

**TOPICS IN FRACTAL AND NON-EUCLIDEAN GEOMETRIES,  
INSTRUCTIONAL TECHNOLOGY, AND THE LOWER-DIVISION COLLEGE  
MATHEMATICS CURRICULUM**

Reza Sarhangi  
Department of Mathematics  
Towson University  
8000 York Road  
Towson MD 21215, USA  
[rsarhangi@towson.edu](mailto:rsarhangi@towson.edu)

Mehri Arfaei  
Instructional Technology  
Towson University  
8000 York Road  
Towson MD 21215, USA  
[marfaei@towson.edu](mailto:marfaei@towson.edu)

**Abstract**

*Similarity* plays an important role in art, industry, architecture, and technology. For a broad understanding of a large construction project, a common and perhaps necessary approach is to generate a model; a scaled-down version of the larger structure that in every aspect is similar to the larger structure.

In this article we would like to show how in a lower-division mathematics course, that perhaps can be considered as a general education curriculum class, the terms “*magnification*”, “*similarity*”, “*scale factor*” and “*self-similarity*” can be explored through a software such as the *Geometer’s Sketchpad*. The use of Sketchpad, and in general the computer, greatly enhances the understanding of these geometric ideas. The self-similarity concept is normally categorized as a part of a bigger spectrum that is called fractal geometry. Even though we present some examples of fractal geometry, we leave the task of studying this geometry to other articles.

Among some new instructional technologies, there are two software utilities that provide students the opportunity of experiencing the properties of the hyperbolic geometry dynamically: *NonEuclid*, creates an interactive environment for learning and exploring non-Euclidean geometry, and a utility which has been developed based on the *Geometer’s Sketchpad* that allows students to study aspects of hyperbolic geometry using the Poincaré model. These two utilities will be introduced in this article.

**Self-Similarity and Fractal Geometry**

Fractal geometry is a modern branch of geometry that has captured the attention and enthusiasm of mathematicians and scientists, as well as computer graphic and visual artists around the world. We do not intend to study this particular geometry in this article. Nevertheless, this section serves as an introduction to the main property of fractal geometry, which exhibits the quality of self-similarity. A geometric figure is called **self-similar**, if parts of it contain smaller scale replicas of the whole. The underlying idea of self-similarity is immediate and intuitive.

One way to develop a self-similar image is to begin with a figure, which is called the **initiator**, and a mathematical rule, which is called the **algorithm**. After the first

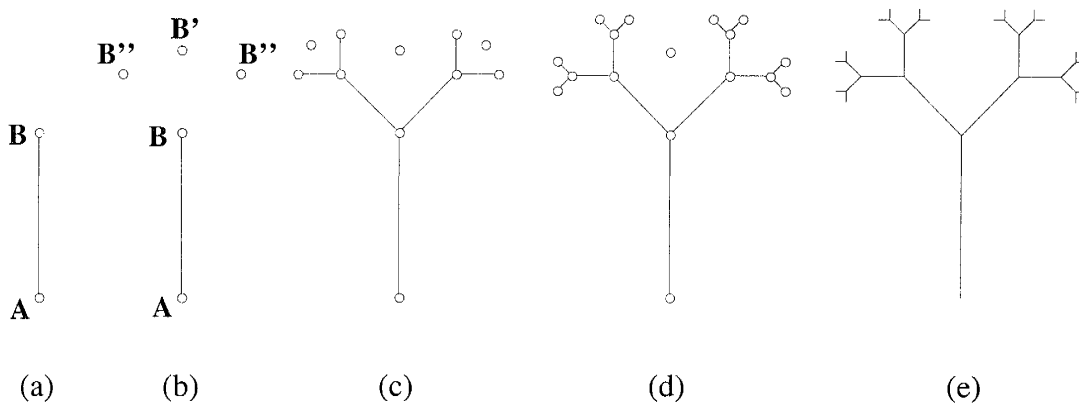
application of the algorithm we create the first generation, which is called the **generator**. Now we apply the algorithm to the first generation to obtain the second generation and so on. Each step is called iteration. The process of repeating an application of the algorithm to the output to generate a new input is called an **iterative process**.

### Constructing a Binary Tree Fractal Using Sketchpad

We would like to use Sketchpad to create the first few stages of the binary tree fractal that is illustrated in Figure 1.e. Please note that the labels are shown to help you follow the directions, but you don't need to label your points in your drawing.

Construct the vertical segment  $\overline{AB}$  (Figure 1.a). Select and mark  $A$  as the center and then dilate point  $B$  upwards by a scale factor of 3 to 2 to find  $B'$ . With this action, we informed the computer that the size of each branch of  $V$  is  $\frac{1}{2}$  of the original segment. Now mark  $B$  as a center and rotate first  $45^\circ$  and then  $-45^\circ$  to find  $B''$  on the left and  $B'''$  on the right, respectively. This is all we need to have: a Y shape that can be iterated with smaller replicas on the endpoint of each of the two top branches (Figure 1.b).

To set up the iterative steps to create new stages, we need to iterate on points  $A$  and  $B$  as follows: Select  $A$  and then  $B$ . Then go to the **Transform** menu and choose "Iterate". The iterate dialog box will appear (You can click on the box and drag it to a corner). Click on  $B$  to set this point as the image of  $A$ . Now click on the left side  $B''$  to set this point as the image of  $B$ . This iteration only produces the left side branches of your tree. We need to repeat this action for the right side of our tree. For this, we need to add a new mapping using  $B$  and the right side  $B'''$ . On the dialog box, we click on "Structure" and choose **Add New Map**. Click on  $B$  to set this point as the image of  $A$  and click on the right side  $B'''$  to set this point as the image of  $B$ . Then we have the second generation of the tree (Figure 1. c). You can increase the stages of your fractal tree by clicking on "Display" on the dialog box and choosing **Increase Iterations +**. Figure 1.d shows the third iteration. In this step, if you want your iterations to show only non-point images you should click on "Structure" and then choose **Only Non-Point Images**. In order to view your final version of the binary tree, you may click on Iterate and your beautiful tree will appear! (Figure 1.e)



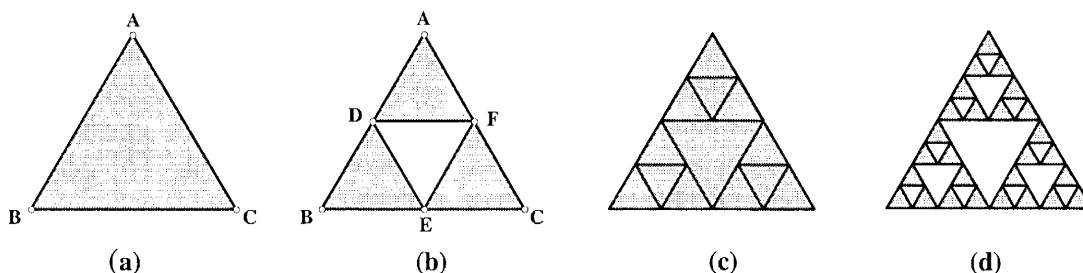
**Figure 1.** The steps of the construction of a Binary Tree Fractal using Sketchpad

### Constructing a Sierpinski Gasket Using Sketchpad

Construct the equilateral triangle  $\triangle ABC$  and its interior (Figure 2.a). We then construct the midpoints of its sides, as is presented in Figure 1.b. Now we want to iterate on  $A$ ,  $B$ , and  $C$ . For this, select  $A$ ,  $B$ , and  $C$  in order and click on “Iterate” on the **Transform** menu. Now in the dialog box select  $A$  as its own image, select  $D$  as the image of  $B$  and select  $F$  as the image of  $C$ . With this action, we create a self-similar triangle with the scale factor of one-half to be placed on the top part of our original triangle. However, as we can observe in Figure 2.b we need to add two more small replicas there, one on the left side and the other on the right side of the triangle. For this, we should add two new mappings. For the first mapping, we use points  $D$ ,  $B$ , and  $E$ . We click on “Structure” and choose **Add New Map**. Then we click on  $D$ ,  $B$ , and  $E$  to be the images of  $A$ ,  $B$ , and  $C$ , respectively. For the second mapping, after choosing “Structure” and **Add New Map**, we click on  $F$ ,  $D$ , and  $C$  to be the images of  $A$ ,  $B$ , and  $C$ , respectively.

We now set our iterations to only show non-point images by going to “Structure” and clicking on **Non-Point Images**. On “Display”, we click on **Final Iteration Only**. A similar figure to Figure 2.c appears.

We only want to see the most recent iterated images and be able to increase the stages of iterations. For this, click on the center of the interior of the original triangle and “Hide Triangle”. This will result in one of the stages of the Sierpinski Gasket. By selecting the entire figure and using the **Plus (+)** and **Minus (-)** keys, we can experiment with different stages of iterations. Now we can click on the **Point** tool, go to the **Edit** menu, and “Select All Points”. Then go to the **Display** menu and “Hide Points”. Our Sierpinski Gasket is ready! (Figure 2.d)

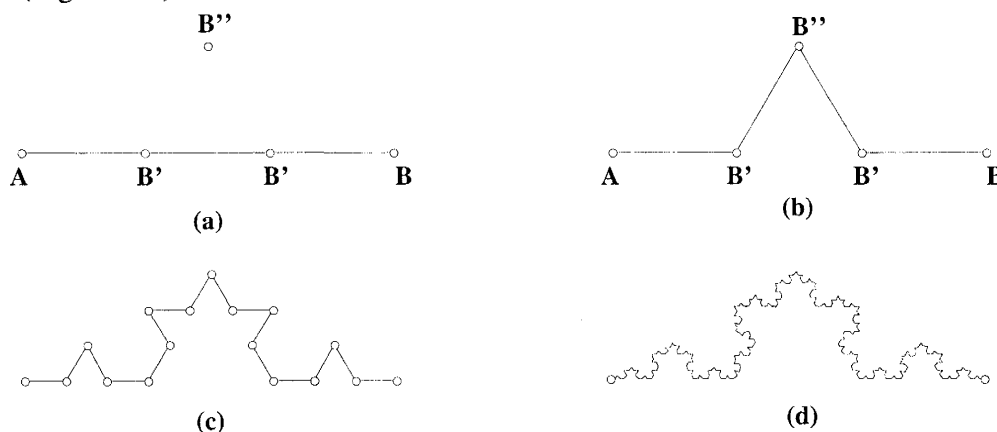


**Figure 2.** The steps of the construction of a Sierpinski Gasket using Sketchpad.


### Constructing a Koch Curve and Koch Snowflake Using Sketchpad

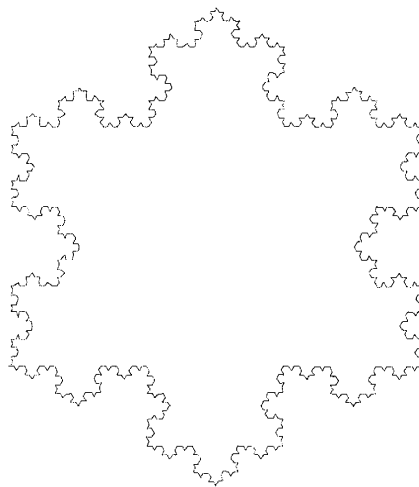
Construct  $\overline{AB}$  in Sketchpad. Select point  $A$  and choose “Mark Center”. Now dilate  $B$  one time  $1/3$  and the other time  $2/3$  to divide  $\overline{AB}$  into three congruent segments. Now we choose left  $B'$  and “Mark Center”. Then we select right  $B'$  and rotate  $60^\circ$  to find  $B''$  (Figure 3.a). We construct  $\overline{AB'}$ ,  $\overline{B'B''}$ ,  $\overline{B''B'}$ , and  $\overline{B'B}$  and then hide  $\overline{AB}$ . The curve  $\overline{AB'B''B'B}$  is the first generation of the Koch Curve (Figure 3.b). Now we are ready to iterate on  $A$  and  $B$ . For this, select  $A$  and  $B$  in order and click on “Iterate” in the **Transform** menu. In the dialog box, select  $A$  as its own image and select left  $B'$  as the image of  $B$ . With this action, we plan to create a self-similar curve to the first generation

curve with endpoints at  $A$  and  $B'$ . Now we click on “Structure”, choose **Add New Map**, and repeat the process for  $\overline{AB'}$  to map the curve to  $\overline{B'B''}$ ,  $\overline{B''B'}$ , and  $\overline{B'B}$  (Figure 3.c). We are almost ready to “Iterate”. However, we notice that segments generated from a stage will stay in the next generation, which is not right. To remedy this problem, we choose **Final Iteration Only** from the “Display” in the dialog box. Now we click on “Iterate” and hide the four segments of the first generation Koch Curve. Our new Koch Curve is ready! (Figure 3.d).



**Figure 3.** The steps of the construction of a Koch Snowflake using Sketchpad.

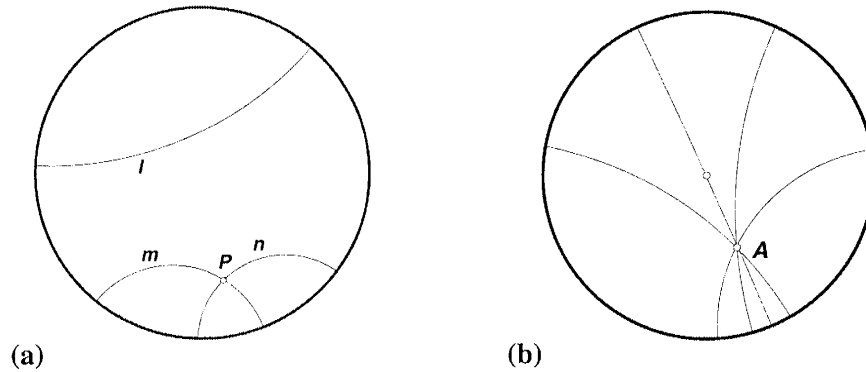
To construct the Koch Snowflake we use the Koch Curve as a “Custom Tool”. For this, after the last step of our Koch Curve we select all the objects and click on the *Custom Tools* menu . Choose “Create New Tool” and enter the name *Koch Curve Maker* and click OK. Now construct the equilateral triangle  $\Delta ABC$  and use this tool to construct the Koch Curve over each side of the triangle and hide those sides. Your Koch Snowflake construction will be complete (Figure 4).



**Figure 4.** A Koch Snowflake which has been constructed using a Custom Tool.

### The Poincaré Disk Model for Studying Hyperbolic Geometry

In the Poincaré model we start with a circle,  $C$ , in the Euclidean plane. The boundary of the circle does not really exist, and distances become distorted in this model. All the points in the interior of the circle are part of the hyperbolic plane. In this plane, two points lie on a “line” if the “line” forms an arc of a circle orthogonal to  $C$ . The only hyperbolic lines that are straight in the Euclidean sense are those that are diameters of the circle.



**Figure 5.** (a) *The hyperbolic parallel property in the Poincaré model, (b) A line in the Poincaré disk becomes straight in the Euclidean sense when it passes through the center of the disk.*

The model is conformal: The angle between two lines is the measure of the Euclidean angle between the tangents drawn to the lines at their points of intersection.

There are two software utilities available on the internet that affords students an opportunity to experience the properties of the geometry dynamically. There are two online course modules for hyperbolic geometry that are based on these two utilities [1]. The first utility software, NonEuclid, can be found at <http://cs.unm.edu/~joel/NonEuclid/>. “NonEuclid creates an interactive environment for learning and exploring non-Euclidean geometry on the high school and undergraduate level. The software package includes explorations, activities, and strategies for incorporating non-Euclidean geometry into the high school curriculum.” NonEuclid was created by John C. Polking as part of an Advanced Mathematics course in the High School Teachers Program at Rice University.

The second utility has been developed by Mike Alexander who has created several script tools for the Geometer’s Sketchpad that allows students to study aspects of hyperbolic geometry using the Poincaré model. These scripts are on the Internet and can be downloaded and used with the Geometer’s Sketchpad 3.0 and 3.05 for both Macintosh and Windows. The website is to be found at

<http://mathforum.org/sketchpad/gsp.gallery/poincare/poincare.html>.

The one disadvantage of NonEuclid is that the center of the Poincaré circle is not shown and lines cannot be moved once they are positioned in the plane. In the Sketchpad model, lines are easily moved using the point tool, and students can observe the effects of changing the position of points, moving lines, and changing the sizes of angles. However, NonEuclid has the advantage that the student can reflect geometric figures, while the Geometer’s Sketchpad scripts do not offer this facility.

## Reference

1. Anileen Gray and Reza Sarhangi, Modules for Non-Euclidean Geometries, <http://www.towson.edu/~gsarhang/geometry.html>