The Fractal of Branches

Team Member:Li Mingzhe, Liang Dong, Wang Jiarui

Teacher:Liu Zhengfeng, Lv Dongliang

School:Zhengzhou No.8 High School

Abstract

This paper mainly focuses on the fractal of branch which can be seen in the life. It is known that the fractal is completely determined by the generator, thus the study of fractal comes down to the study of its generator. Two types of generators are considered in this paper. Each type is divided into four kinds according to whether the node sequence of generator decreases or not, and whether the end of generator is divided or not. This paper presents the 3-dimensional fractal graphics of each kind under the help of computer program.

Key words and phrases:branch,fractal,generator,the sequence of line segment,node sequence.

Contents

1 Introduction

Figure 1: The Fractal of Branch of Jasmine

In this paper,we mainly study the fractal of branches. It began to attract our attention in the summer of last year when we were having rest under a tree. Beside the tree there was an unknown plant with regular branches (Latter we knew it is jasmine). There are many nodes on the branch, two small branches come from each node symmetrically. Thus one unique plane is determined by each node. It is very interesting that the planes determined by adjacent nodes are mutually perpendicular. Furthermore, the small branches grew in the same way as the main branch. This phenomenon deeply amazed us. Later we use *Mathematica program([2])* to simulate the picture of the fractal(figure 1).

We can find many similar examples in our life. Taking one piece of cloud as example, the whole piece and its small part are approximately similar in shape. The tree branches look like its trunk. Also you might notice on map that each part of coastline is similar to the coastline in whole. In fact all these examples in our life are described already by one mathematic subject. That is fractal!

Fractal is a set which consists of points. It should have the following classical geometric properties:

∗ Fractal set has unlimited fine structure;

∗ We can not use traditional geometric language to describe the fractal set. It is not the root of some equations.

∗ Fractal set has self-similar forms, including similar and statistically self-similar;

∗ Fractal set can be defined and produced, generally, by simple methods, such as iteration.
∗ According to the definition of dimension, the fractal dimension of fractal set is greater than the corresponding topological dimension.
Note: Generally, we regard the graphics that satisfies most of above properties as fractal. This definition comes from 'Fractal Art Programming' written by Pan Jingui [1].

With this definition, we can find that the branch of jasmine just described is fractal. Later we also found that many branches of the trees have the above growth rule. Therefore, we use the *fractal of branches* to describe them.

2 Main Text

The fractal is completely determined by its generator. Given a generator,we can produce all kinds of fractal patterns. Therefore, the research of fractal of branches comes down to the research of different generators. Owing to the space is limited, we mainly study two types of generators: the first type of generator can be seen in figure 2(section 2.1,page 4), the second type of generator can be seen in figure 11(section 2.2,page 7).

2.1 The First Type of Generator

In this section we discuss the generator satisfying the following properties: The trunk is divided into n segments, correspondingly, there will be n-1 nodes. Suppose there are two branches coming from each node symmetrically, so the branches together with the trunk are on the same plane. Now notice that there is a unique plane determined by the node. We also assume that the planes determined by alternate nodes are perpendicular(figure 2).

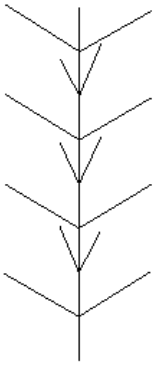

Figure 2: the First Type of Generator

Because the trunk is evenly divided into n segments, we can get two cases according to whether the end of generator is divided or not. We should also consider whether the number of nodes is reduced or not in the process of iteration. Assuming that the number of segments during the kth iteration is X_k, correspondingly,the node number is J_k. Then we have $X_k = J_k + 1$. According to the above discussion,the node number can be divided into two cases: (a) $J_1 = \cdots = J_k = \cdots$; (b) $J_1 > \cdots > J_k > \cdots$.

Thus we divide the first type of generator into four kinds :

(1)the end of generator is divided and the node sequence $\{J_k\}$ *is in the case of* (a) :

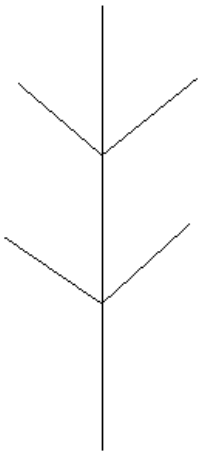

Figure 3:

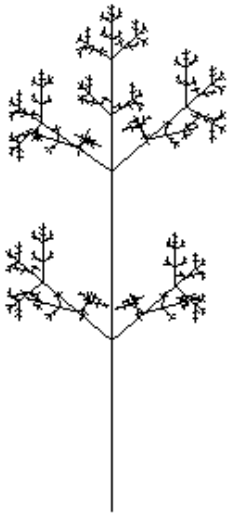

Figure 4:

Let $n = 3$,then $\forall k$,$J_k = 2$. We show the generator (figure 3)and the fractal picture under iteration(figure 4).

(2)the end of generator is divided and $\{J_k\}$ *is in the case of* (b) :

We only take a special case into consideration. Suppose $J_{k+1} = J_k - 1$, because $X_k = J_k + 1$,then $X_{k+1} = X_k - 1$.

Let $n = 5$, we show the generator (figure 5)and the fractal picture under iteration(figure 6).

(3)the end of generator is not divided and $\{J_k\}$ *is in the case of* (a) :

Let $n = 4$, we show the generator (figure 7)and the fractal picture under iteration(figure 8).

(4)the end of generator is not divided and $\{J_k\}$ *is in the case of* (b) :

Because there are many cases for $\{J_k\}$ to satisfy $J_1 > \cdots > J_k > \cdots$, for convenience, we also take a special case into consideration. Suppose

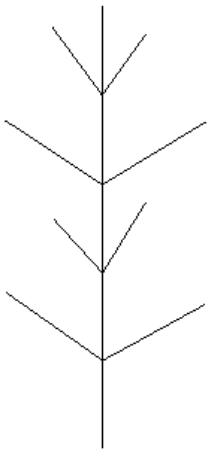

Figure 5:

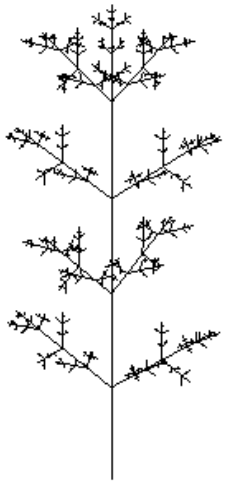

Figure 6:

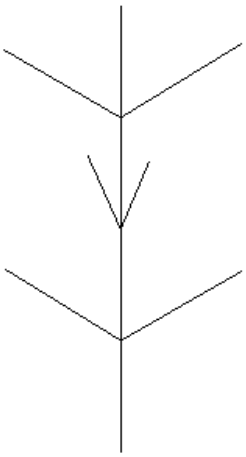

Figure 7:

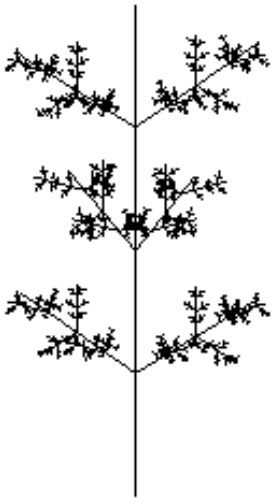

Figure 8:

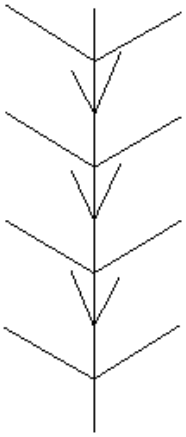

Figure 9:

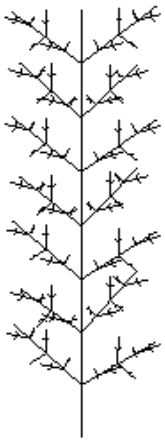

Figure 10:

$X_{k+1} = \lfloor X_k/2 \rfloor$ (where $\lfloor\ \rfloor$ implies to get the maximum integer of a number). As $X_k = J_k + 1$,then $J_{k+1} + 1 = \lfloor (J_k + 1)/2 \rfloor$. It is easy to verify that $\{J_k\}$ satisfies $J_1 > \cdots > J_k > \cdots$.

Let $n = 8$, we show the generator (figure 9)and the fractal picture under iteration(figure 10).

2.2 The Second Type of Generator

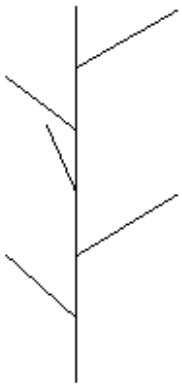

Figure 11: the Second Type of Generator

In this section we discuss the generator satisfying the following properties: The trunk is divided into n segments, correspondingly, there will be n-1 nodes. One branch comes from each node. If the branches are projected

onto the horizontal plane, then the branches which are adjacent form a fixed angle(suppose$\frac{2\pi}{3}$).

Analogous to section 2.1, we divide the second type of generator into four kinds:

(1)the end of generator is divided and $\{J_k\}$ *is a constant sequence:*

Let $n = 6$, we show the fractal picture under iteration(figure 12).

(2)the end of generator is divided and $\{J_k\}$ *is in the case of (b):*

Suppose $X_{k+1} = X_k - 1$notice $X_k = J_k + 1$so $J_{k+1} = J_k - 1$ certainly the node sequence $\{J_k\}$ satisfies $J_1 > \cdots > J_k > \cdots$.

Let $n = 6$, we show the generator (figure 11)and the fractal picture under iteration(figure 13).

(3)the end of generator is not divided and $\{J_k\}$ *is a constant sequence:*

Let $n = 6$, we show the generator (figure 11)and the fractal picture under iteration(figure 14).

(4)the end of generator is not divided and $\{J_k\}$ *is in the case of (b):*

Suppose $\{J_k\}$ is the same as *(2)*, i.e.$J_{k+1} = J_k - 1$. Thus $\{J_k\}$ satisfies $J_1 > \cdots > J_k > \cdots$.

Let $n = 6$, we show the generator (figure 11)and the fractal picture under iteration(figure 15).

Figure 12:

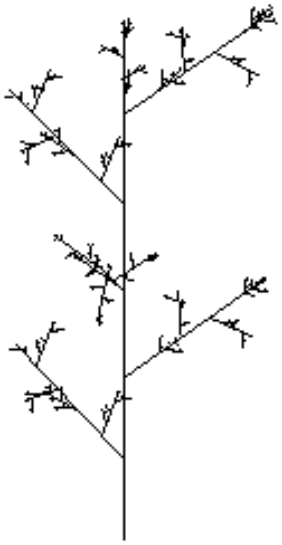

Figure 13:

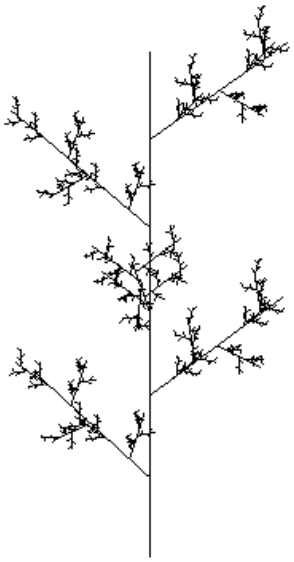

Figure 14:

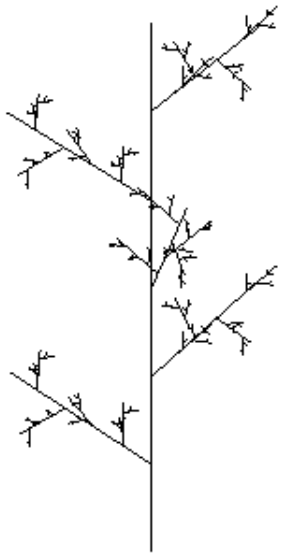

Figure 15:

3 Appendix

With ***Mathematica program***, we have drawn the fractal of branches for each kind. To save space, we just show the program which have been used to draw the second kind of the first type(*Program 1*) and the first kind of the second type(*Program 2*).

Program1 :

$oy = \{\{0,0,0\},\{0,1,0\}\};$
$oz = \{\{0,0,0\},\{0,0,1\}\};$
$LineSet = \{\{\{0,0,0\},\{0,0,1\}\}\};$
$LineCounting = \{1\};$
$n = 5;$
$dividenumber = 3;$
$\theta = \pi/3;$
$f[a\_, b\_, \eta\_] := Cos[\eta] * a + Sin[\eta] * b;$
$LineOnPlane[a\_, b\_, \eta\_] := Block[\{temp, vector, vector1, vector2\},$
$vector1 = a[[2]] - a[[1]];$
$vector2 = b[[2]] - b[[1]];$
$vector = Cross[vector1, vector2];$
$vector2 = Cross[vector, vector1];$
$vector2 = Normalize[vector2];$
$temp = a[[1]] + f[vector1, vector2, \eta];$

```
        {a[[1]], temp}];
LineOffPlane[a_, b_, η_] := Block[{temp, vector1, vector2},
        vector1 = a[[2]] − a[[1]];
        vector2 = b[[2]] − b[[1]];
        vector2 = Cross[vector1, vector2];
        vector2 = Normalize[vector2];
        temp = a[[1]] + f[vector1, vector2, η];
        {a[[1]], temp}];
LineNormalize[a_] := Block[{temp},
        temp = a[[1]] + Normalize[a[[2]] − a[[1]]];
        {a[[1]], temp}];
LineScaling[a_, t_] := Block[{temp},
        temp = a[[1]] + t ∗ (a[[2]] − a[[1]]);
        {a[[1]], temp}];
BranchGenerate[a_, b_, number_] :=
 Block[{i, length, vector, temp, temp1, temp2, tempa, tempb},
   AppendTo[LineSet, {a[[1]]+(number−1)∗(a[[2]]−a[[1]])/number, a[[2]]}];
   length = Norm[a[[2]] − a[[1]]];
   For[i = 1, i < number,
     vector = i ∗ (a[[2]] − a[[1]])/number;
     temp1 = {a[[1]] + vector, a[[2]]};
     temp2 = {a[[1]] + vector, a[[1]] + vector + b[[2]] − b[[1]]};
     temp1 = LineNormalize[temp1];
     If[Mod[i, 2] == 1,
       temp = LineOffPlane[temp1, temp2, θ];
       temp = LineNormalize[temp];
       AppendTo[LineSet, LineScaling[temp, length/dividenumber]];
       temp = LineOffPlane[temp1, temp2, −θ];
       temp = LineNormalize[temp];
       AppendTo[LineSet, LineScaling[temp, length/dividenumber]],
       temp = LineOnPlane[temp1, temp2, θ];
       temp = LineNormalize[temp];
       AppendTo[LineSet, LineScaling[temp, length/dividenumber]];
       temp = LineOnPlane[temp1, temp2, −θ];
       temp = LineNormalize[temp];
       AppendTo[LineSet, LineScaling[temp, length/dividenumber]]
     ];
   i++];
  0];
sum[a_, t_] := Block[{temp = 0, i},
   For[i = 1, i <= t, temp = temp + a[[i]]; i++];
   temp];
BranchGenerate[oz, oy, n];
AppendTo[LineCounting, Length[LineSet] − sum[LineCounting, 1]];
```

```
p[a_] := Block[{temp1, temp2},
  If[a == 0, temp1 = 0,
    For[temp1 = 1; temp2 = 1, temp2 <= a,
      temp1 = temp1 * (2(n - temp2) + 1); temp2 + +]
    ];
  temp1];
For[k = 1; i = 1, k < 4,
    For[r = 1; t = 0, t <= k - 1, r = r + p[t]; t + +];
    For[j = r + 1, j <= r + p[k],
      BranchGenerate[LineSet[[j]], LineSet[[j + 2(n - k)]], n - k];
      For[l = j + 1, l <= j + 2(n - k),
        BranchGenerate[LineSet[[l]], LineSet[[i]], n - k];
        l + +];
      i + +;
      j = j + 2(n - k) + 1];
    k + +];
Show[Table[
  Graphics3D[Line[{LineSet[[i, 1]], LineSet[[i, 2]]}]], {i, 1, Length[LineSet]}],
    AspectRatio- > Automatic, Boxed- > False]
Clear[i, j, k, l, r, t];
Clear[oy, oz, LineSet, LineCounting, dividenumber, n, θ];
Clear[sum, f, p, LineOnPlane, LineOffPlane, LineNormalize,
    LineScaling, BranchGenerate];
```

Program2 :

```
oy = {{0, 0, 0}, {0, 1, 0}};
oz = {{0, 0, 0}, {0, 0, 1}};
LineSet = {{{0, 0, 0}, {0, 0, 1}}};
LineCounting = {1};
α = π/3;
θ = 2π/3;
n = 6;
dividenumber = 3;
LineScaling[a_, t_] := Block[{temp},
    temp = a[[1]] + t * (a[[2]] - a[[1]]);
    {a[[1]], temp}];
GetLine[a_, b_, η_] :=
  Block[{temp, vector, vectora, vectorb, vectorc, length},
    length = Norm[a[[2]] - a[[1]]];
    vectora = Normalize[a[[2]] - a[[1]]];
    vectorb = b[[2]] - b[[1]];
    vectorc = Normalize[Cross[vectora, vectorb]];
    vectorb = Cross[vectorc, vectora];
    vector = Cos[α] * vectora + Sin[α] * Cos[η] * vectorb
```

```
                +Sin[α] * Sin[η] * vectorc;
      temp = a[[1]], a[[1]] + vector;
      temp = LineScaling[temp, length/dividenumber];
      temp];
BranchGenerate[a_, b_, number_] :=
  Block[{i, vector, temp, temp1, temp2},
    AppendTo[LineSet, a[[1]] + (number − 1) * (a[[2]] − a[[1]])/number, a[[2]]];
    For[i = 1, i < number,
      vector = i * (a[[2]] − a[[1]])/number;
      temp1 = a[[1]] + vector, a[[2]] + vector;
      temp2 = {a[[1]] + vector, a[[1]] + vector + b[[2]] − b[[1]]};
      temp = GetLine[temp1, temp2, (−i + 1.5) * θ];
      AppendTo[LineSet, temp];
      i + +]; 0];
sum[a_, t_] := Block[{temp = 0, i},
    For[i = 1, i <= t, temp = temp + a[[i]]; i + +];
    temp];
BranchGenerate[oz, oy, n];
AppendTo[LineCounting, Length[LineSet] − sum[LineCounting, 1]];
total = 3;
For[i = 1; k = 1, k <= total, k + +,
  r = (n^k − 1)/(n − 1);
  For[j = r + 1, j <= r + n^k, j = j + n,
    BranchGenerate[LineSet[[j]], LineSet[[j + n − 1]], n];
    For[l = j + 1, l < j + n, l + +,
      BranchGenerate[LineSet[[l]], LineSet[[i]], n]
      ];
    i + +];
  ];
Show[Table[Graphics3D[Line[{LineSet[[i, 1]], LineSet[[i, 2]]}]],
  {i, 1, Length[LineSet]}], AspectRatio− > Automatic, Boxed− > False]
Clear[oy, oz, α, θ, LineSet, LineCounting, dividenumber];
Clear[n, i, j, k, l, p, q, r, total];
Clear[sum, LineNormalize, LineScaling, GetLine, BranchGenerate];
```

References

[1] Pan Jingui, *Fractal Art Programing*, Nanjing University Press, 1998.3.

[2] Li Shangzhi, Chen Falai, Zhang Yunhua and Wu Yaohua, *Maths Experiment*, Higher Education Press, Second Edition, 2004.8.