Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 9(5): 13346–13357.
DOI:10.3934/math.2024651
Received: 13 February 2024
Revised: 01 April 2024
Accepted: 07 April 2024
Published: 11 April 2024

Research article

Generating irregular fractals based on iterated function systems

Baoxing Zhang[1,*], **Yunkun Zhang**[1] **and Yuanyuan Xie**[2]

[1] College of Statistics and Mathematics, Hebei University of Economics and Business, Shijiazhuang 050061, China

[2] Institute of Science and Technology, Tianjin University of Finance and Economics, Tianjin 300222, China

* **Correspondence:** Email: baoxingzhang@hueb.edu.cn.

Abstract: The iterated function system (IFS) is important in different fields like image compression. An important feature of such systems is that they can be used to generate fractals. Yet, for the obtained fractals, it is difficult to locally control them to generate new ones with desired structures at specific places. In this paper, we gave an attempt to solve this problem based on a nonuniform multiple function system. For this, we first analyzed the multiple function systems needed in the generation of the final desired fractals. Based on such analysis, the final fractals with desired structures at specific places can be generated using the nonuniform multiple function system. Moreover, these two procedures were summarized into two algorithms for convenience. Examples were also given to illustrate the performance of the nonuniform multiple function system and the two algorithms in this paper.

Keywords: iterated function system; local control; fractal; subdivision; multiple function system
Mathematics Subject Classification: 28A80, 47J26

1. Introduction

Since Mendelbrot proposed the notion of fractals in the 1970s [1], the research about different properties and applications of fractals has been attracting scholars' attention. Two geometric properties characterizing fractals are that they are self-similar and they are attractors. Therefore, the iterated function system (IFS) is a standard framework to generate them.

In connection with the study on IFS, there have been interesting works after its appearance [2]. For example, Barnsley et al. [3, 4] investigated the super IFS and recurrent IFS. Fisher et al. [5] discussed the partitioned IFS. On the other hand, subdivision is an efficient tool to generate smooth curves/surfaces and plays an important role in different applications [6]. For the connection between these two different topics, Schaefer et al. [7] showed that stationary subdivision curves/surfaces are fractals and that many standard fractals, like the Koch curve, can be obtained using subdivision rules. Later, Levin et al. [8] generalized this connection to give the connection between nonstationary subdivision and IFS. For other similar works, refer to [9, 10] and the references therein.

However, none of the above works discussed how to generate fractals with desired structures at specific places. Such a problem appeared in [11] in an example showing a fractal with different structures at different locations. In fact, such fractals are irregular ones (see also [9]) and can better reflect the nature. Therefore, in this paper, we give a first attempt to solve this problem and generate fractals with desired structures at specific places. The fractals in this paper are obtained by first analyzing two kinds of multiple function systems [12] and then are generated using a kind of nonuniform multiple function system. Related works can be found in [13] and the references therein. The nonuniform multiple function system in this paper chooses different IFSs for different sets of generated points through iterations (see Section 3 for details). These corresponding procedures are summarized into two algorithms for convenience. We point out that, with the nonuniform multiple function system and the two algorithms, we can indeed generate fractals with desired structures at specific places, which can enrich the theory and applications of fractals. Examples are also given to show the performance of the new multiple function system and the two algorithms.

The rest of this paper is organized as follows. In Section 2, we give a brief review of the related works together with an overview of the method to generate the final desired fractals. Section 3 is devoted to present the new nonuniform multiple function system with the analysis of convergence. In Section 4, we discuss how to use the new nonuniform multiple function systems to generate fractals with desired structures at specific places and summarize the procedures into two corresponding algorithms. In Section 5, we give examples to illustrate the performance of the new nonuniform multiple function system and algorithms, while Section 6 concludes this paper.

2. Some preliminaries

In this section, we briefly review some related works and give an overview of the method to generate the final desired fractals.

2.1. Related works

Let (X, d) be a complete metric space and the function $f : X \to X$ be a contractive one, which means that the Lipschitz constant is

$$Lip(f) := \sup_{x,y \in X, x \neq y} \frac{d(f(x), f(y))}{d(x, y)} < 1.$$

An IFS is a collection of contractive functions $\mathcal{F} = \{f_1, \cdots, f_s\}$. A set X is an attractor of $\mathcal{F}$ if $\mathcal{F}(X) = X$, where $\mathcal{F}(X) = f_1(X) \bigcup \cdots \bigcup f_s(X)$. Using IFS, the fractals can be obtained as the corresponding attractors.

On the other hand, subdivision schemes can generate smooth curves/surfaces [6] and play an important role in different fields like wavelets [14] and computer-aided geometric design [15]. After Schaefer et al. [7] gave the connection between stationary subdivision and IFS, Dyn et al. [11] discussed the connection between IFS and a wider class of subdivision, like the nonuniform subdivision, and also gave the tree of maps. Similar works can be seen in [12], which generalized the trees of maps to multiple function systems and can generate different kinds of fractals.

In fact, similar to the trees of maps in [11], the multiple function systems can also be arranged in a tree structure. Let $B^{[k]} := \{\boldsymbol{\epsilon}_k : \boldsymbol{\epsilon}_k = (\epsilon_1\epsilon_2\cdots\epsilon_k), \epsilon_j \in \{1,2\}\} = \{1,2\}^k$ and $B^{[\infty]} := \{\boldsymbol{\epsilon} : \boldsymbol{\epsilon} = (\epsilon_1\epsilon_2\cdots\epsilon_k\cdots), \epsilon_j \in \{1,2\}\} = \{1,2\}^\infty$. Let τ_l be the truncation operator defined as $\tau_k\boldsymbol{\epsilon} = \boldsymbol{\epsilon}_k = (\epsilon_1\cdots\epsilon_k) \in B^{[l]}$. With such notations, we give the multiple function system $\mathcal{F}_{\boldsymbol{\epsilon}} = \{\mathcal{F}_0, \mathcal{F}_1\}$ as an example to show the corresponding tree structure as in Figure 1. Here, each $\boldsymbol{\epsilon} \in B^{[\infty]}$ defines a path in the tree structure as in Figure 1.

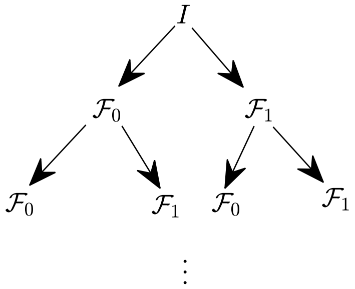

Figure 1. The tree structure of the multiple function system $\mathcal{F}_{\boldsymbol{\epsilon}}$.

Remark 1. *Here, for each $\boldsymbol{\epsilon}_k = (\epsilon_1, \cdots, \epsilon_k) \in B^{[k]}$, we have the function $f_{\boldsymbol{\epsilon}_k}(\cdot) = f_{\epsilon_k} \circ \cdots \circ f_{\epsilon_1}(\cdot)$.*

2.2. The method overview

We first point out that the contractive functions in this paper are affine transformations. The fractals with desired structures at specific places in this paper are generated using the nonuniform multiple function system.

In fact, the generation of such fractals starts from two kinds of multiple function systems: the one generating the fractal with the specific place where we hope the desired structures to locate and the one generating the fractals with the desired structures. For simplicity, we denote the fractal with the specific place by the target fractal. In fact, the ‘specific places’ in this paper are polygons formed with points generated by different rounds of iterations along the paths of multiple function systems, then the final fractals with desired structures at specific places means that specific polygons in the target fractal have the desired structures.

To generate such fractals, some necessary items are needed, like the paths leading to the specific polygons in the target fractal and the paths leading to the desired structures. This can be done by analyzing the above two kinds of multiple function systems (see Section 4 for details). Once the necessary items are obtained, the final desired fractal can be generated by locally controlling the structure of the specific polygons in the target fractal using the paths leading to the desired structures and the nonuniform multiple function systems (see Section 4 for details).

We also summarize the above procedures into two algorithms to obtain the final desired fractals (see Section 4 for details) for convenience. We point out that with such a nonuniform multiple function system and the given two algorithms, we can indeed locally control the structures of fractals to generate new ones with desired structures at specific places.

3. The nonuniform multiple function system

This section is devoted to presenting the new nonuniform multiple function system and analyzing its convergence. For this, we first review the connection between subdivision and IFS.

3.1. Subdivision and IFS

In fact, according to Schaefer et al. [7], based on subdivision rules, we can give a contractive function in the form

$$f(A) = AP^{-1}SP, \quad A \in Q^{n-1},$$

where S is the corresponding $n \times n$ subdivision matrix, Q^{n-1} is the $n-1$ dimensional hyperplane with points of the form $(x_1, \cdots, x_{n-1}, 1)$, and P is the $n \times n$ matrix composed of n initial points in $\mathbb{R}^m$, with the last column $\mathbf{1}_n$ being the $n \times 1$ vector of 1, i.e., $(1, \cdots, 1)^\top$, and the rest of the columns such that P is invertible [7]. Schaefer et al. [7] showed that such a contractive function can be rewritten as

$$f(A) = A\begin{pmatrix} G & \mathbf{0}_{n-1} \\ \boldsymbol{v} & 1 \end{pmatrix},$$

where G is a $(n-1) \times (n-1)$ matrix, $\boldsymbol{v}$ is a $1 \times (n-1)$ vector, and $\mathbf{0}_{n-1}$ is the $(n-1) \times 1$ vector of 0, $(0, \cdots, 0)^\top$.

Conversely, Schaefer et al. [7] also gave a general way to convert fractals into subdivision schemes. Specifically speaking, given an IFS defined by a set of affine transformations $\mathcal{F} = \{f_1, \cdots, f_n\}$, we first choose a set of control points $P = \{p_1, \cdots, p_m\}$, then apply $\mathcal{F}$ to the set P to get the corresponding set of points $f_i(P)$. For the vertex p_j in $f_i(P)$, we represent it as

$$p_j = \sum_k \alpha_{j,k} p_k.$$

The subdivision matrix S_i can be constructed by setting the (j, k) entry of S_i to be $\alpha_{j,k}$, and the subdivision matrix S for this fractal is the concatenation of S_i.

In this way, from the IFS $\mathcal{F} = \{f_1, \cdots, f_n\}$, we have the new IFS $\tilde{\mathcal{F}} = \{\tilde{f}_1, \cdots, \tilde{f}_n\}$, with

$$\tilde{f}_i(A) = AP^{-1}S_iP, \quad A \in Q^{n-1}. \tag{3.1}$$

Remark 2. *With the above connection between stationary subdivision and IFS, all the contractive functions in this paper are of the form (3.1).*

Now, we give an observation of the IFS consisting of the contractive functions of the form (3.1). In fact, for some contractive function of the form (3.1), when the matrix P is changed to be P', this function is also changed and we get a modified IFS. Here, P' is also an invertible $n \times n$ matrix with the last column $\mathbf{1}_n$. Yet, since the final fractal is independent of the initial set [7] and the subdivision matrices are kept the same, the fractal obtained using this modified IFS is a scaled one with the same structure as the original one. Additionally, this obtained fractal is limited to the polygon formed with the points constituting the first m columns of P'. We summarize this point as follows:

Proposition 1. *For the IFS with the contractive functions of the form (3.1), if the $n \times n$ matrix P is replaced by another invertible $n \times n$ matrix P' with the last column $\mathbf{1}_n$ while the subdivision matrices S_i are kept the same, the corresponding fractal is a scaled one with the same structure as the original fractal.*

3.2. The nonuniform multiple function system

Now, we derive the nonuniform multiple function system. Note that in the process of generating fractals through iterations, both the trees of maps in [11] and the multiple function systems in [12] choose some certain contractive function or IFS for all the generated points, as shown in Figure 2 (right) for the multiple function system. Based on this point, now we present the new nonuniform multiple function system.

Let $\tilde{\mathcal{F}}_{\boldsymbol{\epsilon}} = \{\tilde{\mathcal{F}}_1, \tilde{\mathcal{F}}_2\}$ with $\tilde{\mathcal{F}}_1 = \{\tilde{f}_{11}, \cdots, \tilde{f}_{1s_1}\}$, $\tilde{\mathcal{F}}_2 = \{\tilde{f}_{21}, \cdots, \tilde{f}_{2s_2}\}$ denote a multiple function system, where each contractive function $\tilde{f}_{ij}$ ($j = 1, \cdots, s_i$ and $i = 1, 2$) are of the form (3.1). Let $\bigcirc := \{p_0, \cdots, p_n\}$ be a set of n initial points. For the first step iteration of the nonuniform multiple function system, we choose $\tilde{\mathcal{F}}_{\epsilon_1} \in \tilde{\mathcal{F}}_{\boldsymbol{\epsilon}}$ with $\epsilon_1 \in \{1, 2\}$ and generate s_{ϵ_1} sets consisting of n points, each starting from $\bigcirc$. Now, we modify the functions in $\tilde{\mathcal{F}}_1$ and $\tilde{\mathcal{F}}_2$ by replacing the corresponding matrix P with a new one, P', according to Subsection 3.1. The new IFSs are denoted by $\mathcal{F}_1$ and $\mathcal{F}_2$, with the corresponding multiple function system $\mathcal{F}_{\boldsymbol{\epsilon}}$. For the next step of iteration, we choose $\mathcal{F}_{\epsilon_2,1} \in \mathcal{F}_{\boldsymbol{\epsilon}}$ for the first set and use $\mathcal{F}_{\epsilon_{2,2}} \in \mathcal{F}_{\boldsymbol{\epsilon}}$ for the second set, until $\mathcal{F}_{\epsilon_{2,s_{\epsilon_1}}} \in \mathcal{F}_{\boldsymbol{\epsilon}}$ for the last set. Suppose after k steps of iteration, we obtain m sets consisting of n points each. For the next step of iteration, from the modified multiple function system, which is still denoted by $\mathcal{F}_{\boldsymbol{\epsilon}}$, we choose $\mathcal{F}_{\epsilon_{k+1,1}} \in \mathcal{F}_{\boldsymbol{\epsilon}}$ for the first set and use $\mathcal{F}_{\epsilon_{k+1,2}} \in \mathcal{F}_{\boldsymbol{\epsilon}}$ for the next set, until $\mathcal{F}_{\epsilon_{k+1,m}} \in \mathcal{F}_{\boldsymbol{\epsilon}}$ for the last set. The above process is shown in a tree structure shown in Figure 2 (left).

From Figure 2, it can be seen that along a path of a multiple function system, if we replace $\tilde{\mathcal{F}}_{\epsilon_j}$ in the jth time of iteration by different modified ones for different sets, we get the nonuniform multiple function system. In this way, 'nonuniform' here implies that the choice of IFS in each time of iteration depends on the generated points.

In addition, for the multiple function system shown in Figure 2 (right), the path is $(\epsilon_1, \cdots, \epsilon_k, \cdots)$. Since the nonuniform multiple function system can be seen as obtained by replacing the chosen IFS of a multiple function system with different modified ones in each time of iteration, in this paper, we still denote one path of the nonuniform multiple function system by $(\epsilon_1, \cdots, \epsilon_k, \cdots)$. In fact, the path $(\epsilon_1, \cdots, \epsilon_k, \cdots)$ here can actually be seen as the union of the ones $(\epsilon_1, \epsilon_{2,i_1}, \cdots, \epsilon_{k,i_k}, \cdots)$ shown in Figure 2 (left).

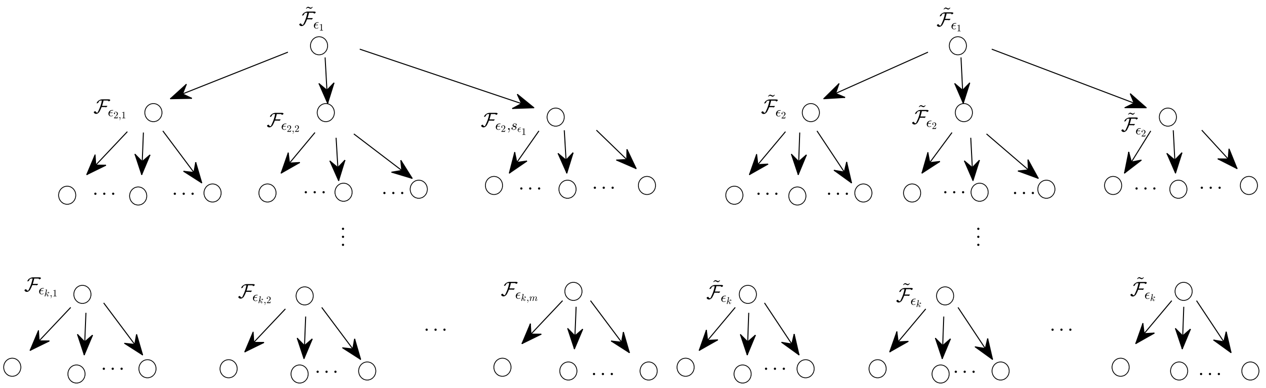
$\tilde{\mathcal{F}}_{\epsilon_1}$
$\mathcal{F}_{\epsilon_{2,1}}$
$\mathcal{F}_{\epsilon_{2,2}}$
$\mathcal{F}_{\epsilon_{2,s_{\epsilon_1}}}$
$\mathcal{F}_{\epsilon_{k,1}}$
$\mathcal{F}_{\epsilon_{k,2}}$
$\mathcal{F}_{\epsilon_{k,m}}$
$\tilde{\mathcal{F}}_{\epsilon_1}$
$\tilde{\mathcal{F}}_{\epsilon_2}$
$\tilde{\mathcal{F}}_{\epsilon_2}$
$\tilde{\mathcal{F}}_{\epsilon_2}$
$\tilde{\mathcal{F}}_{\epsilon_k}$
$\tilde{\mathcal{F}}_{\epsilon_k}$
$\tilde{\mathcal{F}}_{\epsilon_k}$

Figure 2. The tree structure along a path of a nonuniform multiple function system (left) and a multiple function system (right).

3.3. Convergence

Now, we present the convergence of the nonuniform multiple function systems along a path. In fact, similar to Proposition 2.3 in [11], we have the following result.

Theorem 1. *For the path $(\epsilon_1, \epsilon_{2,i_1}, \cdots, \epsilon_{k,i_k}, \cdots)$ in the tree structure of a nonuniform multiple function system with convergent IFSs, the iterations along this path starting from an initial set converges to a unique limit.*

In this way, similar to Proposition 2.4 in [11], the final fractal along a certain path $(\epsilon_1, \cdots, \epsilon_k, \cdots)$ of a nonuniform multiple function system can be obtained as the union of all the limits obtained along the paths $(\epsilon_1, \epsilon_{2,i_1}, \cdots, \epsilon_{k,i_k}, \cdots)$ constituting $(\epsilon_1, \cdots, \epsilon_k, \cdots)$, which is stated as follows:

Theorem 2. *For a nonuniform multiple function system, the attractor along the path $(\epsilon_1, \cdots, \epsilon_k, \cdots)$, which is composed of $(\epsilon_1, \epsilon_{2,i_1}, \cdots, \epsilon_{k,i_k}, \cdots)$, is actually the union of the limits along $(\epsilon_1, \epsilon_{2,i_1}, \cdots, \epsilon_{k,i_k}, \cdots)$.*

4. Generation of fractals with desired structures at specific places

Based on Section 3, we now show how to use the nonuniform multiple function system to generate fractals with desired structures at specific places.

To this purpose, we first denote the multiple function systems generating the target fractal and the fractals containing the desired structures by $\mathcal{F}_1 = \{\mathcal{F}_{11}, \cdots, \mathcal{F}_{1n_1}\}$ and $\mathcal{F}_2 = \{\mathcal{F}_{21}, \cdots, \mathcal{F}_{2n_2}\}$, respectively. Here, all the contractive functions in the IFS $\mathcal{F}_{ij}$ with $i = 1, 2$ and $j = 1, \cdots, n_i$ are of the form (3.1) (see Remark 2). The contractive functions in $\mathcal{F}_{1j}$ with $j = 1, \cdots, n_1$ are from $\mathcal{F} = \{f_1, \cdots, f_N\}$, which consists of all the contractive functions based on the relevant subdivision matrices (see Subsection 3.1).

The whole process of generating the final desired fractals starts from analyzing $\mathcal{F}_1$ and $\mathcal{F}_2$ to derive the following items:
1) for $\mathcal{F}_1$, the paths leading to the specific polygons in the target fractal;
2) for $\mathcal{F}_2$, the paths leading to the desired structures.
In fact, for the multiple function system $\mathcal{F}_1$, try different paths to generate the target fractal and record the corresponding path. For the multiple function system $\mathcal{F}' = \{\mathcal{F}'_i\}$ with $\mathcal{F}'_i = \{f_i\}, i = 1, \cdots, N$, by comparing the results generated along different paths of $\mathcal{F}'_i$ and the target fractal, get the specific polygons and the paths leading to them. For the multiple function system $\mathcal{F}_2$, try different paths in order to generate the fractals containing the desired structures and record the corresponding paths. This helps us to know along which paths the desired structures can be obtained. We summarize the above analysis into Algorithm 1 as follows:

Algorithm 1 Analysis of the multiple function systems.

```
Step 0: Get the multiple function systems generating the target fractal and the fractals containing the
    desired structures, i.e., $\mathcal{F}_1$ and $\mathcal{F}_2$;
Step 1: For $\mathcal{F}_1$, generate the target fractal and record the path;
Step 2: For $\mathcal{F}'$, get the specific polygons in the target fractal and the paths leading to them by
    comparing the results generated along the paths of $\mathcal{F}'$ and the target fractal;
Step 3: For $\mathcal{F}_2$, generate the fractals with the desired structures and record the paths.
```

With all the needed items gotten by Algorithm 1, we can now generate the final desired fractal. For this, we first adjust each specific polygon so that we can use the points of each adjusted polygon to construct a new matrix P' with the same order as P in the contractive functions in $\mathcal{F}_{2j}$, $j = 1, \cdots, n_2$. We then modify these contractive functions by replacing the matrix P with P' and get the modified multiple function system $\tilde{\mathcal{F}_2}$ corresponding to each adjusted specific polygon. Starting from each adjusted specific polygon, we then take iterations along the path leading to the desired structure using the modified multiple function system corresponding to this adjusted polygon and get a scaled fractal with the desired structure according to Proposition 1. By combining all the obtained fractals and the target fractal, we get the final desired fractal. For this procedure, we also summarize it into Algorithm 2 as follows:

Algorithm 2 Generation of the desired fractal.

```
Step 0: Adjust each specific polygon according to the contractive functions in $\mathcal{F}_{2j}$ with $j$=1, $\cdots$, $n_2$, and get the new adjusted specific polygons;
Step 1: Take the control points of each adjusted polygon in the target fractal;
Step 2: With the points of each adjusted polygon, construct a matrix $P'$ to replace $P$ in the contractive functions of $\mathcal{F}_{2j}$ with $j = 1, \cdots, n_2$ in order to get the modified one, $\tilde{\mathcal{F}_2}$, corresponding to this adjusted polygon;
Step 3: Starting from each adjusted polygon from step 0, take iterations along the path leading to the desired structures using the modified multiple function system corresponding to this adjusted polygon, and get the corresponding scaled fractals;
Step 4: Get the final desired fractal by combining all the scaled fractals and the target fractal.
```

Remark 3. *If we let P_1 be the matrix in the contractive functions of the multiple function system generating the target fractal and P_2 be the matrix in the contractive functions of the multiple function systems generating the fractals with desired structures, the adjustment of each specific polygon means that the order of P_1 is not less than that of P_2.*

5. Examples

Now, we use the algorithms in Section 4 to generate fractals with desired structures at specific places.

Example 1. For the Sierpinski Carpets, some of them are based on quadrilaterals and the others are based on triangles. Now, we try to generate a new interesting 'Sierpinski Carpet-like' fractal by suitably combining them.

Zhang et al. [12] gave an example showing 'Sierpinski Carpet-like' fractals based on quadrilaterals. Here, we use the points $(0, 0), (1, 0), (0, 1), (1, 1)$ to derive the matrix P as

$$P = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

The multiple function system is $\mathcal{F}^1_\epsilon = \{\mathcal{F}^1_0, \mathcal{F}^1_1\}$ with $\mathcal{F}^1_0 = \{f^1_1, \cdots, f^1_9\}$ and $\mathcal{F}^1_1 =$

$\{f_1^1, \cdots, f_4^1, f_6^1, \cdots, f_9^1\}$, where $f_i^1(A) = AP^{-1}S_iP$, $i = 1, \cdots, 9$ and the corresponding subdivision matrices S_i are as follows:

$$S_1 = 1/9\begin{pmatrix} 9 & 0 & 0 & 0 \\ 6 & 3 & 0 & 0 \\ 6 & 0 & 3 & 0 \\ 4 & 2 & 2 & 1 \end{pmatrix}, S_2 = 1/9\begin{pmatrix} 6 & 3 & 0 & 0 \\ 3 & 6 & 0 & 0 \\ 4 & 2 & 2 & 1 \\ 2 & 4 & 1 & 2 \end{pmatrix}, S_3 = 1/9\begin{pmatrix} 3 & 6 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ 2 & 4 & 1 & 2 \\ 0 & 6 & 0 & 3 \end{pmatrix},$$

$$S_4 = 1/9\begin{pmatrix} 6 & 0 & 3 & 0 \\ 4 & 2 & 2 & 1 \\ 3 & 0 & 6 & 0 \\ 2 & 1 & 4 & 2 \end{pmatrix}, S_5 = 1/9\begin{pmatrix} 4 & 2 & 2 & 1 \\ 2 & 4 & 1 & 2 \\ 2 & 1 & 4 & 2 \\ 1 & 2 & 2 & 4 \end{pmatrix}, S_6 = 1/9\begin{pmatrix} 2 & 4 & 1 & 2 \\ 0 & 6 & 0 & 3 \\ 1 & 2 & 2 & 4 \\ 0 & 3 & 0 & 6 \end{pmatrix},$$

$$S_7 = 1/9\begin{pmatrix} 3 & 0 & 6 & 0 \\ 2 & 1 & 4 & 2 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 6 & 3 \end{pmatrix}, S_8 = 1/9\begin{pmatrix} 2 & 1 & 4 & 2 \\ 1 & 2 & 2 & 4 \\ 0 & 0 & 6 & 3 \\ 0 & 0 & 3 & 6 \end{pmatrix}, S_9 = 1/9\begin{pmatrix} 1 & 2 & 2 & 4 \\ 0 & 3 & 0 & 6 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 0 & 9 \end{pmatrix}.$$

Let the initial set be $\bigcirc_1 = \{(0,0), (1,0), (0,1), (1,1)\}$. Along different paths in the tree structure of the above multiple function system, different 'Sierpinski Carpet-like' fractals can be obtained, like the ones shown in Figure 3.

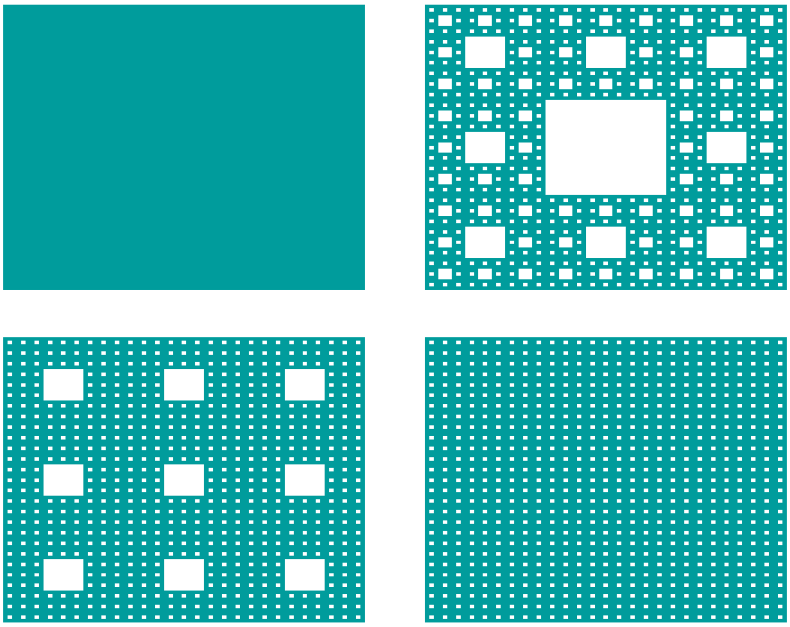

Figure 3. 'Sierpinski Carpet-like' fractals generated by the multiple function system $\mathcal{F}_\epsilon^1$ with $\epsilon = (0,0,0,0), (1,1,1,1), (1,0,1,0), (1,0,0,0)$ (top to bottom and left to right).

Another multiple function system generating the 'Sierpinski Carpet-like' fractals are based on triangles. The corresponding multiple function system here is $\mathcal{F}_\epsilon^2 = \{\mathcal{F}_0^2, \mathcal{F}_1^2\}$ with $\mathcal{F}_0^2 = \{f_1^2, f_2^2, f_3^2, f_4^2\}$ and $\mathcal{F}_1^2 = \{f_1^2, f_3^2, f_4^2\}$, where $f_i^2(X) = XP^{-1}S_iP$ with

$$S_1 = 1/2\begin{pmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}, S_2 = 1/2\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}, S_3 = 1/2\begin{pmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{pmatrix}, S_4 = 1/2\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix},$$

and the matrix P formed using the points $(0,0),(1,0),(1/2,2/3)$ is as follows:

$$P = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1/2 & 2/3 & 1 \end{pmatrix}.$$

Starting from the initial set $\bigcirc_2 = \{(0,0),(1,0),(1/2,2/3)\}$, we can generate the corresponding Sierpinski Carpet using the multiple function system $\mathcal{F}_\epsilon^2$, as shown in Figure 4 (left).

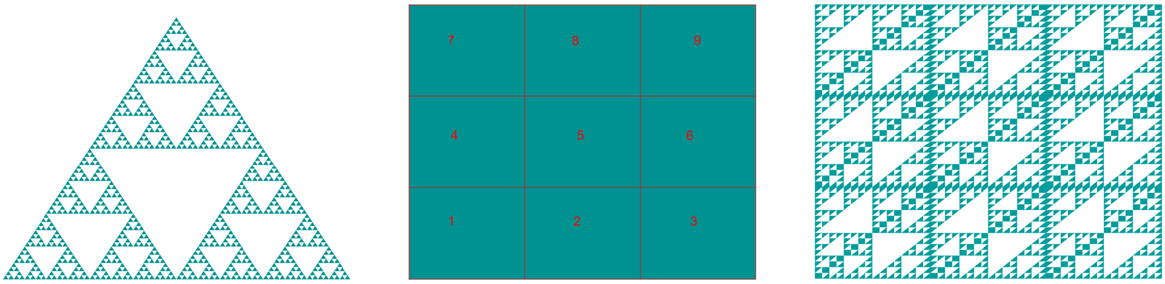

Figure 4. Sierpinski Carpet obtained by $\mathcal{F}_\epsilon^2$ with $\boldsymbol{\epsilon} = (1,1,1,1)$ (left), the labeled result by applying $\mathcal{F}_0^1$ once (middle), and the final desired fractal (right).

Now, we give the first new fractal by combining the left fractal in the top row of Figure 3 with the Sierpinski Carpet in Figure 4 (left). To be more specific, by applying $\mathcal{F}_0^1$ once, we get the labeled result with 9 smaller quadrilaterals as in Figure 4 (middle), and we want the parts in each smaller quadrilateral (the labeled parts in the fractal in Figure 4 (middle)) of the left fractal in the top row of Figure 3 to have the same structure as the Sierpinski Carpet in Figure 4 (left).

To generate such a fractal, we first analyze the multiple function systems generating the target fractal, i.e., the left fractal in the top row of Figure 3, and the Sierpinski Carpet in Figure 4 (left). In fact, $\mathcal{F}_\epsilon^1$ generates the target fractal and $\mathcal{F}_\epsilon^2$ generates the Sierpinski Carpet in Figure 4 (left). Starting from the initial set $\bigcirc_2$, Figure 4 shows that the fractal with the desired structure can be obtained along the path $(1,1,1,1)$ using $\mathcal{F}_\epsilon^2$ while the target fractal is obtained along (0) using $\mathcal{F}_0^1$ once. Let $F_i^1 = \{f_i^1\}$ with $i = 1,\cdots,9$, and use them to give a new multiple function system, with which we can get each desired place by taking each IFS once.

Based on this analysis, we now get the final desired fractal. For this, we first split each desired place into two triangles and take the points of each triangle. With such points, we then construct a new matrix P' and use it to replace the matrix P in the contractive functions f_j^2 with $j = 1,\cdots,4$. In this way, $\mathcal{F}_\epsilon^2$ is changed to be $\mathcal{F}_\epsilon^{2'}$ corresponding to each triangle. Starting from each triangle, take the path $(1,1,1,1)$ using $\mathcal{F}_\epsilon^{2'}$ corresponding to this triangle to get the corresponding scaled fractals, then with all the scaled fractals, we get the final one as, shown in Figure 4 (right).

Example 2. With the fractals in Figure 3 and Figure 4 (left), now we obtain the fractal with the features: keeping the structure in the four corner (see the parts labeled $1,3,7,9$ in Figure 4 (middle)) of the right fractal in the top row of Figure 3, and the absent center part (see the part labeled 5 in Figure 4 (middle)) with the structure as the one in Figure 4 (left) but without the rest parts (see the parts labeled $2, 4, 6, 8$ in Figure 4 (middle)) of the right fractal in the top row of Figure 3.

Similar to the analysis in Example 1, $\mathcal{F}_\epsilon^1$ generates the target fractal, which is the right fractal in the top row of Figure 3, along the path $(1, 1, 1, 1)$ starting from the initial set $\bigcirc_1$. With $\mathcal{F}_\epsilon^2$, we can generate the fractal with the desired structure along the path $(1, 1, 1, 1)$ starting from the initial set $\bigcirc_2$. The desired place in the target fractal can be obtained by taking $\mathcal{F}_i^1 = \{f_i^1\}$, $i = 1, 3, 5, 7, 9$ once.

Now, we generate the final desired fractal. For this, we first split the part (labeled 5 in the fractal in Figure 4 (middle)) into two triangles, then we take the points of each triangle to construct a new matrix P', with which we replace P in f_j^2 with $j = 1, 2, 3, 4$. In this way, $\mathcal{F}_\epsilon^2$ is changed to be $\mathcal{F}_\epsilon^{2'}$ corresponding to each triangle. Starting from each triangle, we apply $\mathcal{F}_\epsilon^{2'}$ corresponding to this triangle along $(1, 1, 1, 1)$ to get the desired scaled fractal. In a similar procedure, we can get the scaled fractal in the four labeled corners. Combining all these fractals, we get the final desired fractal in Figure 5 (left). In a similar way, we can get the right two fractals in Figure 5.

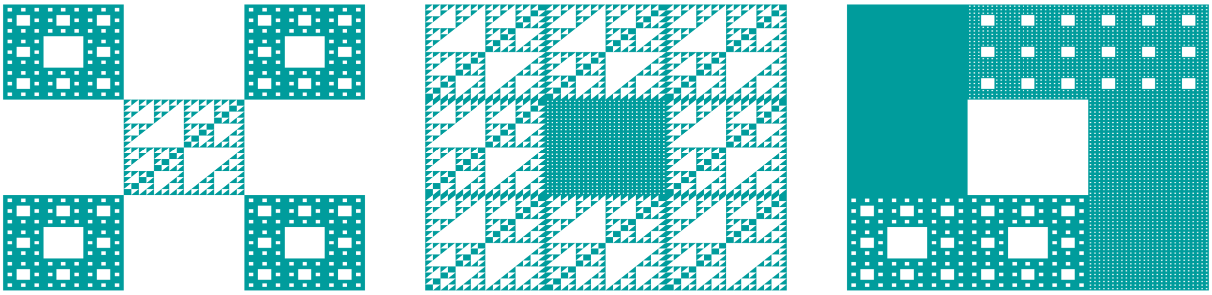

Figure 5. New fractals with different structures at different places.

Example 3. Now, we continue to get a more interesting fractal. For the right fractal in the top row of Figure 3, we want the center quadrilateral and the 8 smaller ones (the labeled ones in Figure 6 (left)) to have the same structure as the Sierpinski Carpet in Figure 4 (left).

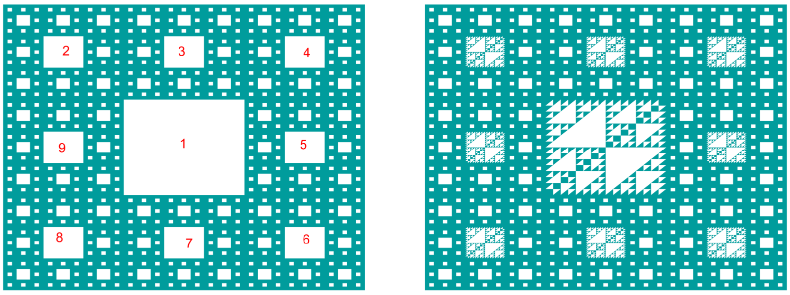

Figure 6. The labeled fractal (left) and the new fractal (right).

The analysis of the corresponding multiple function systems is the same with that in Example 1, except that the part labeled 1 in the fractal in Figure 6 (left) is obtained by taking $\mathcal{F}_5^1 = \{f_5^1\}$ once while the other labeled ones need the relevant iterations twice. For example, the one labeled 2 is obtained by first applying $\mathcal{F}_2^1 = \{f_2^1\}$ and then $\mathcal{F}_5^1$. To generate the final desired fractal, for each desired place, we split it into two triangles, then we take the corresponding points and use them to get a matrix P' to replace P in f_j^2, $j = 1, 2, 3, 4$. In this way, we get the modified multiple function system $\mathcal{F}_\epsilon^{2'}$.

Starting from each triangle, we take the path $(1, 1, 1, 1)$ using $\mathcal{F}_{\epsilon}^{2'}$ corresponding to this triangle to get the corresponding scaled fractal. By combining all the scaled fractals with the target fractal, we get the final desired fractal as shown in Figure 6 (right).

6. Conclusions

This paper proposed a way to generate fractals with desired structures at specific places, which can be done using a new nonuniform multiple function system. To generate such desired fractals, we first analyze two kinds of multiple function systems needed in the generation of such desired fractals, and then generate them using the nonuniform multiple function system. Additionally, we also presented two algorithms corresponding to these procedures for the generation of the final desired fractals for convenience. The given examples show that the nonuniform multiple function system and the two given algorithms can indeed be used to generate fractals with desired structures at specific places. Note that the new fractals in this paper are actually generated by locally controlling the fractals obtained using the multiple function systems. For the given nonuniform multiple function systems and algorithms, it is still difficult to locally control the structures of the fractals with different structures at different locations to obtain new fractals. Therefore, in the future, we hope to give a new way to locally control such fractals to generate more interesting fractals and investigate the corresponding applications.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research is funded by Science Research Project of Hebei Education Department (QN2024065), the Scientific Research and Development Program of Hebei University of Economics and Business (2022QN15), and the National Natural Science Foundation of China (12201453).

The authors thank all the reviewers for their careful reading and for their valuable comments.

Conflict of interest

No conflicts.

References

1. J. Kigami, *Analysis on fractals*, Cambridge: Cambridge University Press, 2001. http://dx.doi.org/10.1017/CBO9780511470943
2. J. Hutchinson, Fractals and self-similarity, *Indiana U. Math. J.*, **30** (1981), 713–747.
3. M. Barnsley, *SuperFractals*, Cambridge: Cambridge University Press, 2006. http://dx.doi.org/10.1017/CBO9781107590168
4. M. Barnsley, J. Elton, D. Hardin, Recurrent iterated function systems, *Constr. Approx.*, **5** (1989), 3–31. http://dx.doi.org/10.1007/BF01889596

5. Y. Fisher, *Fractal image compression: theory and application*, New York: Springer, 1995. http://dx.doi.org/10.1007/978-1-4612-2472-3

6. A. Cavaretta, W. Dahmen, C. Micchelli, *Stationary subdivision*, Boston: American Mathematical Society, 1991.

7. S. Schaefer, D. Levin, R. Goldman, Subdivision schemes and attractors, *Proceedings of the Third Eurographics Symposium on Geometry Processing*, 2005, 171–180. http://dx.doi.org/10.2312/SGP/SGP05/171-180

8. D. Levin, N. Dyn, V. Veedu, Non-stationary versions of fixed-point theory, with applications to fractals and subdivision, *J. Fixed Point Theory Appl.*, **21** (2019), 26. http://dx.doi.org/10.1007/s11784-019-0659-1

9. T. Narayaninsamy, A method to construct irregular fractal curves, *Appl. Math. Comput.*, **192** (2007), 260–273. http://dx.doi.org/10.1016/j.amc.2007.02.120

10. R. Goldman, The fractal nature of Bezier curves, *Proceedings of Geometric Modeling and Processing*, 2004, 3–11. http://dx.doi.org/10.1109/GMAP.2004.1290020

11. N. Dyn, D. Levin, P. Massopust, Attractors of trees of maps and of sequences of maps between spaces with applications to subdivision, *J. Fixed Point Theory Appl.*, **22** (2020), 14. http://dx.doi.org/10.1007/s11784-019-0750-7

12. B. Zhang, H. Zheng, Y. Chen, Multiple-function systems based on regular subdivision, *Fractal Fract.*, **6** (2022), 677. http://dx.doi.org/10.3390/fractalfract6110677

13. J. Andres, J. Fiser, L. Gornierwicz, Fixed points and sets of multivalued contractions: an advanced survey with some new results, *Fixed Point Theor.*, **22** (2021), 15–30. http://dx.doi.org/10.24193/fpt-ro.2021.1.02

14. C. Chui, J. De Villiers, *Wavelet subdivision methods: GEMS for rendering curves and surfaces*, Boca Raton: CRC Press, 2010.

15. J. Warren, H. Weimer, *Subdivision methods for geometric design: a constructive approach*, San Francisco: Morgan Kuafmann Publishers, 2002. http://dx.doi.org/10.1016/B978-1-55860-446-9.X5000-5

AIMS Press