

# World of Fractal



Submitted By:

Chen Ting (Matric No. U017596H)

Huang Liming (Matric No. U015818R)

# Contents

<b>1. Introduction to Fractal</b>		
1.1 Definition of Fractal	1	1
1.2 Properties of Fractal	1	1
<b>2. The Sierpinski Triangle</b>		
2.1 Construct the Sierpinski Triangle	2	2
2.2 Area of the Sierpinski Triangle	3	3
2.3 Number of Triangles Grow	3	3
<b>3. Von Koch Curve</b>		
3.1 Construct the Koch Edge	4	4
3.2 Properties of Koch Edge		
3.2.1 Total length of the Koch Edge		5
3.2.2 Total number of the Koch Edge		5
3.3 Other Examples of Koch Curve	6	6
<b>4. The Hilbert Curve</b>		
4.1 Construct the Hilbert Curve		8
4.2 The L-syste		9
4.3 Length of the Curve		9
<b>5. Applications of Fractals</b>		
5.1 Tiling up the Plane		10
5.1.1 Tessellation with Von Koch Curves		10
5.1.2 Space-filling fractal		12
5.2 Reproducing Realistic Images		13
5.3 Fractal Compression		16
<b>6. Conclusion</b>		17
<b>Appendix</b>		
A. Koch Edge		18
B. Koch Star		18
C. Binary Tree		19
D. Random Binary Tree		20

## **1 Introduction to Fractal**

### 1.1 Definition of Fractal

The formal mathematical definition of fractal is defined by Benoit Mandelbrot. It says that a fractal is a set for which the Hausdorff Besicovich dimension strictly exceeds the topological dimension. However, this is a very abstract definition.

Generally, we can define a fractal as a rough or fragmented geometric shape that can be subdivided in parts, each of which is (at least approximately) a reduced-size copy of the whole. Fractals are generally self-similar and independent of scale.

### 1.2 Properties of Fractal

A fractal is a geometric figure or natural object that combines the following characteristics:

- a) Its parts have the same form or structure as the whole, except that they are at a different scale and may be slightly deformed;
- b) Its form is extremely irregular or fragmented, and remains so, whatever the scale of examination;
- c) It contains "distinct elements" whose scales are very varied and cover a large range;
- d) Formation by iteration;
- e) Fractional dimension.

## 2 The Sierpinski Triangle

The Sierpinski's Triangle is named after the Polish mathematician Waclaw

Sierpinski who described some of its interesting properties in 1916. It is one of the simplest fractal shapes in existence

It can be generated by infinitely repeating a procedure of connecting the midpoints of each side of the triangle to form four separate triangles, and cutting out the triangle in the center.

### 2.1 Construct the Sierpinski Triangle

Taking a equilateral triangle as an example:

1. Start with the equilateral triangle.



2. Connet the midpoints of each side of the triangle to form four separate triangles.



3. Cut out the triangle in the center.

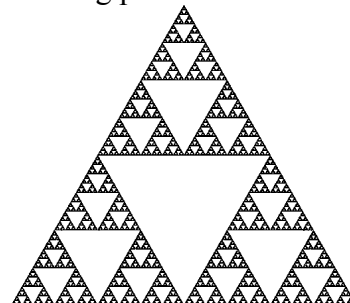


4. Repeat the steps 1, 2 and 3 on the three black triangle left behind.



The center triangle of each black triangle at the corner were cut out as well.

5. Further repetition with adequate screen resolution will give the following pattern.



2.2 Area of the Sierpinski Triangle

As we know, at each level, one quarter of the triangle is removed. That is, three quarters of the area of the original triangle is left after the first iteration. Thus, it is not hard to infer that after n iterations, the area of the Sierpinski's Triangle would be  $(0.75)^n$  times the area of the original triangle. So after an infinite number of iterations, you would find there was no area at all.

2.3 Number of Triangles Grow

After observing the number of triangles pointing down for several iterations, we have the following table:

Iteration	No. of Triangles pointing down
1	1
2	4
3	13
4	40
5	121

From the table, we can come up with a general formula to predict the number of triangles being removed for any iteration:

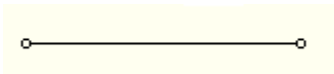
At nth iteration, the number of triangles being removed,  $N = \sum_{i=0}^{i=n-1} 3^i$

### 3 Von Koch Curve

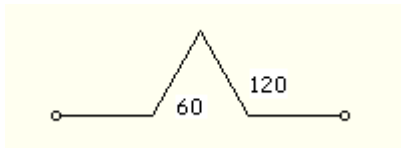
The curve of Von Koch is generated by a simple geometric procedure which can be iterated an infinite number of times by dividing a straight line segment into three equal parts and substituting the intermediate part with two segments of the same length. The Von Koch Curve is a very elementary example of fractal, as it follows a simple rule of construction.

#### 3.1 Construct the Koch Edge

1. Start with a straight line.



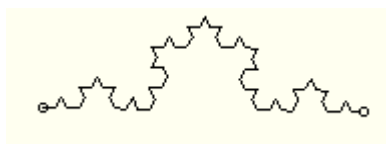
2. The straight line is divided into 3 equal parts, and the middle part is replaced by two linear segments at angles  $60^\circ$  and  $120^\circ$ .



3. Repeat the steps 1 and 2 to the four line segments generated in two



4. Further iterations will generate the following curves




repeat three times



repeat four times

### 3.2 Properties of Koch Edge

The Von Koch Curve clearly shows the **self-similarity** of fractal. The same pattern  appears everywhere along the curve in different scale, from visible to infinitesimal.

Ideally the iteration process should go on indefinitely; however, in practice, the curve displayed on the screen no longer changes when the elementary side becomes less than the pitch. And the iteration can be stopped as well.

#### *3.2.1 Total length of the Koch Edge*

In order to maintain the displacement in between the two points constant, the length of the size multiplied by  $4/3$  in each iteration.

From this simple rule of iteration, we can come up with a formula for the total length of the Koch Edge in the  $n$ th iteration.

$$L = (4/3)^n$$

#### *3.2.2 Total number of the Koch Edge*

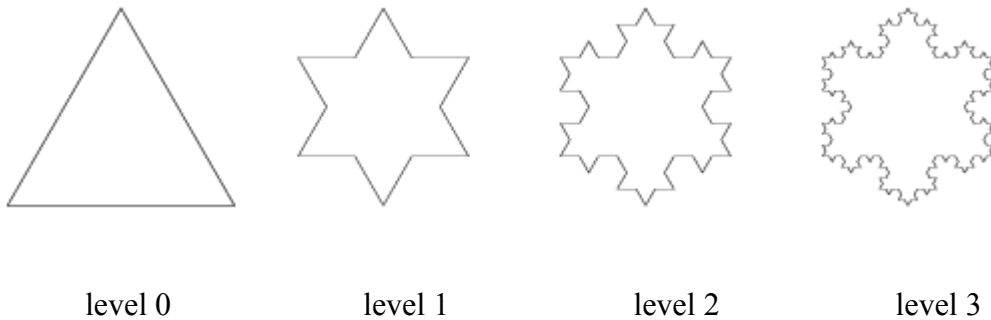
After each iteration, every single edge turns into four equal sized segments with length  $1/3$  of the original length.

Thus, after  $n$  iterations, the total number of the Koch Edge will be  $4^n$ .

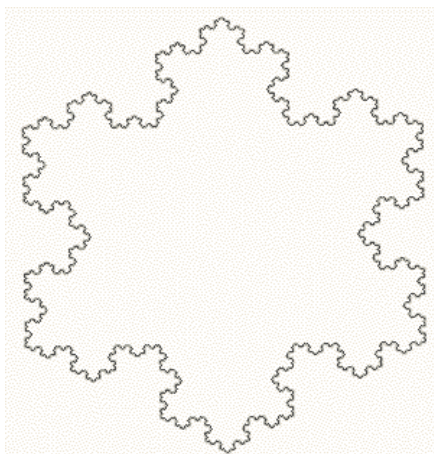
3.3 Other Examples of Koch Curve

a) Koch Snowflake

If we apply the above process on the two sides of an equilateral triangle(exclude the inner side), we will have the following pattern:



At very high level, we will have the below image:




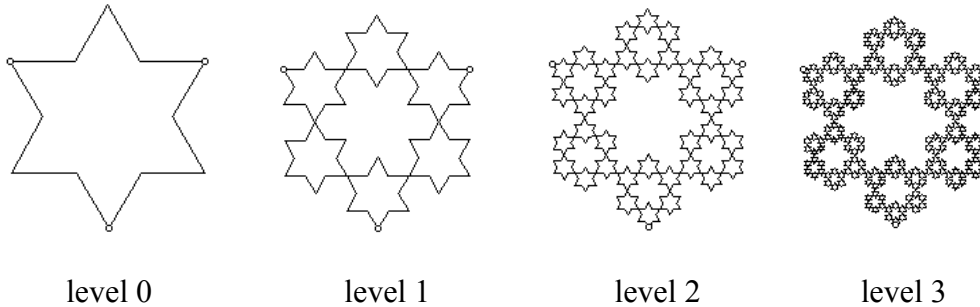
This genuine Koch Curve is also called snowflake curve.

b) Koch Star

Instead of equilateral triangle, the same process can be applied to sides of any polygon.

Now, we take the hexagram as another example. This time round, we include

the inner sides of the hexagram. At each level, new smaller hexagrams  are generated at the six vertices of the original hexagram. The newly generated hexagram is similar to the original one.



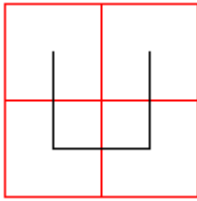
(We generated the curves of Koch Edge and Koch Star by the software called Geometer's Sketchpad)

## 4 The Hilbert Curve

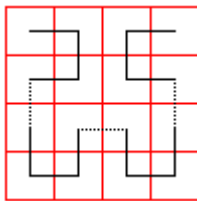
The Hilbert Curve was first introduced by David Hilbert (1862-1943). This curve is called a space-filling curve, as it will eventually cover the entire plane after several iterations.

### 4.1 Construct the Hilbert Curve

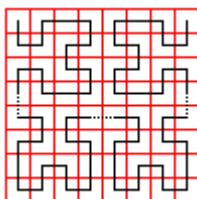
1. Start with the basic staple-like shape as depicted on the first diagram



2. Shrink the previous curve to half its size. Simultaneously, decrease the grid size by the factor of two. Place four copies of the curve on the grid. The lower two must be placed directly as they are. The upper two must be rotated a quarter turn - one left, another right. Lastly, connect the four pieces with short straight segments to obtain the next step curve. Sometimes the connecting segments are horizontal, and sometimes they are vertical.

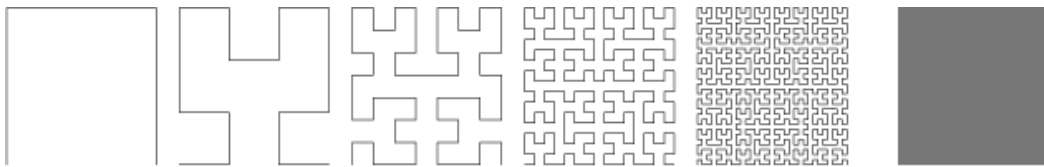


3. All the rest of the curves are created sequentially one from another using the same algorithm.



#### 4.2 The L-system

Unlike the Sierpinski Triangle and Von Koch Curve, in the process of constructing the Hilbert Curve, the same staple-like shape gets shrink and transform into other places. Some of them are rotated by 90 degree as well. This algorithm is called Lindenmayer System (L-system). This is a string rewriting system that is particularly used to generate fractal with dimension between 1 and 2. After transformation, the curve has to be linked by introducing some line segments. That is why, after certain iteration (about 7<sup>th</sup> to 9<sup>th</sup>), the curve will eventually cover up the whole plane.



#### 4.3 Length of the Curve

At every iteration, the fractal becomes 2 units longer. After n iterations the length of the curve will grow up to 2n units longer.

## 5 Applications of Fractals

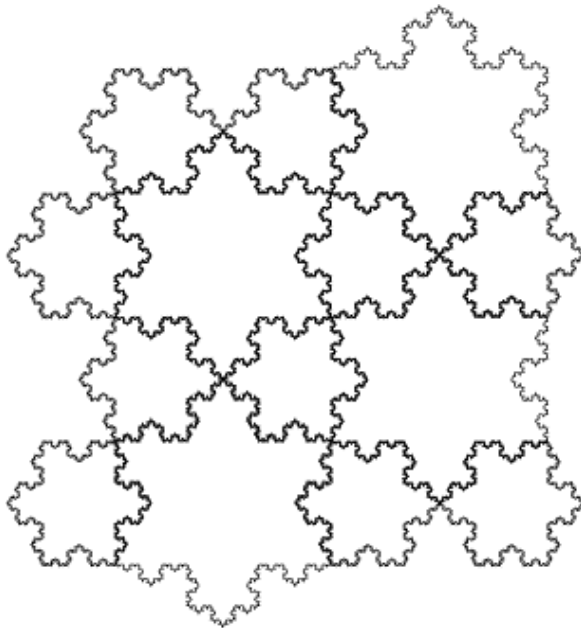
In fact, there are hundreds of applications of fractal from different aspects, such as generating computer aided mammography, creating realistic image, producing fractal music and etc. Here, we are only going to describe some of these practical applications.

### 5.1 Tiling up the Plane

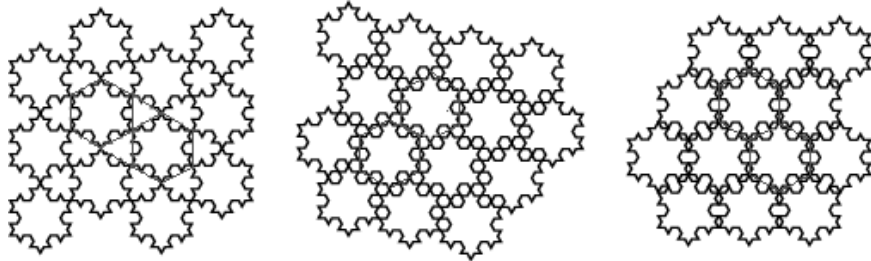
Because of the beautiful image of fractals, they are often used to fulfill the plane. Fractal images can be sometimes found in patterns on the floor, window lattice, or on the carpet.

#### *5.1.1 Tessellation with Von Koch Curves*

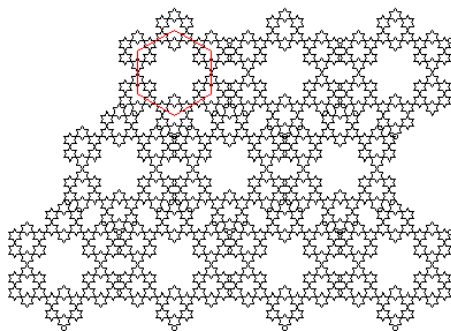
A very nice tile for tessellation is the Koch Snowflake we discussed before. Here, we use two sizes of Koch Snowflakes in area ratio 1:3 to tile the plane; we will get an image called Mandelbrot as shown below:



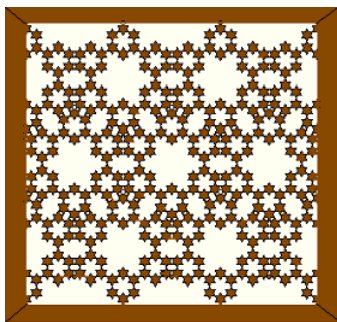
In fact, when we tessellate the Koch Snowflake, we are actually considering the Snowflake as a hexagon, and then tessellate the hexagon in various ways. Here are other examples of tessellation with Koch Snowflakes.



Following the same rule, we use the Koch Star to generate a new tessellation pattern on our own. We now consider the Koch Star as a hexagon, and tessellate this hexagon all over the plane. We have:



And this kind of tessellation can be usually found in ancient Chinese window lattice. Here, we have made use of the above pattern and generated an ideal Chinese window lattice, shown below:



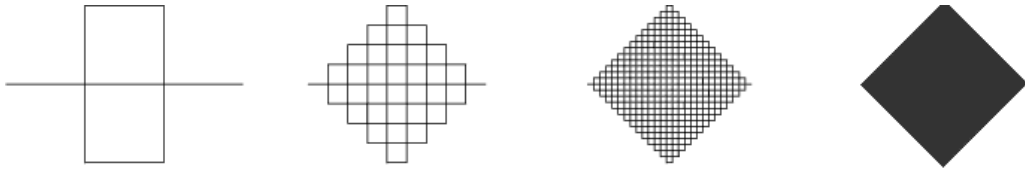
(Tessellation of Koch Star and the Chinese window lattice are generated by Geometer' Sketchpad)

### 5.1.2 Space-filling fractal

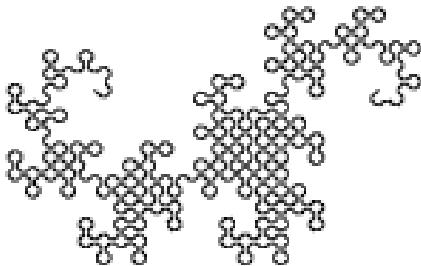
There is a type of fractals continuously attempts to fill in the empty area without any holes as it goes along its iteration. We call them space-filling fractals.

The Hilbert Curve we discussed before is a good example of space-filling fractal.

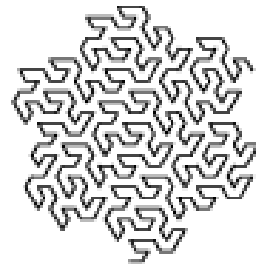
Another very interesting space-filling fractal is the Peano Curve. Unlike the Hilbert Curve which fill up a square plane, the Peano Curve fills up a plane in rhombus shape.



Other space-filling fractal such as Peano-Gosper Curve, Dragon Curve are shown below:



Dragon Curve



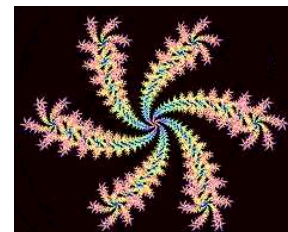
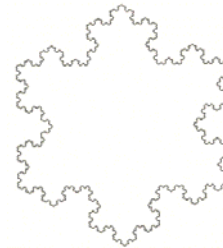
Peano-Gosper Curve

5.2 Reproducing Realistic Images



A very important application of fractal is to reproduce the natural images such as clouds, trees, mountains and etc. This is because many natural things, such as plants, are very complex and exhibit some self-similarity. The complexity of fractals and the their property of self-similarity allows fractals to reproduce a large set of real-world images.

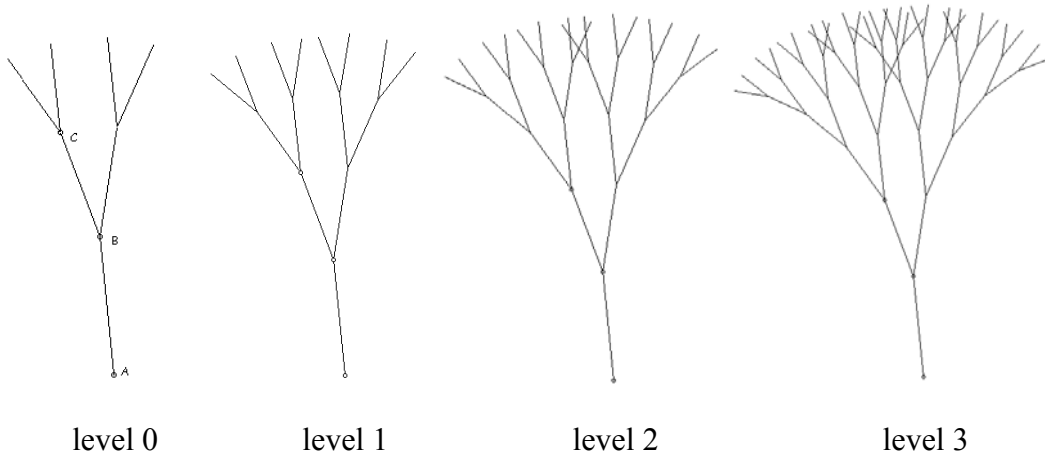
In fact, the Koch Snowflake we discussed before is already a possible starting point for the design complex natural curves. The Koch Snowflake itself is already very similar to the snowflake in the natural world. However, everyone can tell that the Koch Snowflake does not look like a natural curve. It is too regular in shape. This regularity comes from the strictness of its construction process.



Thus, the first step to loosen the regularity is to introduce some random fluctuations during the construction process. The final curve can be strongly modified when the iteration process is changed. Mandelbrot gives the following example, where the symmetry of the original Von Koch curve is broken:



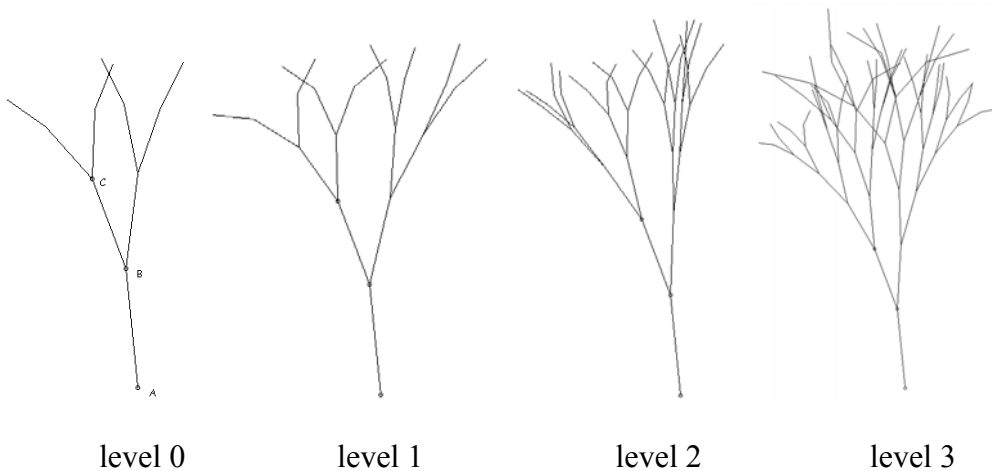
To further explore how random factor affects the iteration process, we did an experiment on the fractal called Binary Tree with Geometer's Sketchpad. The original Binary Tree goes like this:



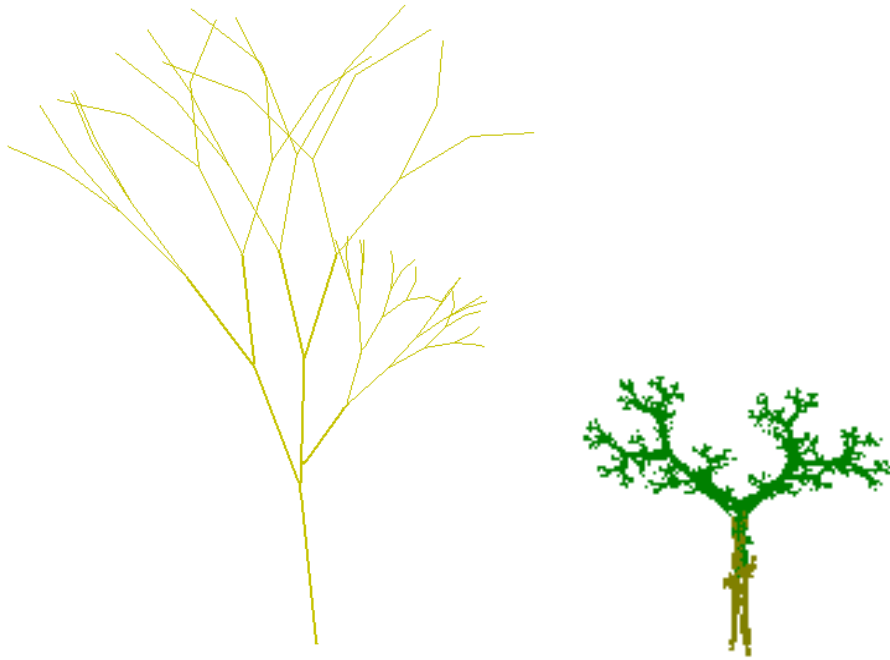
As we can see from above, the Binary Tree grows up very regularly. The ratio of the stem AB and bough BC is kept constant for all iterations. The angle of the boughs bended at the fork is also strictly kept the same.

Now, we modify the iteration process by introducing some random factors on the angle of the boughs bended at the fork.

With the same starting point A, B, C, we have:



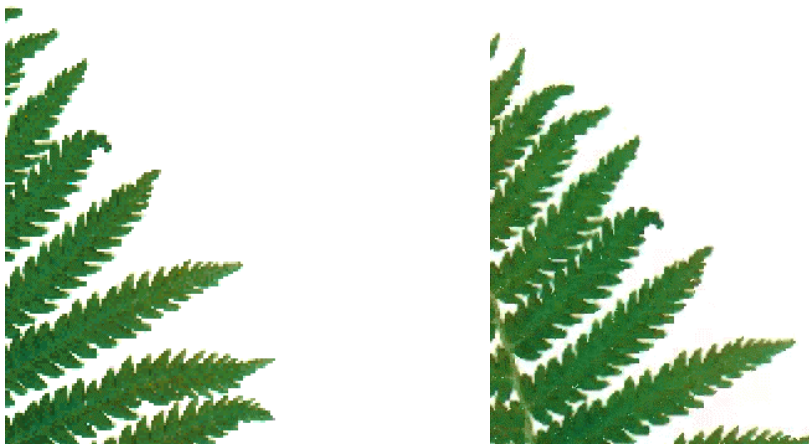
The random binary tree generated no long has the property of regularity. By introducing the random color on the binary tree, we can actually produce a natural tree image ourselves. Of course, further modification will make this skeleton tree structure more like a real tree.



### 5.3 Fractal Compression

We've already known that fractals can be used to resemble real-life object, and we've discussed how it works as well. However, as researchers proceeded on this field, they came upon the question: instead of generating fractals for real-life object, is it possible to do it that other way around? In other words, to turn an image into a fractal by approximation.

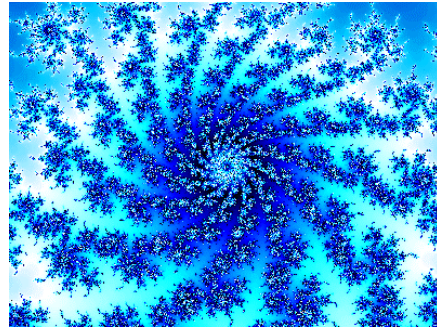
With the faster computer nowadays, they actually achieve the goal of compressing real images into fractal images. The advantage of using fractal compression is being able to enlarge the fractal without getting a blocky picture. We could say the fractal "smooths out the details," making it much better for pictures required some sort of zoom. That is because fractals are infinitely detailed and so enlargement causes it is "smooth" the rough spots. Compare the two pictures of ferns below:



If you compare the two closely, you will find that the one on the right has more details. The one on the left is just an enlargement of a bitmap. The compressed image (on the right), however, has much more detail.

## 6 Conclusion

By now, we have talked about what fractals are, and we used some famous fractals to illustrate how we can create a fractal image. However, fractals are much more than that.



The study of fractals has been around longer than you think. Over 100 years ago, the idea of fractals was introduced in mathematical journals and literature.

Nevertheless, the study of fractal has never stopped. Even now, the new fractal images are generated everyday around the world. The researchers are still working dedicatedly to find out more about fractals, so that we can make use of them in better ways.

## Appendix

### Codes for generating fractals in Geometer's sketchpad

#### A. Koch Edge

**Steps:**

1. Let [j] = Segment between Point A and Point B (hidden).
2. Let [C] = Image of Point A dilated by 33.33% about center Point B (hidden).
3. Let [D] = Image of Point A dilated by 66.67% about center Point B (hidden).
4. Let [E] = Image of Point [C] rotated 60 degrees about center Point [D] (hidden).
5. Let [k] = Segment between Point A and Point [D].
6. Let [m] = Segment between Point [D] and Point [E].
7. Let [n] = Segment between Point [E] and Point [C].
8. Let [p] = Segment between Point [C] and Point B.
9. Recurse on A and [D].
10. Recurse on [D] and [E].
11. Recurse on [E] and [C].
12. Recurse on [C] and B.

#### B. Koch Star

**Steps:**

1. Let [j] = Segment between Point A and Point B (hidden).
2. Let [C] = Image of Point A dilated by 33.33% about center Point B (hidden).
3. Let [D] = Image of Point A dilated by 66.67% about center Point B (hidden).
4. Let [E] = Image of Point [C] rotated 60 degrees about center Point [D] (hidden).
5. Let [k] = Segment between Point A and Point [D].
6. Let [m] = Segment between Point [D] and Point [E].
7. Let [n] = Segment between Point [E] and Point [C].
8. Let [p] = Segment between Point [C] and Point B.
9. Let [q] = Segment between Point B and Point C (hidden).
10. Let [G] = Image of Point B dilated by 33.33% about center Point C (hidden).
11. Let [H] = Image of Point B dilated by 66.67% about center Point C (hidden).
12. Let [J] = Image of Point [G] rotated 60 degrees about center Point [H] (hidden).
13. Let [r] = Segment between Point B and Point [H].
14. Let [s] = Segment between Point [H] and Point [J].
15. Let [t] = Segment between Point [J] and Point [G].
16. Let [u] = Segment between Point [G] and Point C.
17. Let [v] = Segment between Point C and Point A (hidden).
18. Let [K] = Image of Point C dilated by 33.33% about center Point A (hidden).
19. Let [L] = Image of Point C dilated by 66.67% about center Point A (hidden).
20. Let [M] = Image of Point [K] rotated 60 degrees about center Point [L] (hidden).
21. Let [w] = Segment between Point C and Point [L].
22. Let [x] = Segment between Point [L] and Point [M].
23. Let [y] = Segment between Point [M] and Point [K].
24. Let [z] = Segment between Point [K] and Point A.
25. Recurse on [K], A and [D].
26. Recurse on [D], [E] and [C].
27. Recurse on [C], B and [H].
28. Recurse on [H], [J] and [G].
29. Recurse on [G], C and [L].
30. Recurse on [L], [M] and [K].

### C. Binary Tree

**Steps:**

1. Let [j] = Segment between Point root and Point fork.
2. Let [k] = Segment between Point fork and Point bough.
3. Let [A] = Image of Point bough reflected across mirror Segment [j] (hidden).
4. Let [l] = Image of Segment [k] reflected across mirror Segment [j].
5. Let [B] = Image of Point fork dilated by ratio  $|forkbough|/|forkroot|$  about center Point bough (hidden).
6. Let [C] = Image of Point fork rotated by angle  $root-fork-bough$  about center Point bough (hidden).
7. Let [m] = Ray between Point bough and Point [C] (hidden).
8. Let [c1] = Circle with center at Point bough passing through Point [B] (hidden).
9. Let [D] = Intersection of Circle [c1] and Ray [m] (hidden).
10. Let [n] = Segment between Point [D] and Point bough.
11. Let [o] = Segment between Point fork and Point [A].
12. Let [E] = Image of Point [A] reflected across mirror Segment [j] (hidden).
13. Let [p] = Image of Segment [o] reflected across mirror Segment [j].
14. Let [F] = Image of Point fork dilated by ratio  $|fork[A]|/|forkroot|$  about center Point [A] (hidden).
15. Let [G] = Image of Point fork rotated by angle  $root-fork-[A]$  about center Point [A] (hidden).
16. Let [q] = Ray between Point [A] and Point [G] (hidden).
17. Let [c2] = Circle with center at Point [A] passing through Point [F] (hidden).
18. Let [H] = Intersection of Circle [c2] and Ray [q] (hidden).
19. Let [r] = Segment between Point [H] and Point [A].
20. Let [s] = Image of Segment [n] reflected across mirror Segment [p].
21. Let [t] = Image of Segment [r] reflected across mirror Segment [o].
22. Recurse on fork, bough and [D].
23. Recurse on fork, [A] and [H].

## D. Random Binary Tree

### Steps:

1. Let [j] = Segment between Point root and Point fork.
2. Let [k] = Segment between Point fork and Point bough.
3. Let [A] = Image of Point bough reflected across mirror Segment [j] (hidden).
4. Let [B] = Image of Point [A] rotated 10 degrees about center Point fork (hidden).
5. Let [C] = Image of Point [A] rotated 10 degrees about center Point fork (hidden).
6. Let [a1] = Arc through Points [C], [A] and [B] (hidden).
7. Let bough' = Random point on Arc [a1] (hidden).
8. Let [l] = Segment between Point fork and Point bough'.
9. Let [E] = Image of Point fork dilated by ratio  $|forkbough|/|forkroot|$  about center Point bough (hidden).
10. Let [H] = Image of Point [E] dilated by 90.00% about center Point bough (hidden).
11. Let [I] = Image of Point [E] dilated by 110.00% about center Point bough (hidden).
12. Let [n] = Segment between Point [l] and Point [H] (hidden).
13. Let [J] = Random point on Segment [n] (hidden).
14. Let [K] = Image of Point fork rotated by angle root-fork-bough about center Point bough (hidden).
15. Let [L] = Image of Point [K] rotated 10 degrees about center Point bough (hidden).
16. Let [M] = Image of Point [K] rotated 10 degrees about center Point bough (hidden).
17. Let [a2] = Arc through Points [L], [K] and [M] (hidden).
18. Let [N] = Random point on Arc [a2] (hidden).
19. Let [c1] = Circle with center at Point bough passing through Point [J] (hidden).
20. Let [o] = Ray between Point bough and Point [N] (hidden).
21. Let [O] = Intersection of Circle [c1] and Ray [o] (hidden).
22. Let [p] = Segment between Point bough and Point [O].
23. Let [P] = Image of Point fork dilated by ratio  $|forkbough'|/|forkroot|$  about center Point bough' (hidden).
24. Let [Q] = Image of Point [P] dilated by 110.00% about center Point bough' (hidden).
25. Let [R] = Image of Point [P] dilated by 90.00% about center Point bough' (hidden).
26. Let [q] = Segment between Point [R] and Point [Q] (hidden).
27. Let [S] = Random point on Segment [q] (hidden).
28. Let [T] = Image of Point fork rotated by angle root-fork-bough' about center Point bough' (hidden).
29. Let [U] = Image of Point [T] rotated 10 degrees about center Point bough' (hidden).
30. Let [V] = Image of Point [T] rotated 10 degrees about center Point bough' (hidden).
31. Let [a3] = Arc through Points [U], [T] and [V] (hidden).
32. Let [W] = Random point on Arc [a3] (hidden).
33. Let [c2] = Circle with center at Point bough' passing through Point [S] (hidden).
34. Let [r] = Ray between Point bough' and Point [W] (hidden).
35. Let [X] = Intersection of Circle [c2] and Ray [r] (hidden).
36. Let [s] = Segment between Point bough' and Point [X].
37. Recurse on fork, bough and [O].
38. Recurse on fork, bough' and [X].