to the rule described by $\mathcal{D} \subset \{0, 1, \ldots, n - 1\}^2$, again divide each chosen squares into n^2 congruent ones, choose



(a) The first step (b) The second step

(c) The third step

Figure 2: The first three steps of constructing $E_{\mathcal{D}_2}$.



(a) i=3 (b) i=4

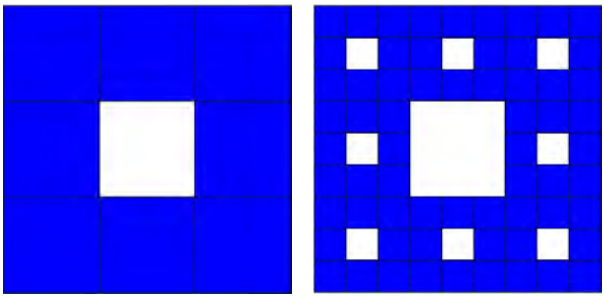
(c) i=5 (d) i=6

Figure 3: The third step of constructing $E_{\mathcal{D}_i}(i = 3, 4)$ and the second step of constructing $E_{\mathcal{D}_i}(i = 5, 6)$.

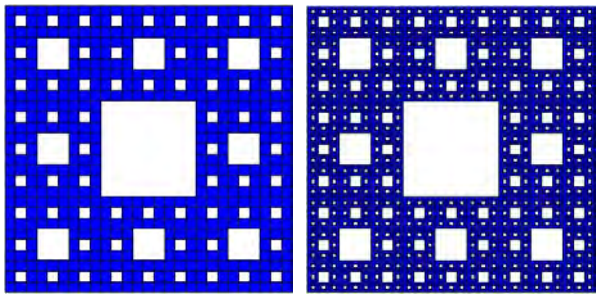
the square according to the same rule and repeat the procedure inductively to the infinity, we can obtain the set $E_{\mathcal{D}}$. For any $k \geq 1$, $E_{\mathcal{D}}^k$ here is the union of squares that are chosen in the step k. Firstly, we have the following program design by using the similar algorithm and procedure as in Section 3.

4.1 Program design (1)

function Sierpinskicarpet1(M, x, y, d, n, k)
% SIERPINSKICARPET1, Display the geometric construction process of Sierpinski carpet-type fractal.
% Call format: Sierpinskicarpet1(M, x, y, d, n, k).
% $M = [a_1, a_2, \ldots, a_m; b_1, b_2, \ldots, b_m]$, where $(a_i, b_i) \in \mathbb{R}^2$, $i = 1, \ldots, m$, are the lower-left coordinates of squares that chosen according to certain rule after the first division for the initial square.
% x is the $x-$coordinate of lower-left corner of initial square.
% y is the $y-$coordinate of lower-left corner of initial square.
% d is the side length of initial square.
% $1/n$ is contraction radio.
% k is iterated depth.
for $j = 1 : k$
$a1 = []$; $b1 = []$; % Two empty arrays, will be used to store the x and y coordinates of the lower-left corner of chosen squares after the kth step construction respectively.
for $i =$ 1:length(x)
$x1 = x(i) + d * M(1, :)$;
$y1 = y(i) + d * M(2, :)$;
squaregrid$(x(i), y(i), n, d)$ % Divide the square with lower-left coordinate $(x(i), y(i))$ and side d into n^2 congruent squares.
$a1 = [a1, x1]$; $b1 = [b1, y1]$;
end
$d = d/n$; $x = a1$; $y = b1$;
end
for $i =$ 1: length(x)
fill$(x(i) + [0, d, d, 0, 0], y(i) + [0, 0, d, d, 0],' b')$
% Fill the square with lower-left coordinate $(x(i), y(i))$ and side d by blue.
hold on
end
hold off, axis off, axis equal
% set(findobj(gcf,'type','patch'),'edgecolor','none')
function squaregrid(x, y, r, s)
$a = [x : s/r : x + d]$; $b = [y : s/r : y + d]$;
plot$(a$,meshgrid$(b, a),' b')$
hold on
plot(meshgrid$(a, b), b,' b')$
Saving the above text in a file called Sierpinskicarpet1.m in your current directory. For given $\mathcal{D} \subset$

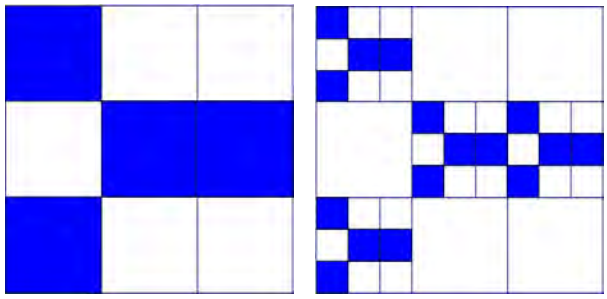


(a) The first step (b) The second step

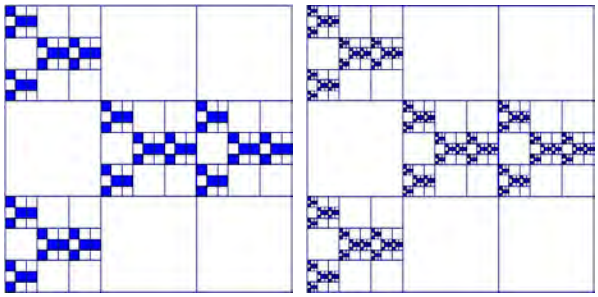


(c) The third step (d) The forth step

Figure 4: The first four steps of constructing Sierpinski carpet $E_{\mathcal{D}_7}$.



(a) The first step (b) The second step



(c) The third step (d) The forth step

Figure 5: The first four steps of constructing $E_{\mathcal{D}_8}$.

$\{0,1,\ldots,n-1\}^2$, let

$$M_{\mathcal{D}} = [a_1, a_2, \ldots, a_{\sharp\mathcal{D}}; b_1, b_2, \ldots, b_{\sharp\mathcal{D}}],$$

where $(na_i, nb_i) \in \mathcal{D}$ for each i. We can obtain the fine details of the kth step of constructing $E_{\mathcal{D}}$(Begin with an unit square $[0,1]^2$) by running the following command:

$$\text{Sierpinskitriangle}(M_{\mathcal{D}}, 0, 0, 1, n, k)$$

at the Matlab command window prompt.

Example 3. *Let* $n = 3$,

$$\mathcal{D}_7 = \{(0,0),(1,0),(2,0),(0,1),(2,1),$$

$$(0,2),(1,2),(2,2)\}.$$

We can obtain the details of the first four steps of constructing Sierpinski carpet $E_{\mathcal{D}_7}$ *by running continuous the following commands:*

$$\textit{Sierpinskicarpet1}(M_{\mathcal{D}_7}, 0, 0, 1, 3, k), k = 1, 2, 3, 4,$$

where $M_{\mathcal{D}_7} = [0, 1/3, 2/3, 0, 2/3, 0, 1/3, 2/3; 0, 0, 0, 1/3, 1/3, 2/3, 2/3, 2/3]$, *see Figure 4.*

Example 4. *Let* $n = 3$,

$$\mathcal{D}_8 = \{(0,0),(1,1),(2,1),(0,2)\}.$$

We can obtain the fine details of the first four steps of constructing Sierpinski carpet-type fractal $E_{\mathcal{D}_8}$ *by running continuous the following commands: Sierpinskitrigle*$([0, 1/3, 2/3, 0; 0, 1/3, 1/3, 2/3], 0, 0, 1, 2, k), k = 1, 2, 3, 4$, *see Figure 5.*

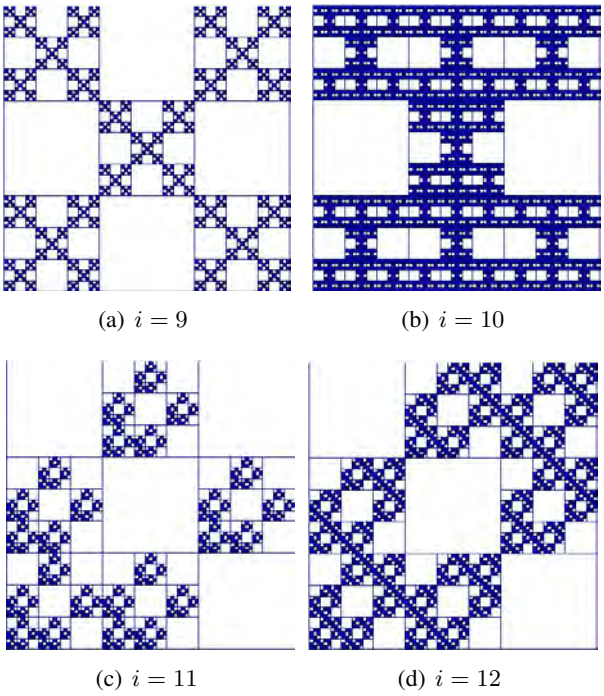
Take
$\mathcal{D}_9 = \{(0,0),(2,0),(1,1),(0,2),(2,2)\}$,
$\mathcal{D}_{10} = \{(0,0),(1,0),(2,0),(1,1),(0,2),(1,2),(2,2)\}$,
$\mathcal{D}_{11} = \{(0,0),(1,0),(1,1),(2,1),(1,2)\}$,
$\mathcal{D}_{12} = \{(0,0),(1,0),(1,1),(2,2),(1,2),(2,2)\}$
and run the following commands in turn:

$$\text{Sierpinskitrigle}(M_{\mathcal{D}_i}, 0, 0, 1, 3, 4)(i = 9, 10, 11, 12).$$

We can obtain the corresponding fine details as Figure 6.

For given rule $\mathcal{D} \subset \{0,1,\ldots,n-1\}^2$. Note that the square has better symmetry than equilateral triangle, so it is more easier to describe this rule $\mathcal{D}$ by a $n \times n$ matrix with element equal 0 or 1 for square. For example, for the above $\mathcal{D}_i (i = 1, 2)$, we denote the chosen squares according to the rule $\mathcal{D}_i (i = 1, 2)$ by 1, the others by 0. Then we can use the following two 3×3 matrixes to describe the rules $\mathcal{D}_1$ and $\mathcal{D}_1$ respectively:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}.$$



(a) $i = 9$ (b) $i = 10$

(c) $i = 11$ (d) $i = 12$

Figure 6: The fouth step of constructing $E_{\mathcal{D}_i}(i = 9, 10, 11, 12)$.

Thus we could simulate the geometric construction process of Sierpinski carpet-type fractal by using the other algorithm and program as follows, too.

4.2 Algorithm and procedure

The steps of creating the Sierpinski-carpet fractals as following:

①Suppose t is the lower-left coordinate of a square with side d.

②Fill the initial square by blue, and divide it into n^2 congruent squares. We denote the chosen squares according to the certain rule by 1, the others by 0, and store these information by a $n \times n$ matrix.

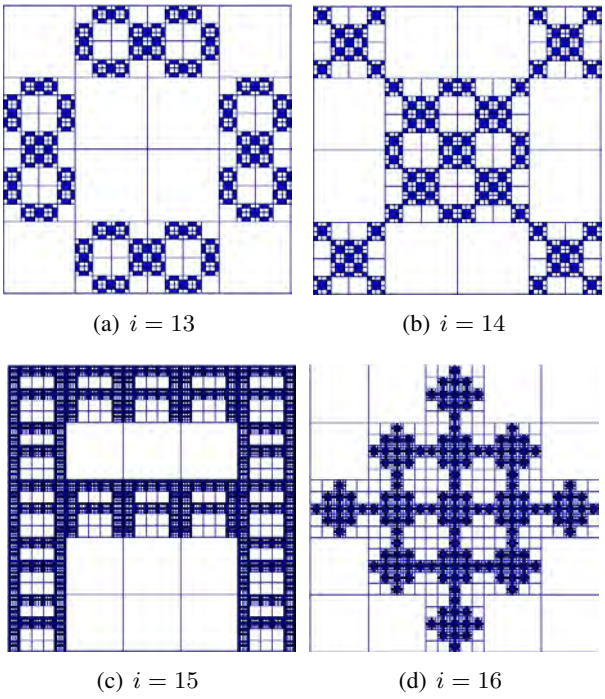
③Suppose the iterated depth is k. Traversal the matrix in step 2. If encounter 0, fill the corresponding place with white. If encounter 1, assign the lower-left coordinate and side length of corresponding small square to the variables of lower-left coordinate and side length of initial square. While $k = 0$, end; while $k > 0$, execute the step 2.

4.3 Program and design

function Sierpinskicarpet2(M, t, d, k)
% SIERPINSKICARPET2, Display the geometric construction process of Sierpinski carpet-type fractal.
% Call format: Sierpinskicarpet2(M, t, d, k).
% $M = [a_{11}, a_{12}, \ldots, a_{1n}; \ldots; a_{n1}, a_{n2}, \ldots, a_{nn}]$, $a_{ij} = 0$ or 1, $1 \le i \le n, 1 \le j \le n$. If we type it at the Matlab command window prompt, we can see a $n \times n$ matrix with elements equal 0 or 1 in screen.
% k is iterated depth.
% $t = [t(1), t(2)]$ is lower-left coordinate of initial square.
% d is side length of initial square.
if $k == 0$
return
end
m =length$(M(1,:))$;
drawgrid(t, d, m) % Traversal each element in M, fill the corresponding square with lower-left coordinate t and side d by white if encounter 0.
for $i = 1 : m$
for $j = 1 : m$
if$(M(i, j) == 0)$
square$([t(1) + d * (j - 1)/m, t(2) + d * (m - i)/m], d/m)$
else
fillsquare$(M, [t(1) + d * (j - 1)/m, t(2) + d * (m - i)/m], d/m, k - 1)$
end
end
end
function square(t, d) % Fill the square with lower-left coordinate t and side d by white.
fill$([t(1), t(1) + d, t(1) + d, t(1)], [t(2), t(2), t(2) + d, t(2) + d], 'k')$;
hold on
function drawgrid(t, d, m) % Fill the square with lower-left coordinate t and side d by blue and divide it into n^2 congruent squares.
fill([t(1),t(1)+d,t(1)+d,t(1)],[t(2),t(2),t(2)+d,t(2)+d],'b')
hold on
$x = [t(1) : d/m : t(1) + d]$; $y = [t(2) : d/m : t(2) + d]$;
plot(x,meshgrid$(y, x), 'b')$
hold on
plot(meshgrid$(x, y), y, 'b')$
axis off, axis equal

Saving the above text in a file called Sierpinskicarpet2.m in your current directory. As previously $\mathcal{D} \subset \{0, 1, \ldots, n-1\}^2$ and $E_{\mathcal{D}}$. Let $M_{\mathcal{D}}$ denote the $n \times n$ matrix with elements equal 0 or 1 corresponding with the rule $\mathcal{D}$. We can obtain the fine details of the kth step of constructing $E_{\mathcal{D}}$ by running the following command: Sierpinskitriangle2$([a_{11}, a_{12}, \ldots, a_{1n}; \ldots; a_{n1}, a_{n2}, \ldots, a_{nn}], [0, 0], 1, k)$ at the Matlab command window prompt, where a_{ij} is the element of matrix $M_{\mathcal{D}}$ in row i of column j. Set $M'_{\mathcal{D}} = [a_{11}, a_{12}, \ldots, a_{1n}; \ldots; a_{n1}, a_{n2}, \ldots, a_{nn}]$.

Example 5. *Let*



(a) $i = 13$ (b) $i = 14$

(c) $i = 15$ (d) $i = 16$

Figure 7: The third step of constructing $E_{\mathcal{D}_i}(i = 13, 14, 15, 16)$.

$\mathcal{D}_{13} = \{(1,0),(2,0),(0,1),(3,1),(0,2),(3,2),(1,3),(2,3)\}$,

$\mathcal{D}_{14} = \{(0,0),(3,0),(1,1),(2,1),(1,2),(2,2),(0,3),(3,3)\}$,

$\mathcal{D}_{15} = \{(0,0),(4,0),(0,1),(4,1),(0,2),(1,2),(2,2),(3,2),(4,2),(0,3),(4,3),(0,4),(1,4),(2,4),(3,4),(4,4)\}$,

$\mathcal{D}_{16} = \{(2,0),(1,1),(2,1),(3,1),(0,2),(1,2),(2,2),(3,2),(4,2),(1,3),(2,3),(3,3),(2,4)$.

Then

$$M_{\mathcal{D}_{13}} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$$M_{\mathcal{D}_{14}} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{\mathcal{D}_{15}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{\mathcal{D}_{16}} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

We can obtain the fine details of the third step of constructing Sierpinski carpet-type fractals $E_{\mathcal{D}_i}(i = 13, 14, 15, 16)$ by running the following commands:

Sierpinskicarpet2$(M'_{\mathcal{D}_i}, [0,0], 1, 3), (i = 13, 14, 15, 16)$,

see Figure 7.

Remark 1. *For any $k \geq 1$, according to the actual needs, we also can obtain the pattern of $E^k_{\mathcal{D}}$ by removing the part of division in programming.*

Acknowledgements: The authors cordially thank the editors and referees for their careful reading and helpful comments. The research was supported by Basic Scientific Research Project of Northwest A&F University (No. Z109021203) and Doctoral Scientific Research foundation of Northwest A&F University, (No. Z109021111).

References:

[1] M.-F. Barnsley et al, *The science of fractal images,* Springer–Verlag, New York, 1988.

[2] D. David and S. Semmes, *Fractured Fractals and Broken Dreams: Self-Similar Geometry Through Metric and Measure,* Oxford Univ.Press, 1997.

[3] F. Deng and F.-L. Xi, An Application of L-system and IFS in 3D Fractal Simulation, *WSEAS Transactions on Systems.* 4, 2008, pp. 352–361.

[4] K.-J. Falconer, *Fractal GeometryłMathematical Foundations and Applications,* Chichester: Wiley, 1990.

[5] M. Finlay and K.-A. Blanton, *Real-World Fractals: Object-Oriented Fractal Programming in C++,* M & T Press, 1994.

[6] Y.-X. Gui and W.-X. Li, A generalized multifractal spectrum of the general sierpinsk carpets, *J. Math. Anal. Appl.* 348, 2008, pp. 180–192.

[7] J.-E. Hutchinson, Fractals and self-similarity, *Indiana. Univ. Math. J.* 30, 1981, pp. 713–749.

[8] B.-G. Jia, Bounds of Hausdorff measure of the Sierpinski gasket, *J. Math. Anal. Appl.* 330, 2007, pp. 1016-1024.

[9] K.-S. Lau, J.-J. Luo and H. Rao, Topological structure of fractal squares, *Proceedings of the Edinburgh Mathematical Society.* 155(1), 2013, pp. 73–86.

[10] Q.-H. Liu, L.-F. Xi and Y-.F. Zhao, Dimensions of intersections of the Sierpinski carpet with lines of rational slopes, *Proceedings of the Edinburgh Mathematical Society.* 50, 2007, pp. 411-4428.

[11] N. Lu, *Fractal imaging,* Morgan Kaufmann Publishers, 1997.

[12] Mathworks, *Matlab:The language of technical computing,* version 6.5, 2002.

[13] C. McMullen, The Hausdorff dimension of general Sierpinski carpets, *Nagoya Math. J.* 96, 1984, pp. 1–9.

[14] T.-D. Taylor, Connectivity properties of Sierpinski relatives, *Fractals.* 19(4), 2011, pp. 481–506.

[15] Z.-X. Wen, Z.-Y. Zhu and G.-T. Deng, Lipschitz equivalence of a class of general Sierpinski carpets, *J. Math. Anal. Appl.* 385, 2012, pp. 16–23.

[16] L.-F. Xi and Y. Xiong, Self-similar sets with initial cubic patterns, *C. R. Acad. Sci. Paris. Ser. I.* 348(1), 2010, pp. 15–20.

[17] W.-Q. Zeng and X.-Y. Wang, *Fractal theory and its computer simulation,* Northeastern university press, Shen yang, China, 2001.

[18] Z.-L. Zhou and M. Wu, The hausdorff measure of a Sierpinski carpet, *Sci. China. Ser. A.* 43(7), 1999, pp. 673-680.

[19] Z.–L. Zhou and F. Li, A new estimate of the Hausdorff measure of the Sierpinski gasket, *Nonlinearity.* 13(3), 2000, pp. 479–491.

[20] Z.–L. Zhou, Hausdorff measure of Sierpinski gasket, *Sci. China. Ser. A.* 40(10), 1997, pp. 1016-1021.

[21] Z.-Y. Zhu, Y. Xiong and L.-F. Xi, Lipschitz equivalence of self-similar sets with triangular pattern, *Sci. China. Ser. A.* 54(5), 2011, pp. 1019–1026.