The Computational Beauty of Nature – Spring 2020

Lab 2: Generating Fractals with L-Systems

In this lab, you will experiment with L-sytems and use them to create a new fractal of your own design. To get started, open the LSystem.jar program, which is available under the Applications menu. IMPORTANT: For the starred exercises (**), write down your axioms, rules, and other parameter settings on a piece of paper, as these exercises will be due as part of your next homework assignment.

1. The simulator is initially set up to draw the Koch curve, with the axiom and rule shown in the green portion of the window. The number to the right of the Rules field is the iteration level (initially 0). Clicking on *+Iter* or *–Iter* increases or decreases the iteration level, and *Reset* resets it back to 0. Try out these buttons once or twice to see their effect. When the *Show String* option is checked, the bottom of the window shows the current L-system string corresponding to the drawn image.

2. To better understand how L-systems work, choose the *Simple Line* option from the pull-down menu in the top left corner. This system consists of the axiom X and the rule X = FX (the simulator uses the symbol = in place of the more traditional →). The turtle starts at position (0, –200) in the lower part of the window (position (0, 0) is at the center), with a heading of 90 degrees (due north). Iterate the system and observe how the string and image change on each step. Each iteration adds one more F to the string, causing the turtle to move forward one more step. The *Length* parameter specifies the step size in pixels.

3. Now let's add a small turning angle on each step. Click *Reset*, then change the rule to X = F+X, and the *Turn Angle* parameter to 30 (degrees). IMPORTANT: make sure to hit Return or click *Reset* after changing the field values, in order for the new values to take effect. Iterate a few times to see the result. What happens if you make the turn angle larger? What about smaller?

4. Reset the angle to 30, and click the *Turn Noise* checkbox. This setting adds a small amount (±10%) of uniform random "noise" to the turning angle each time the turtle executes a + or – command. What happens to the resulting image as you iterate? Also try toggling *Turn Noise* on and off a few times. For another interesting effect, temporarily turn the *Show String* option off and change the iteration level to a large number (say, 100 or 1000), then toggle *Turn Noise* on and off. Make sure you understand what is going on.

5. In a similar way, the *Scale Noise* setting adds random noise to the step size each time the turtle executes a movement command (F, G, or |). Experiment with toggling this setting on and off in conjunction with the *Turn Noise* setting. How does the visual effect of turning noise differ from that of scaling noise?

6. Now turn off all noise and reset the turning angle to 30 degrees. Change the *Length* (step size) to 200, and the *Scale Factor* to 0.9. Iterate a few times to see the effect of using a smaller scaling factor. How would you vary the parameters to create a tighter, smoother spiral?

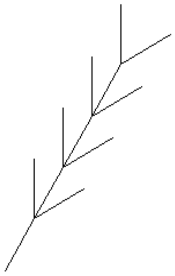
7. The G command moves the turtle forward one step, but without drawing a line. Insert a G into the rule immediately after the F, so that each time the turtle draws a line segment, it skips ahead by an equal amount, in effect producing a dashed line. You will need to reduce the *Length* of each step by one half in order to maintain the same overall image size. Try creating a few dashed lines, circles, and spirals of various sizes.

8. Now let's create a simple plant. Choose *Simple Line* from the menu to reset the parameters to draw a straight vertical line, as before. Also, make sure *Show String* is turned on. We can add “spines” to the left side of the plant by changing the rule to X = F[–F]X and setting *Turn Angle* to 30. The **[** command tells the turtle to remember its current *x, y* position and heading, and the] command restores the most recently saved position and heading. Thus the [–F] substring causes the turtle to remember its current location, turn left 30 degrees, draw a spine, and then jump back to its original location, after which it can continue drawing from there. Iterate a few times and observe how the image and string change on each step.

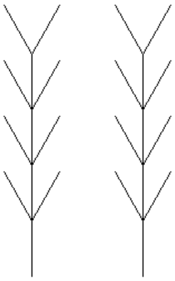
9. What happens if you omit the **[ ]** commands from the rule? (*i.e.* X = F–FX) Why?

10. Change the rule so that spines are drawn on the right side instead of the left. Then change it so that spines are drawn on both sides. What happens if you leave out the second pair of **[ ]**'s? Why?

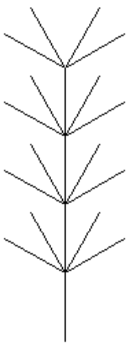
11. ** Suppose we want the entire plant to lean to the right, like the picture below. Without changing the *Heading* parameter, how would you change the axiom to produce this effect?



12. ** The axiom X generates a single copy of the plant. What if we want two copies of the plant growing side by side? Change the axiom so that the system produces the image below:



13. ** Modify the rule so that the plant has two spines on each side, at 30 and 60 degrees to the stem:

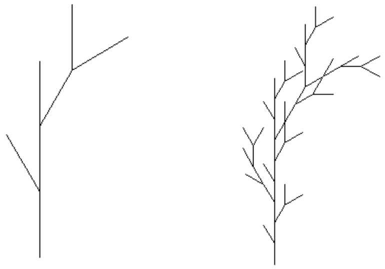


14. Now that you are familiar with the basic operation of L-systems, let's see how to draw more realistic plant structures. Let's start with a basic two-spined plant: axiom X, rule X=F[–F][+F]X, turn angle 30, scale factor 1, and segment length 100. Iterate this a few times and observe the resulting growth pattern. Notice that the plant gets extended by the same shape (generated by the commands F[–F][+F]) on each step, due to the presence of the X at the end of the rule. Instead of extending the plant linearly one step at a time, let's extend both branches simultaneously. Click *Reset* and change the rule to X=F[–X][+X]. This replaces the two inner F's, which previously drew single branches, with X's, which generate complete plants. The X at the end has also been removed, since we want the plant to grow from the branches rather than linearly. Iterate this rule a few times to see the new growth pattern.

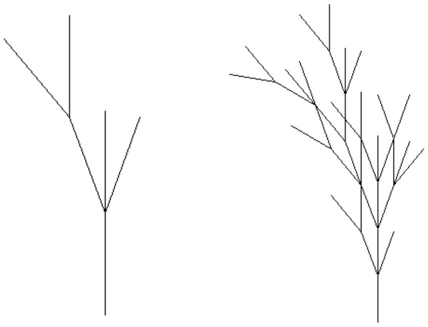
15. Now change the scale factor to 0.8. This reduces the size of each segment on each iteration, and produces a more realistic-looking plant. Also try adding some noise by toggling *Turn Noise* and *Scale Noise* on and off a few times. (Turning off *Show String* may speed up the iteration a bit.)

16. We can achieve exactly the same effect by using the axiom F and the rule F = | [–F][+F]. This rule uses F instead of X to repeatedly expand the two branches. In the previous rule, F was used to draw the stalk, which didn't get expanded, whereas the new rule uses a vertical bar | for the same purpose. In general, | acts just like F, but normally doesn't appear on the left-hand side of rules. Try out the new version of the rule to verify that it generates the same plant as before.

17. ** Using the axiom F, a turn angle of 30 degrees, a scale factor of 0.333, and a segment length of 300, design a rule to produce the following plant structure (the first two iterations are shown):



18. ** Using the axiom F, a turn angle of 20 degrees, a scale factor of 0.5, and a segment length of 300, design a rule to produce the following plant structure (the first two iterations are shown):



19. The pull-down menu contains all of the example fractals from pages 84-91 of *The Computational Beauty of Nature* (as well as a few others). Experiment with each of these by varying their parameter settings and tweaking their rules. In particular, try out the various Twig, Weed, Bush, and Tree L-systems.

20. ** Now create a fractal plant of your own design, and test it out in the simulator. To save a screen-shot of your fractal, press *Shift*+*Command*+4 and then click and drag the mouse to capture the image.