Eduard Pujol Puig
José Luis Pontón Martínez

A3DM Procedural modeling

Implementation

In this project we have created a tree generation algorithm based on L-Systems, we have implemented a Python script for Blender. The script contains a class (*LSystem*) responsible for reading and interpreting a set of rules to create a tree using Blender.

For parsing the grammar we use Lark, which is a Python library (included in the source files), it is similar to other parsing libraries such as ANTLR.

Trees are generated in two phases: branches and leaves. In order to generate the branches we create a set of edges representing the skeleton of the tree, then, we apply the skin modifier to give volume to the edges. We also use the subdivision modifier to get a more realistic result. To generate leaves we use metaballs.

Instructions

Next, we will define how to create different rules:

First, it is necessary to indicate if the algorithm will mark interior vertices as *loose* for the skin modifier. We encountered some problems when branches had very close angles, this option helps to fix the problem, however, it can deform the tree. To enable or disable this write at the beginning of the rule: *loose: True* or *loose: False.*

Then, it is specified the starting call, for this, write the rule name and the initial arguments between brackets. Next, all rules are defined.

For each rule, write the rule name and the arguments between brackets, inside the body of the function the following operations can be performed:

- Rule call: the same or another rule can be called using its name and the arguments between brackets.
- $\wedge(length, radius)$: advance operator (branch): it will advance (length) and create a vertex, then, it creates an edge between the last and the current vertices. Radius is assigned to the edge and used by the skin modifier.
- '$(length, radius)$: advance operator (leaf): it will advance (length) and create a metaball with the radius specified.
- %$(angle)$: rotates the direction used for the advance operator. It rotates around the direction axis (roll).
- \$$(angle)$: rotates the direction used for the advance operator. It rotates around the perpendicular direction axis (pitch).
- [: push stack operator.

-]: pop stack operator.

In addition to the operations defined, it is also possible to use conditionals, the structure is the following:

if **binary expression** *:*
Instructions
;

or

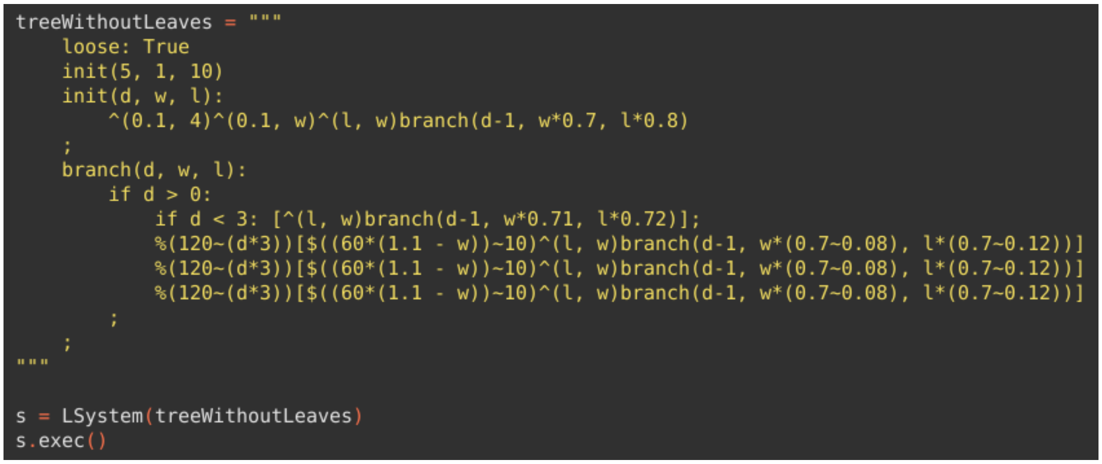
if **binary expression** *:*
Instructions
else:
Instructions
;

Any number expression accepts the operators +, - *, /. Moreover, to add noise to a value use the following structure: **number∼deviation**, this will return numbers from $number - deviation$ to $number + deviation$.

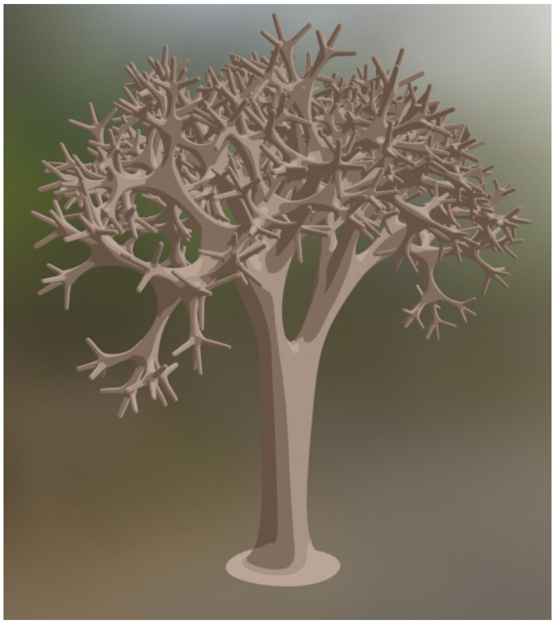
Finally, all rules must be ended by **;**

Examples

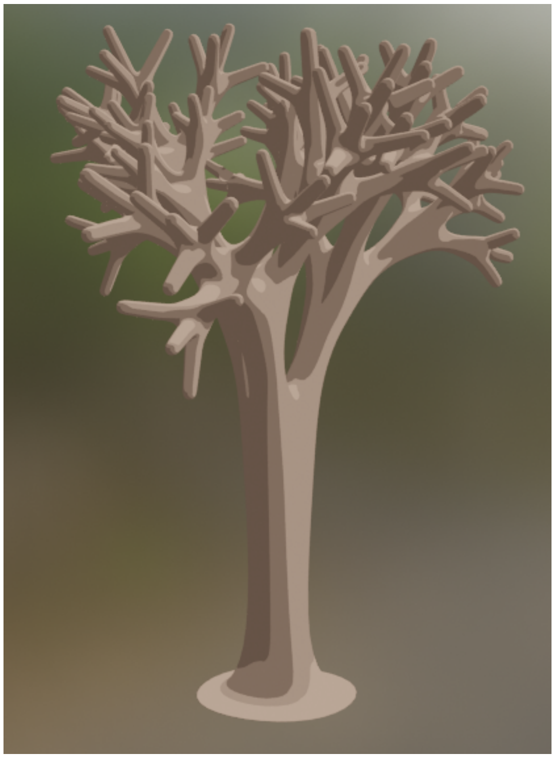
Tree without leaves



```
treeWithoutLeaves = """
    loose: True
    init(5, 1, 10)
    init(d, w, l):
        ^(0.1, 4)^(0.1, w)^(l, w)branch(d-1, w*0.7, l*0.8)
    ;
    branch(d, w, l):
        if d > 0:
            if d < 3: [^(l, w)branch(d-1, w*0.71, l*0.72)];
            %(120~(d*3))[$((60*(1.1 - w))~10)^(l, w)branch(d-1, w*(0.7~0.08), l*(0.7~0.12))]
            %(120~(d*3))[$((60*(1.1 - w))~10)^(l, w)branch(d-1, w*(0.7~0.08), l*(0.7~0.12))]
            %(120~(d*3))[$((60*(1.1 - w))~10)^(l, w)branch(d-1, w*(0.7~0.08), l*(0.7~0.12))]
        ;
    ;
"""

s = LSystem(treeWithoutLeaves)
s.exec()
```

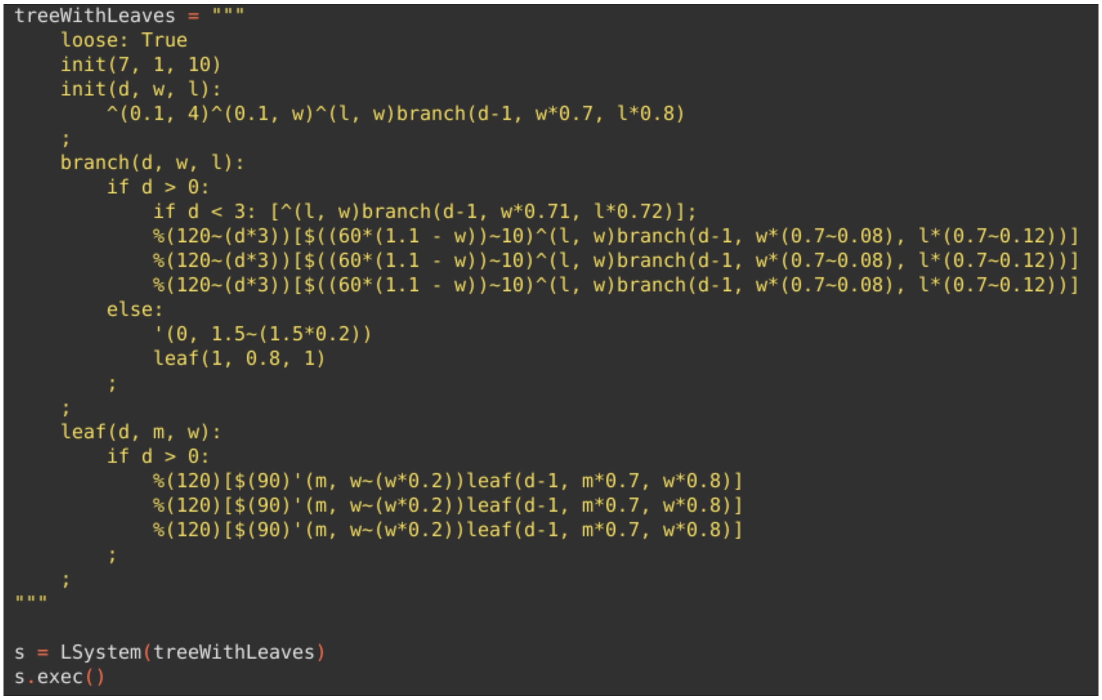


(a) d = 7

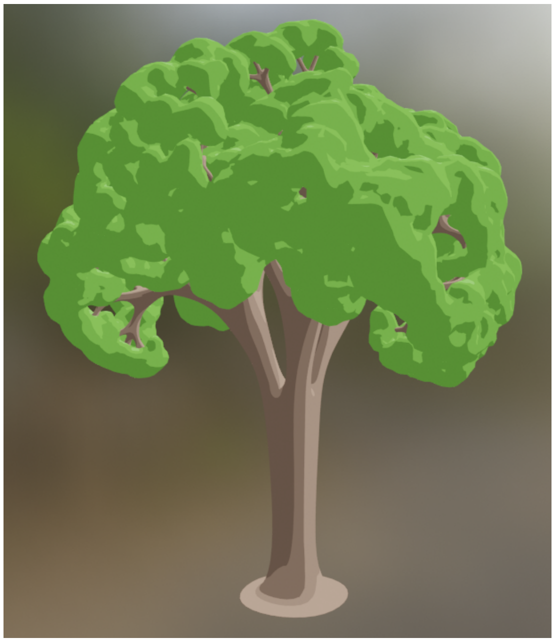


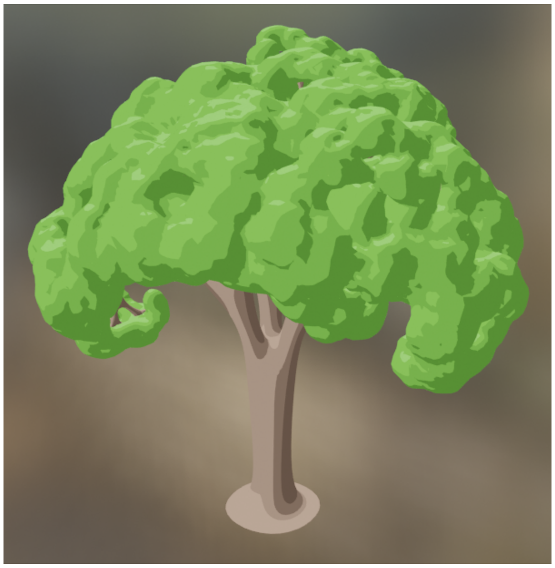
(b) d = 5

Tree with leaves

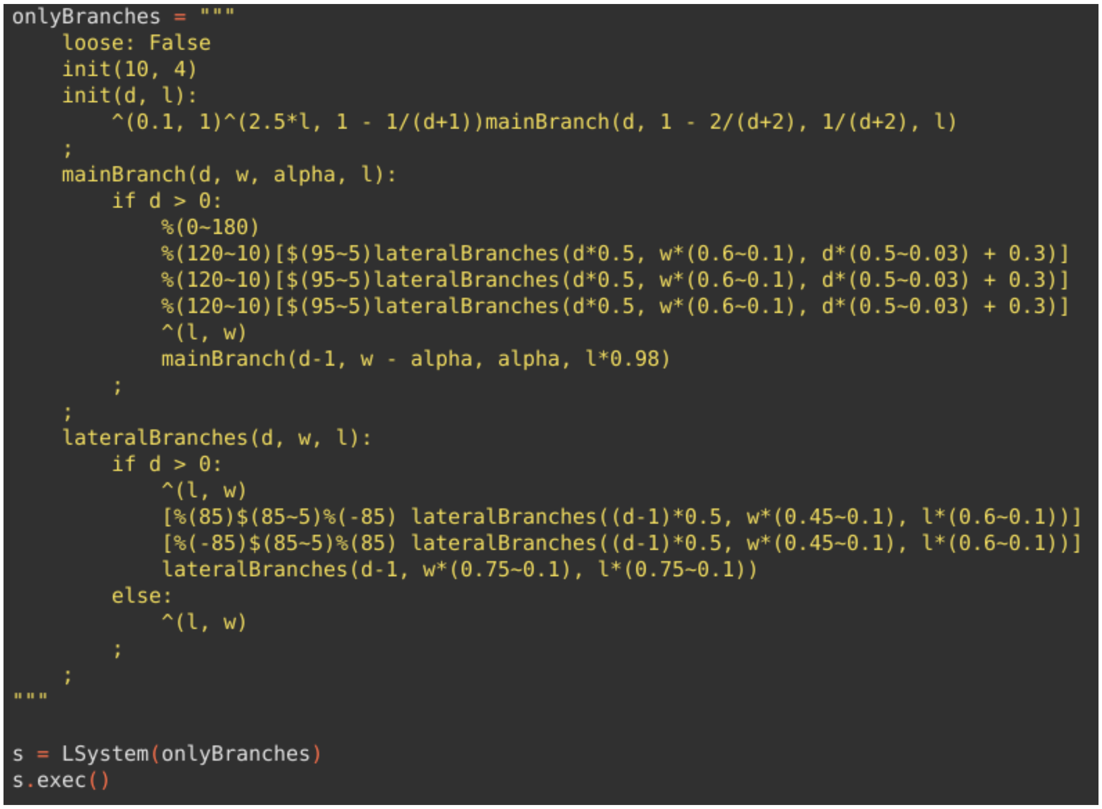


```
treeWithLeaves = """
    loose: True
    init(7, 1, 10)
    init(d, w, l):
        ^(0.1, 4)^(0.1, w)^(l, w)branch(d-1, w*0.7, l*0.8)
    ;
    branch(d, w, l):
        if d > 0:
            if d < 3: [^(l, w)branch(d-1, w*0.71, l*0.72)];
            %(120~(d*3))[$((60*(1.1 - w))~10)^(l, w)branch(d-1, w*(0.7~0.08), l*(0.7~0.12))]
            %(120~(d*3))[$((60*(1.1 - w))~10)^(l, w)branch(d-1, w*(0.7~0.08), l*(0.7~0.12))]
            %(120~(d*3))[$((60*(1.1 - w))~10)^(l, w)branch(d-1, w*(0.7~0.08), l*(0.7~0.12))]
        else:
            '(0, 1.5~(1.5*0.2))
            leaf(1, 0.8, 1)
        ;
    ;
    leaf(d, m, w):
        if d > 0:
            %(120)[$(90)'(m, w~(w*0.2))leaf(d-1, m*0.7, w*0.8)]
            %(120)[$(90)'(m, w~(w*0.2))leaf(d-1, m*0.7, w*0.8)]
            %(120)[$(90)'(m, w~(w*0.2))leaf(d-1, m*0.7, w*0.8)]
        ;
    ;
"""

s = LSystem(treeWithLeaves)
s.exec()
```



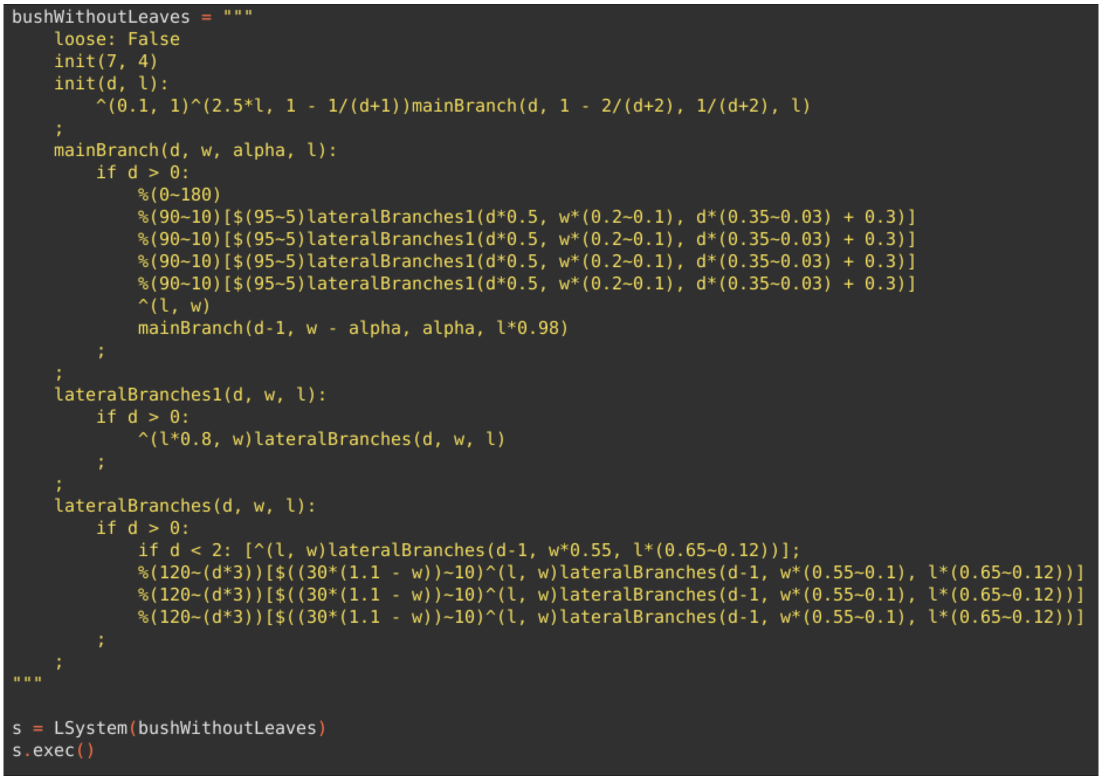


Only branches

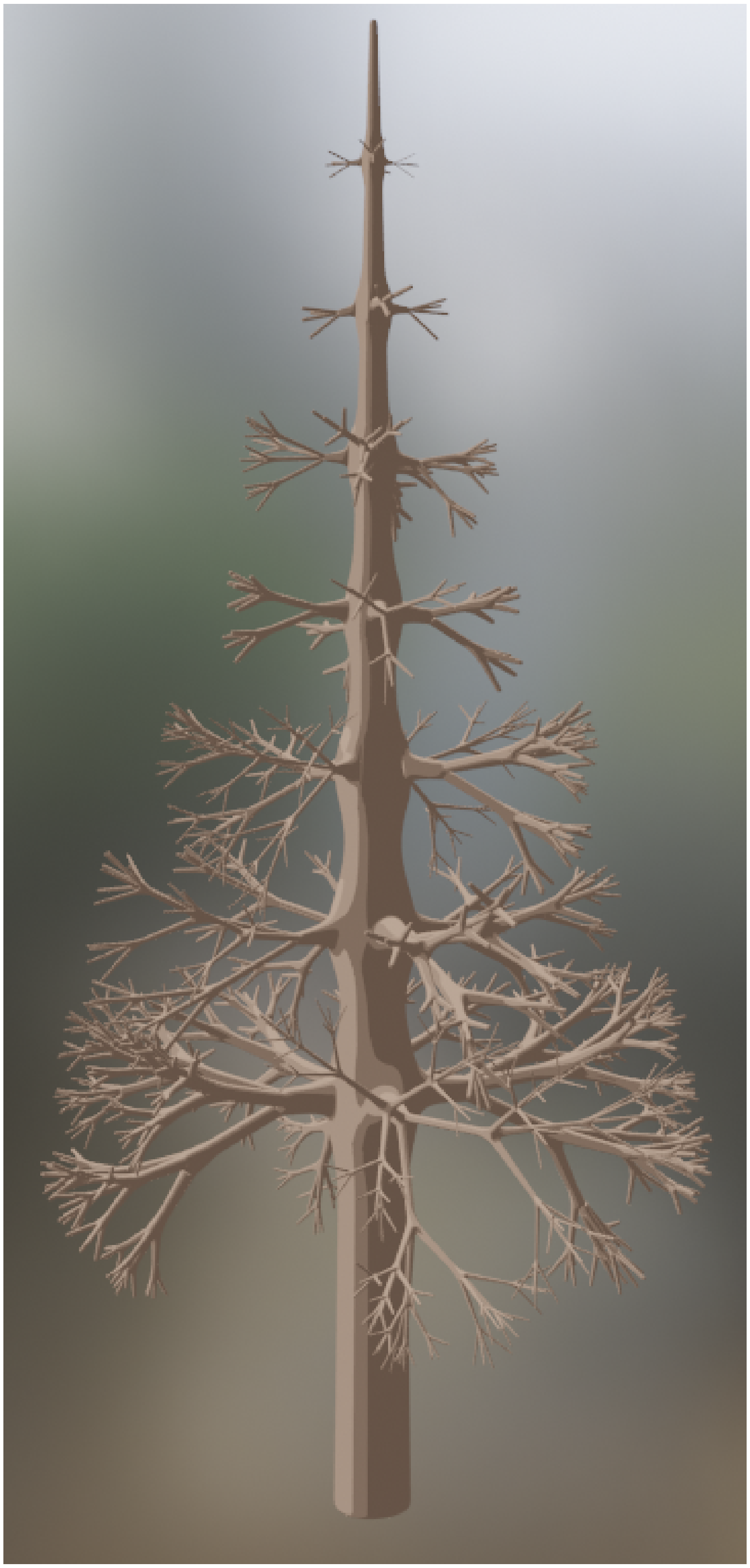


```
onlyBranches = """
    loose: False
    init(10, 4)
    init(d, l):
        ^(0.1, 1)^(2.5*l, 1 - 1/(d+1))mainBranch(d, 1 - 2/(d+2), 1/(d+2), l)
    ;
    mainBranch(d, w, alpha, l):
        if d > 0:
            %(0~180)
            %(120~10)[$(95~5)lateralBranches(d*0.5, w*(0.6~0.1), d*(0.5~0.03) + 0.3)]
            %(120~10)[$(95~5)lateralBranches(d*0.5, w*(0.6~0.1), d*(0.5~0.03) + 0.3)]
            %(120~10)[$(95~5)lateralBranches(d*0.5, w*(0.6~0.1), d*(0.5~0.03) + 0.3)]
            ^(l, w)
            mainBranch(d-1, w - alpha, alpha, l*0.98)
        ;
    ;
    lateralBranches(d, w, l):
        if d > 0:
            ^(l, w)
            [%(85)$(85~5)%(-85) lateralBranches((d-1)*0.5, w*(0.45~0.1), l*(0.6~0.1))]
            [%(-85)$(85~5)%(85) lateralBranches((d-1)*0.5, w*(0.45~0.1), l*(0.6~0.1))]
            lateralBranches(d-1, w*(0.75~0.1), l*(0.75~0.1))
        else:
            ^(l, w)
        ;
    ;
"""

s = LSystem(onlyBranches)
s.exec()
```

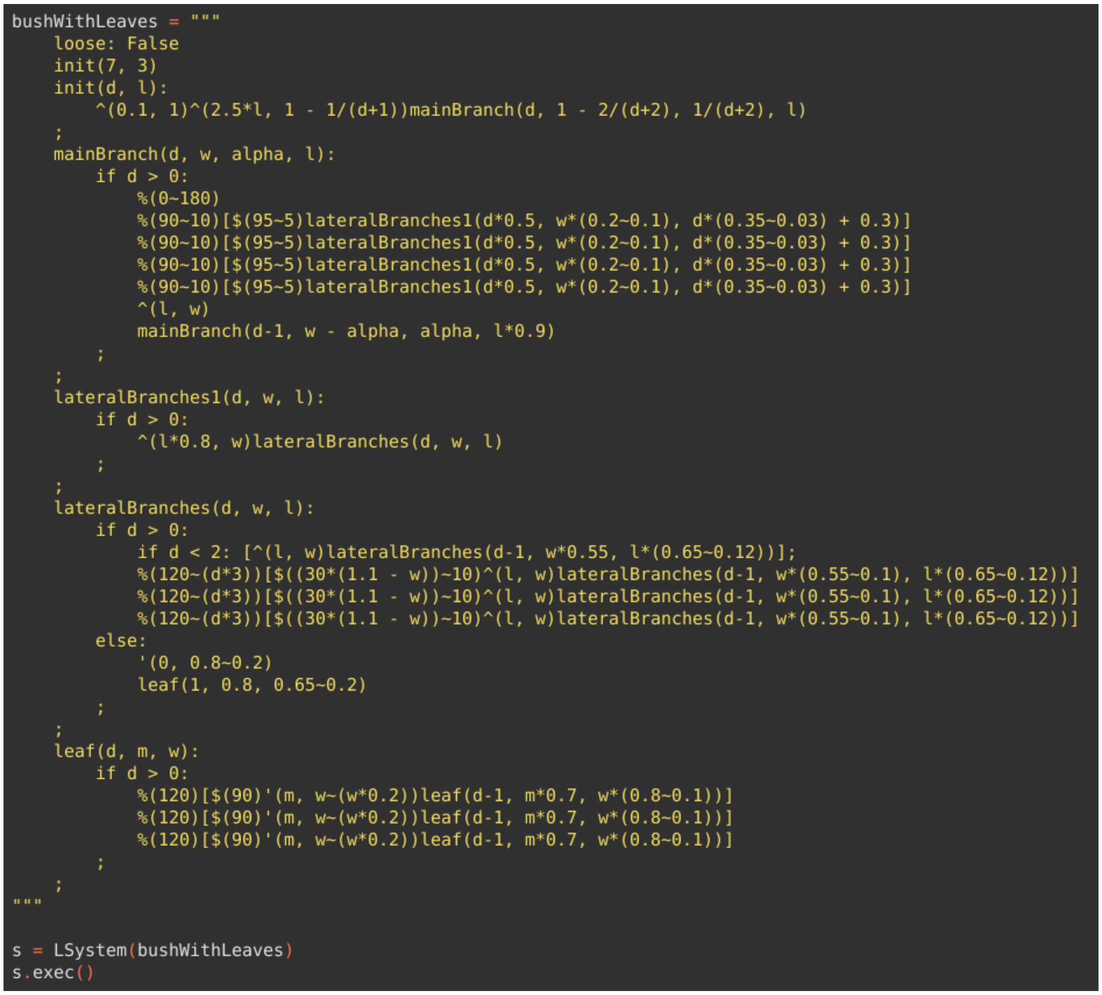
Bush without leaves



```
bushWithoutLeaves = """
    loose: False
    init(7, 4)
    init(d, l):
        ^(0.1, 1)^(2.5*l, 1 - 1/(d+1))mainBranch(d, 1 - 2/(d+2), 1/(d+2), l)
    ;
    mainBranch(d, w, alpha, l):
        if d > 0:
            %(0~180)
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            ^(l, w)
            mainBranch(d-1, w - alpha, alpha, l*0.98)
        ;
    ;
    lateralBranches1(d, w, l):
        if d > 0:
            ^(l*0.8, w)lateralBranches(d, w, l)
        ;
    ;
    lateralBranches(d, w, l):
        if d > 0:
            if d < 2: [^(l, w)lateralBranches(d-1, w*0.55, l*(0.65~0.12))];
            %(120~(d*3))[$((30*(1.1 - w))~10)^(l, w)lateralBranches(d-1, w*(0.55~0.1), l*(0.65~0.12))]
            %(120~(d*3))[$((30*(1.1 - w))~10)^(l, w)lateralBranches(d-1, w*(0.55~0.1), l*(0.65~0.12))]
            %(120~(d*3))[$((30*(1.1 - w))~10)^(l, w)lateralBranches(d-1, w*(0.55~0.1), l*(0.65~0.12))]
        ;
    ;
"""

s = LSystem(bushWithoutLeaves)
s.exec()
```



Bush with leaves



```
bushWithLeaves = """
    loose: False
    init(7, 3)
    init(d, l):
        ^(0.1, 1)^(2.5*l, 1 - 1/(d+1))mainBranch(d, 1 - 2/(d+2), 1/(d+2), l)
    ;
    mainBranch(d, w, alpha, l):
        if d > 0:
            %(0~180)
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            %(90~10)[$(95~5)lateralBranches1(d*0.5, w*(0.2~0.1), d*(0.35~0.03) + 0.3)]
            ^(l, w)
            mainBranch(d-1, w - alpha, alpha, l*0.9)
        ;
    ;
    lateralBranches1(d, w, l):
        if d > 0:
            ^(l*0.8, w)lateralBranches(d, w, l)
        ;
    ;
    lateralBranches(d, w, l):
        if d > 0:
            if d < 2: [^(l, w)lateralBranches(d-1, w*0.55, l*(0.65~0.12))];
            %(120~(d*3))[$((30*(1.1 - w))~10)^(l, w)lateralBranches(d-1, w*(0.55~0.1), l*(0.65~0.12))]
            %(120~(d*3))[$((30*(1.1 - w))~10)^(l, w)lateralBranches(d-1, w*(0.55~0.1), l*(0.65~0.12))]
            %(120~(d*3))[$((30*(1.1 - w))~10)^(l, w)lateralBranches(d-1, w*(0.55~0.1), l*(0.65~0.12))]
        else:
            '(0, 0.8~0.2)
            leaf(1, 0.8, 0.65~0.2)
        ;
    ;
    leaf(d, m, w):
        if d > 0:
            %(120)[$(90)'(m, w~(w*0.2))leaf(d-1, m*0.7, w*(0.8~0.1))]
            %(120)[$(90)'(m, w~(w*0.2))leaf(d-1, m*0.7, w*(0.8~0.1))]
            %(120)[$(90)'(m, w~(w*0.2))leaf(d-1, m*0.7, w*(0.8~0.1))]
        ;
    ;
"""

s = LSystem(bushWithLeaves)
s.exec()
```